Modularization of Research Software for Collaborative Open Source Development

The 9th International Conference on Advanced Collaborative Networks, Systems and Applications (COLLA 2019)

Rome, Italy
July 1, 2019

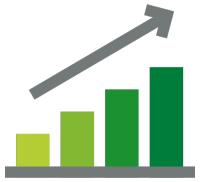
Christian Zirkelbach
Software Engineering Group, Kiel University

Introduction

Software continously evolving during lifetime

Open research software is constantly increasing [1]

Technical or organizational circumstances cause problems [2]



Current trend to move towards microservice architectures, caused by promised benefits [3]

ExplorViz

https://www.explorviz.net
https://github.com/ExplorViz

Live trace visualization for large software landscapes based on monitoring

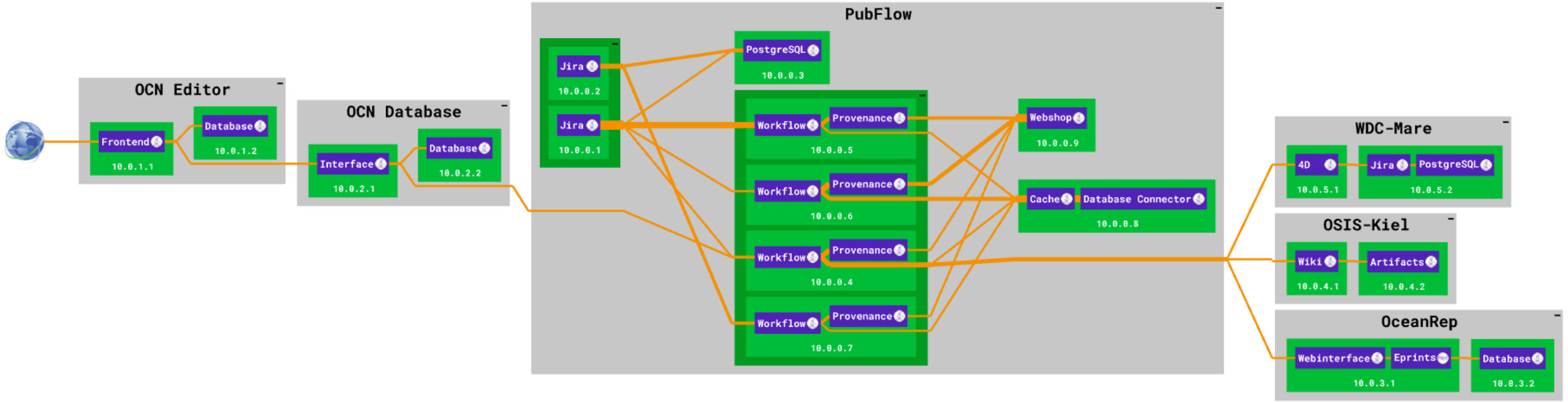
[5,6]

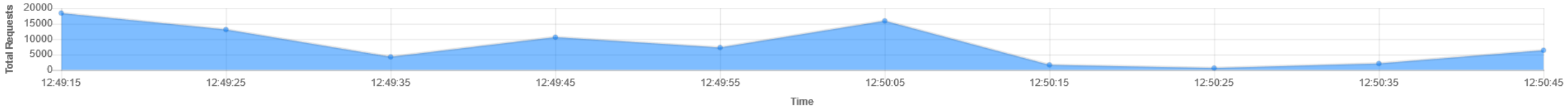
Program- and system comprehension for developers

Started as a Ph.D project in 2012

Free license (Apache License, Version 2.0) from the beginning

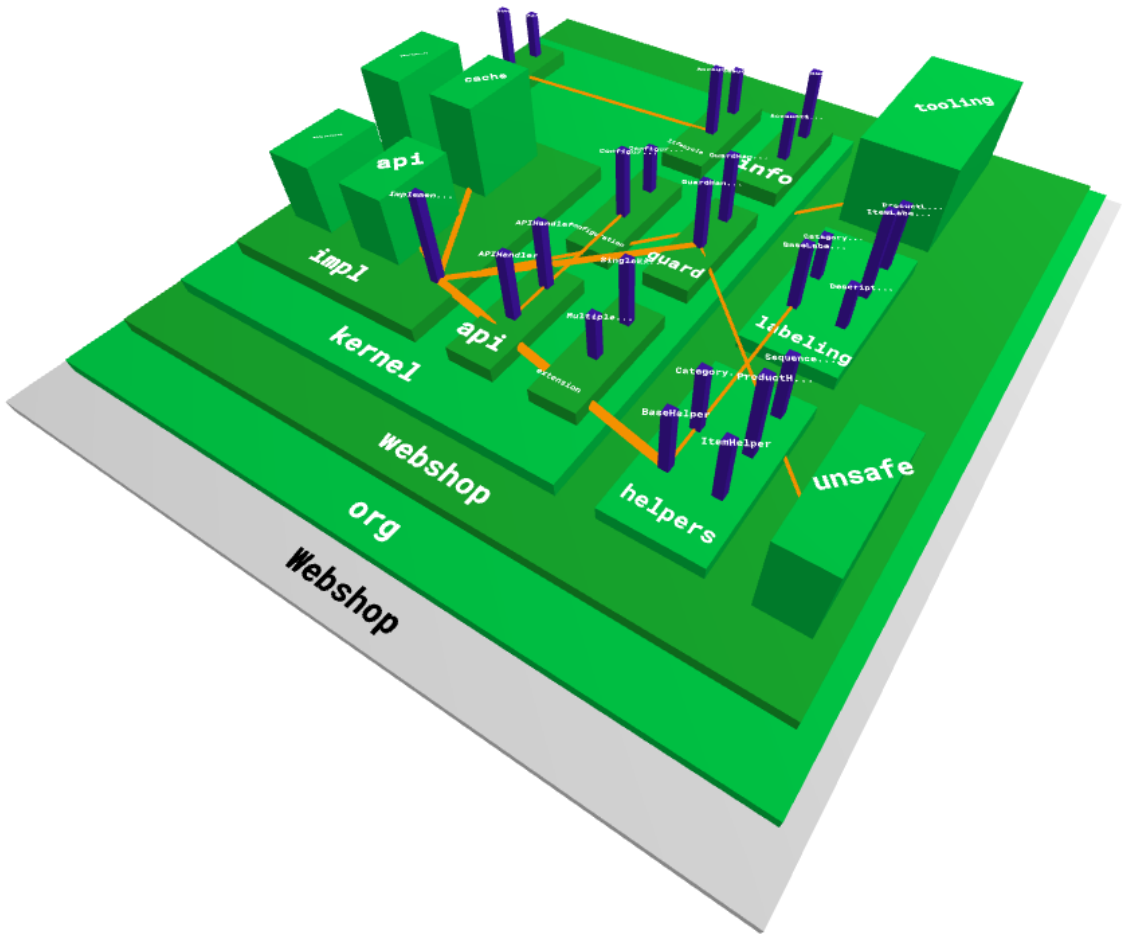
Repeateadly extended since the beginning

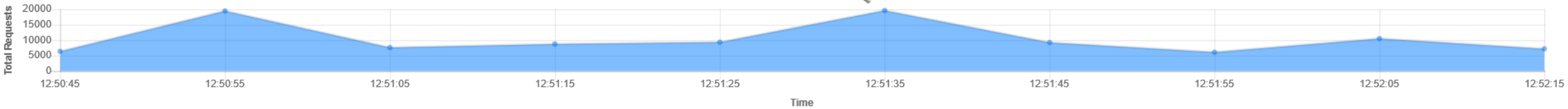




Open All Components

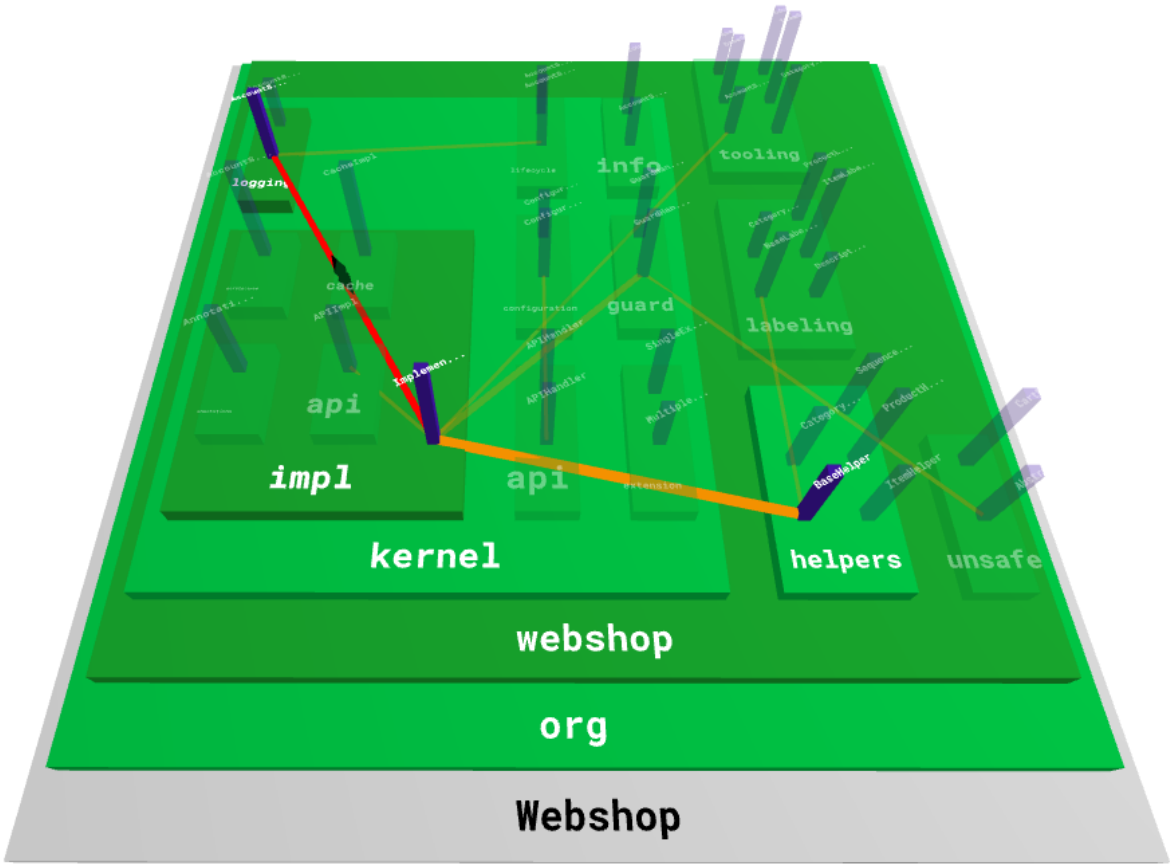
Search Entity...

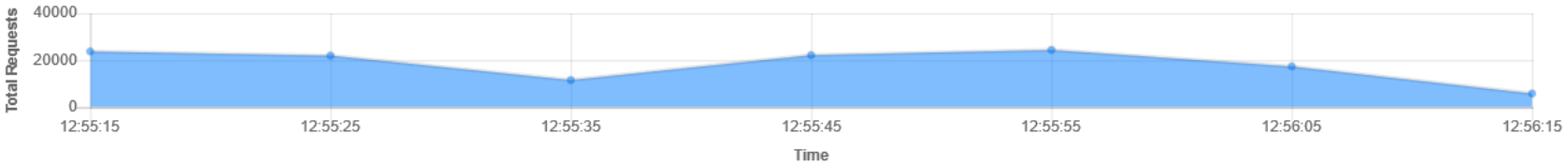




Open All Components

Search Entity...





Trace Selection

Trace ID	Soure Clazz	Target Clazz	Duration (ms)
1	BaseLabeler	Implementatio...	6.655
2	Implementat...	Implementatio...	3.766

Trace Replayer

Step **1** of 6

Trace Step Details

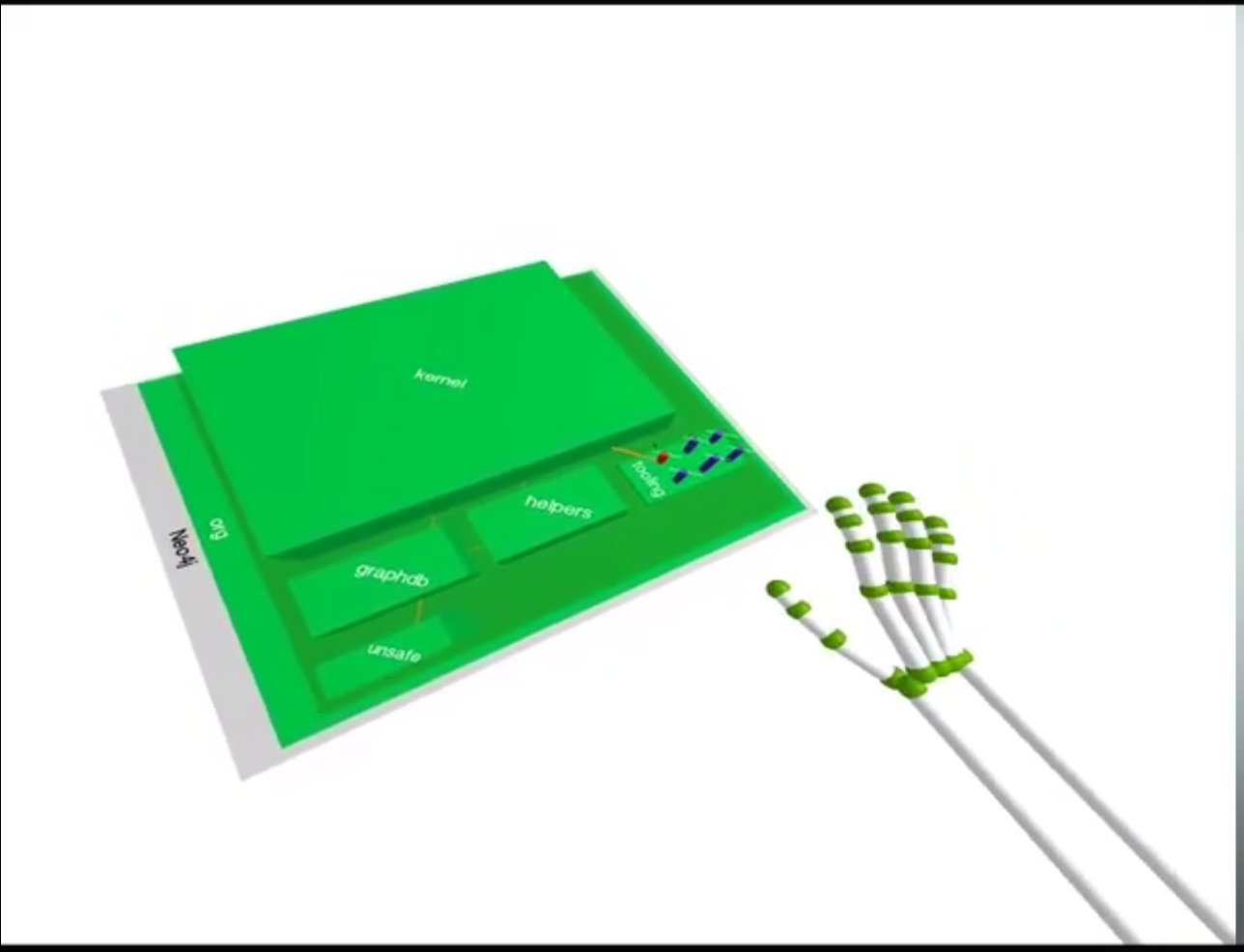
Soure Clazz	ImplementationHandler
Target Clazz	AccountSqlMapDao
Operation Name	getMethod8()
Requests	2.500
Avg. Resp. Time (ms)	982

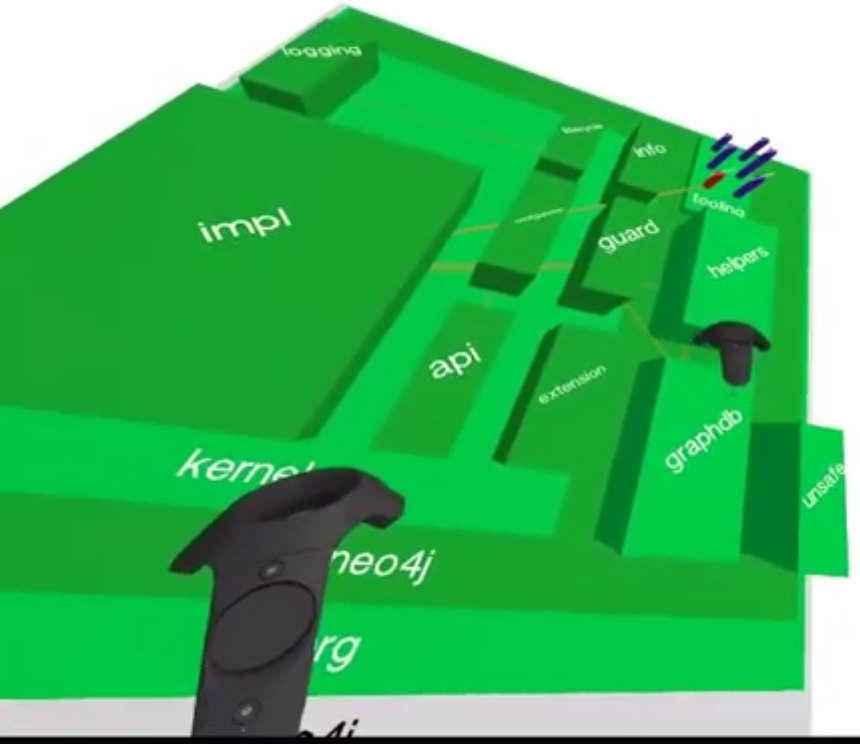
Database Queries

Time	Statement	Statement Type	Return Value
8:31:35	CREATE ...	Statement	null
8:31:35	INSERT ...	Statement	null
8:31:35	INSERT ...	Statement	null
8:31:35	INSERT ...	Statement	null
8:31:35	SELECT ...	Statement	55
8:31:35	SELECT ...	Statement	45
8:31:35	CREATE ...	Statement	null
8:31:35	INSERT ...	Statement	null
8:31:35	INSERT ...	Statement	null

Query Details

Time	14.3.2019, 08:31:35
Timestamp	1552548695474
Statement	CREATE TABLE IF NOT EXISTS `order` (oid integer PRIMARY KEY, name text NOT NULL, email text NOT NULL, odate text NOT NULL, itemid integer NOT NULL);
Type	Statement
Return Value	null
Response Time (ms)	61


kernel
helpers
tooling
org
Neo4j
graphdb
unsafe


logging
info
impl
guard
helpers
api
extension
graphdb
neo4j

Problem Statement

Problem Statement

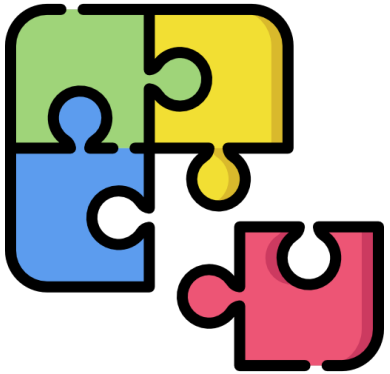
Several extensions since the beginning

Introduced a lot of new features

Software is realized on the Java-based Google Web Toolkit (GWT)

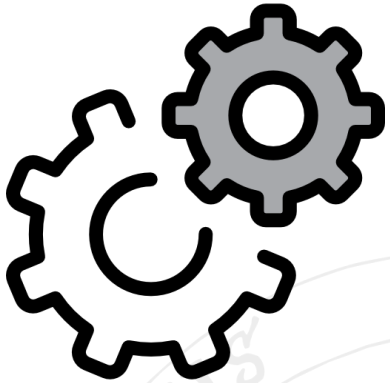
Result
Unstructured architecture due to an unsuitable collaboration and integration process

Introduced Problems



Extensibility
&
Integrability

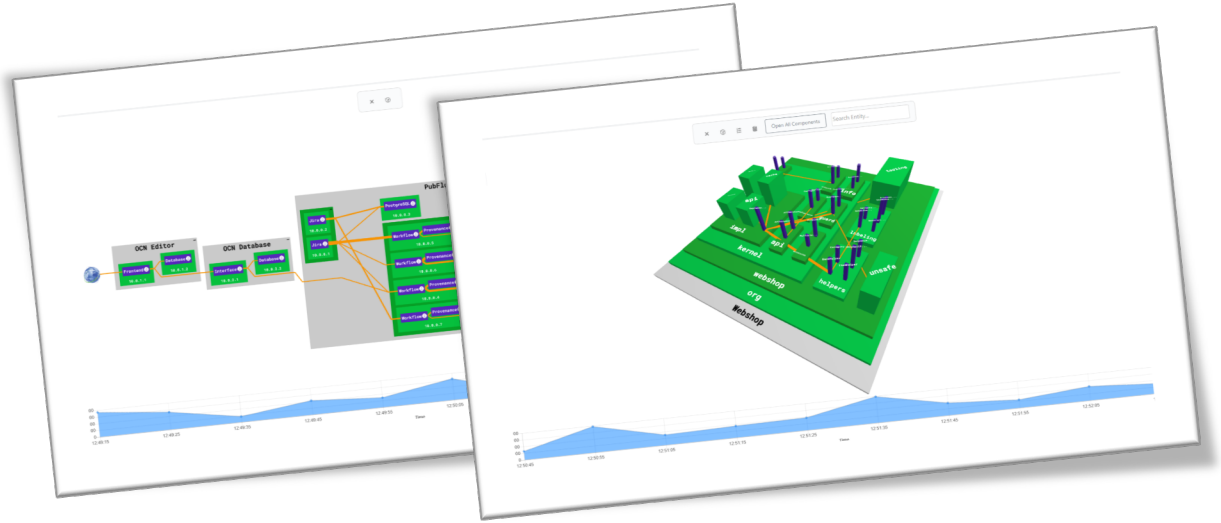
Code Quality
&
Comprehensibility



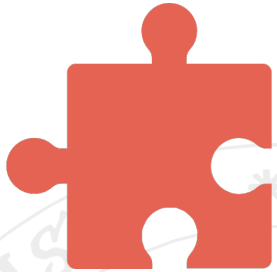
Software Configuration
&
Delivery

Extensibility & Integrability: Architecture

Logic (Core)

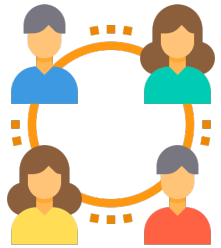


Visualizations



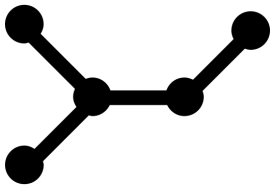
Extensions

Extensibility & Integrability: New Features

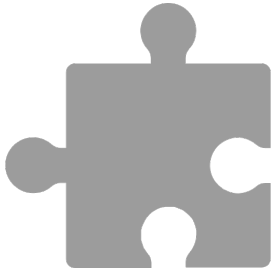


(Student) Collaboration

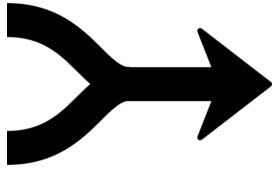
New Feature



New Git Branch



No extension mechanism



Merging Problematic

Single Codebase

Code Quality & Comprehensibility

- Modern JS web Frameworks became increasingly mature
- We used GWT's JavaScript Native Interface (JSNI) to embed JS functionality in client-related Java methods
- Integration of modern JS libraries to improve UX

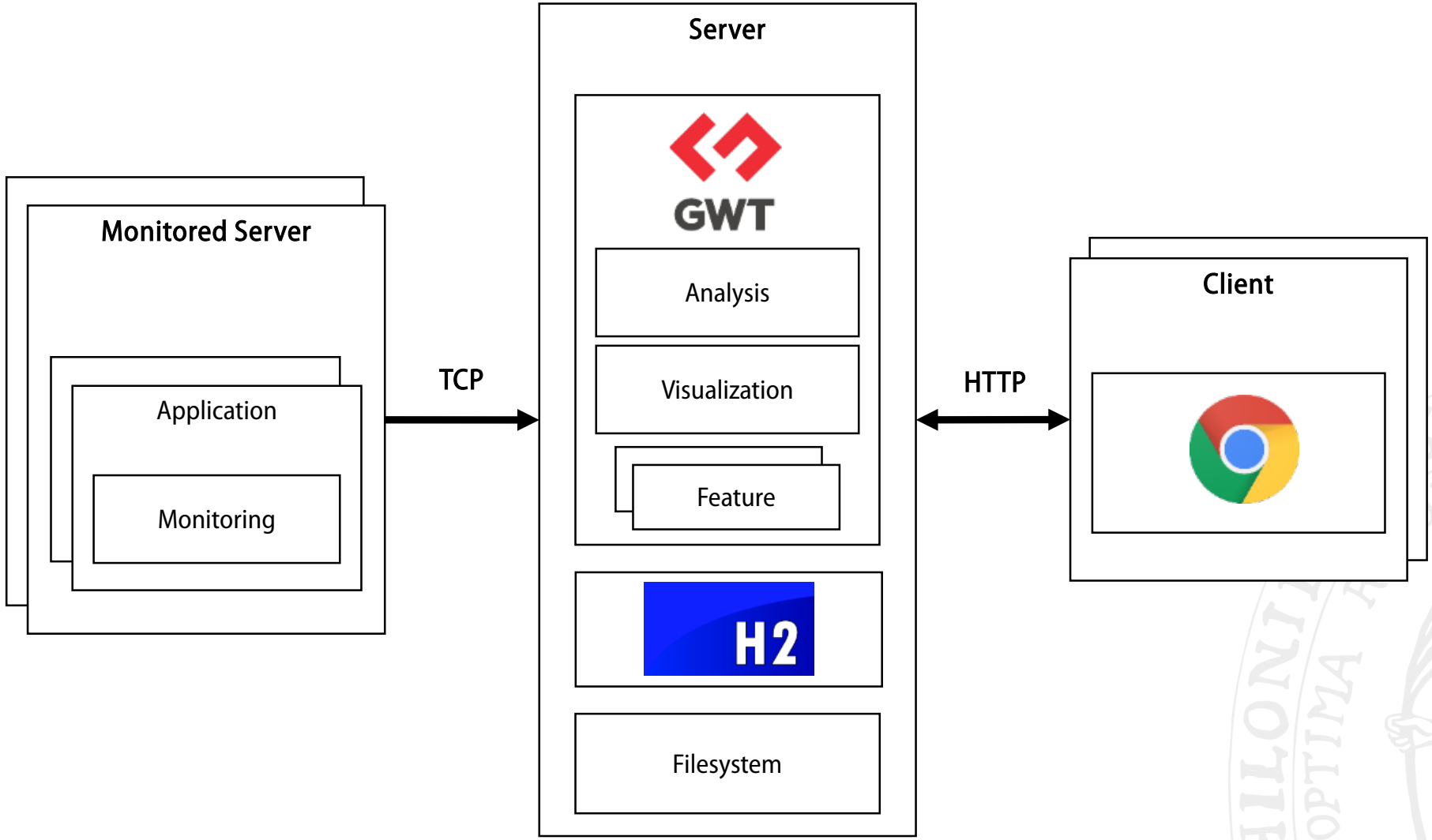
JSNI was planned to be removed in upcoming release version 3 of GWT

Software Configuration & Delivery

- Integrated features (extensions) were deeply coupled
- Users often did only need just a subset of features
  - ✓ Create branches for several use cases, e.g., a live demo

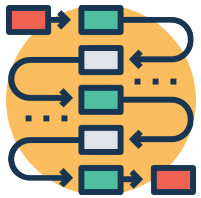
Problems worsened the extensibility, maintainability, and comprehension for developers

Legacy Architecture of ExplorViz


Server
GWT
Analysis
Visualization
Feature
H2
Filesystem
Monitored Server
Application
Monitoring
TCP
HTTP
Client

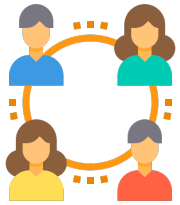
Modularization Process and Architecture of ExplorViz

Modularization Process and Architecture of ExplorViz



First step: Requirement Analysis

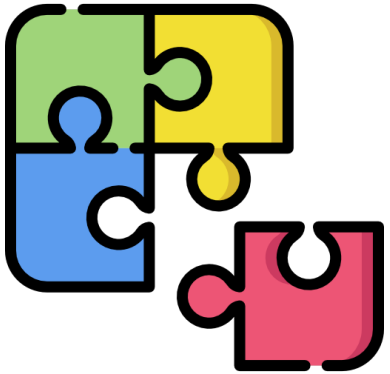
- Identification of obstacles
- Focus: Provide a **collaborative** development process [18]



Second step: Developer Meeting

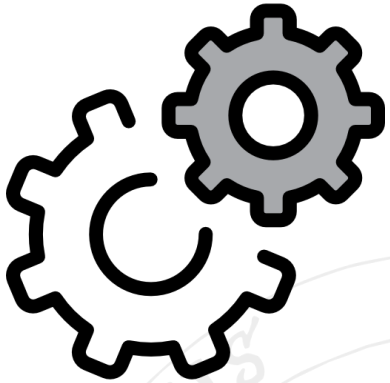
- Agreement to build upon a microservice architecture
- Architectural style features small, lightweight, and independent services

How did we address the problems?

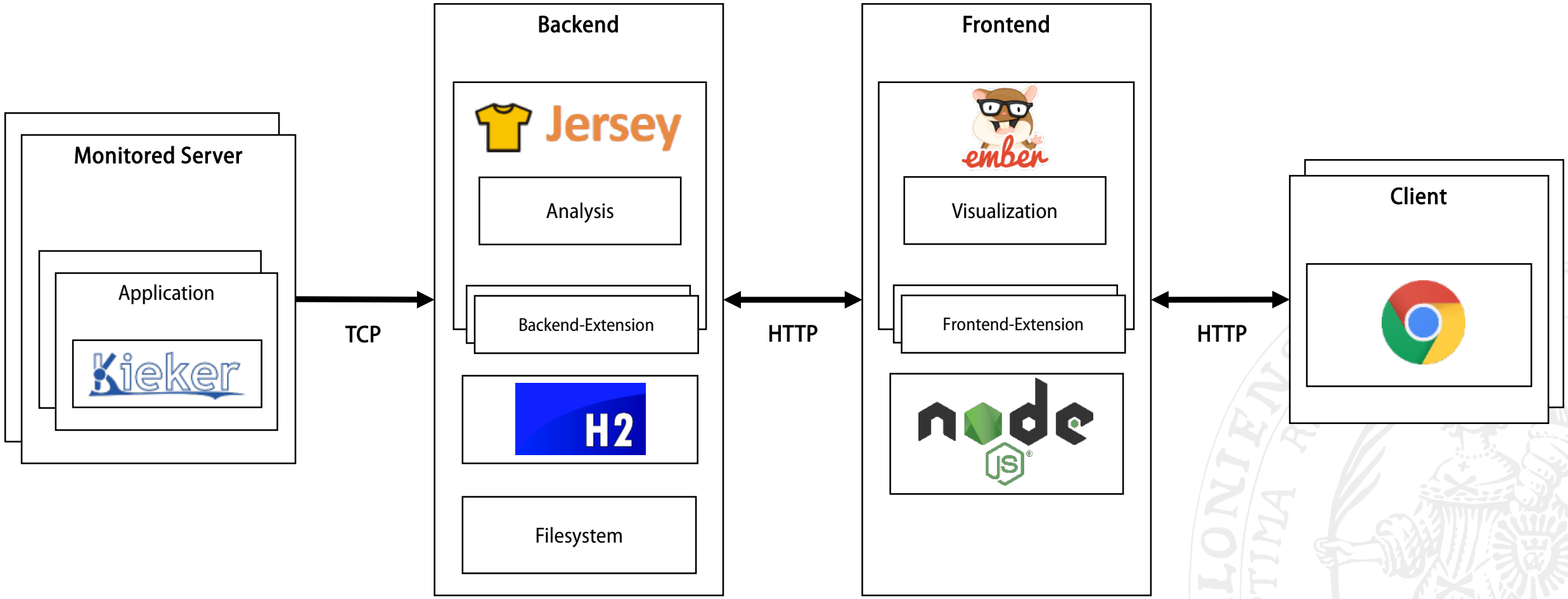


Extensibility
&
Integrability

Code Quality
&
Comprehensibility



Software Configuration
&
Delivery


Monitored Server
Application
Kieker
TCP
Backend
Jersey
Analysis
Backend-Extension
H2
Filesystem
HTTP
Frontend
ember
Visualization
Frontend-Extension
node
JS
HTTP
Client

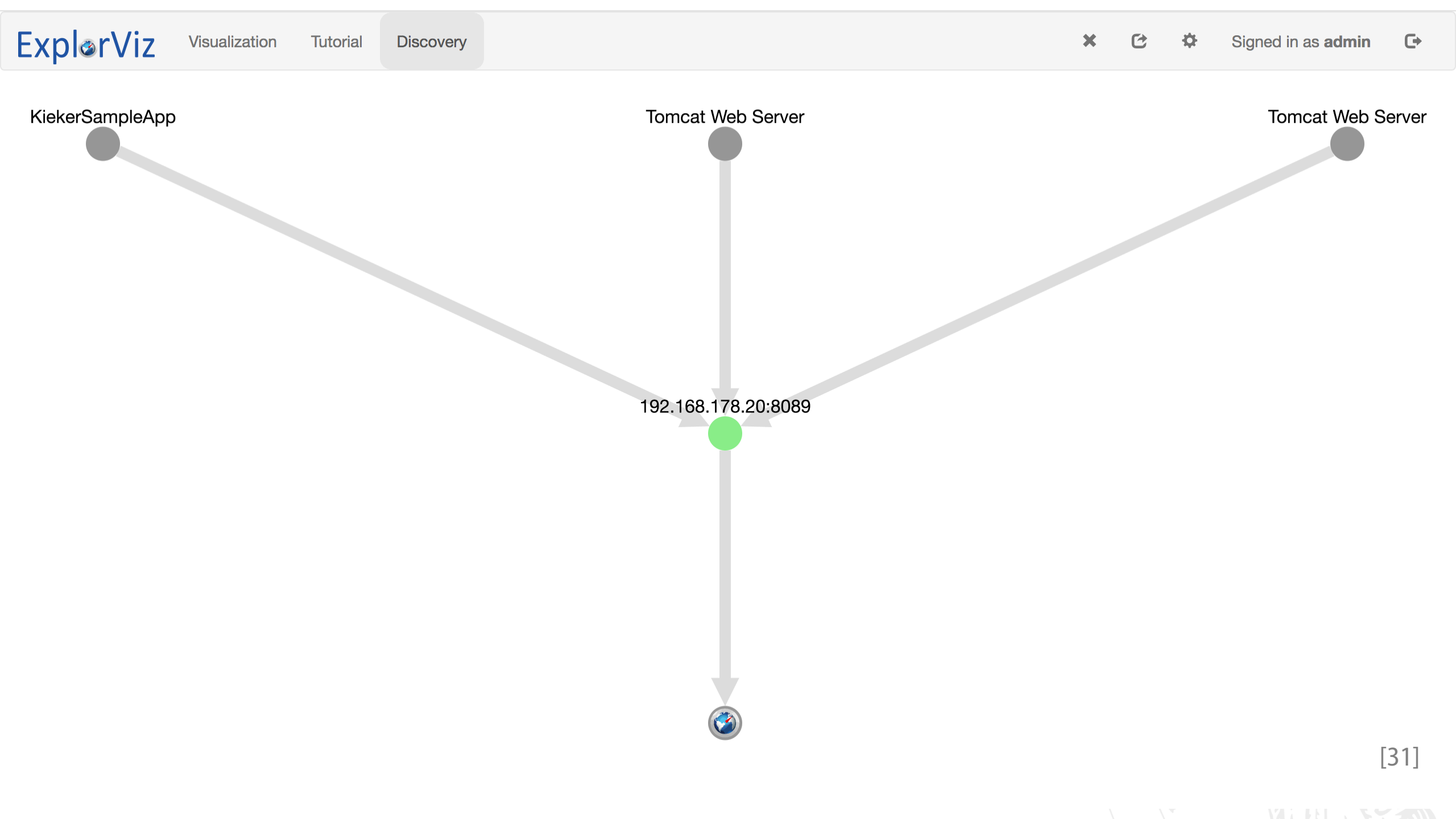
Proof-of-Concept Implementation

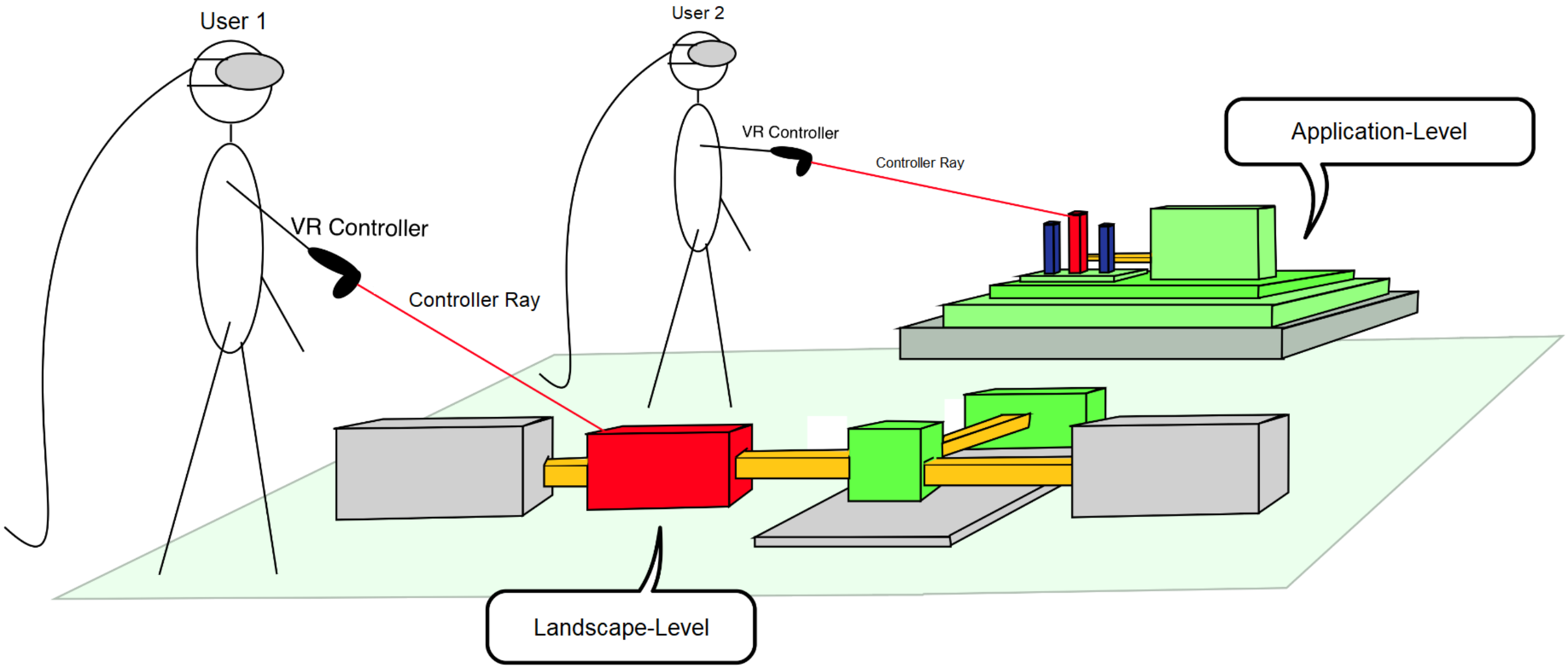
Proof-of-Concept Implementation

- Split the project as planned into two separate projects –
  a backend project Java-based Jersey framework,
  and a frontend project employing the JS web framework ember

- Both have large and active community and offer good documentation

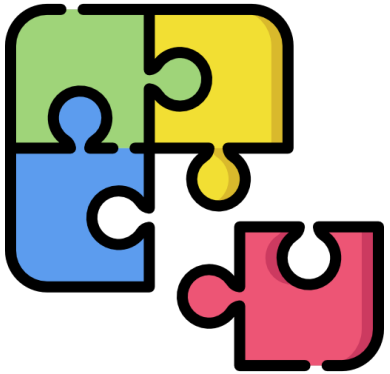
- Since the end of our process in early 2018,
  we successfully develop several extensions (backend and frontend)


ExplorViz
Visualization
Tutorial
Discovery
Signed in as admin
KiekerSampleApp
Tomcat Web Server
Tomcat Web Server
192.168.178.20:8089


User 1
User 2
VR Controller
Controller Ray
VR Controller
Controller Ray
Application-Level
Landscape-Level

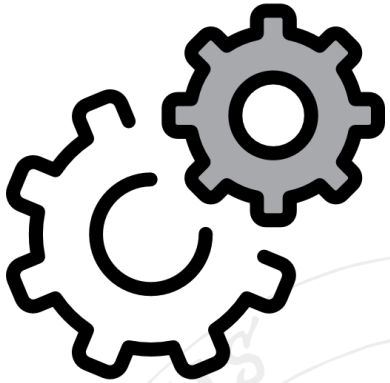
Restructured Architecture and New Process of ExplorViz

How can we improve the first iteration?



Extensibility
&
Integrability

Code Quality
&
Comprehensibility



Software Configuration
&
Delivery

Extensibility & Integrability: New Features

- Frontend extensions are based on Ember's addon mechanism
- The backend used the package scanning feature of Jersey to include extensions

Still a monolithic application with high coupling of modules

- ✓ Decoupling into separated microservices
- ✓ Data-Exchange: API-Gateway, based on NGINX

- Inter-service communication is now realized with APACHE kafka

Code Quality & Comprehensibility

- Improvements for code quality and accessibility, showed a perceptible impact on contributor's work

- Recurring students approved the easier access to ExplorViz and especially the obligatory exchange format `{ json:api }`

Still lacked a common code style in terms of conventions and best practices

- ✓ Compulsory rule sets for *Checkstyle*, *PMD*, and *Spotbugs*
- ✓ *ESLint* with an *Ember* community-driven rule set
- ✓ All tools are integrated into our CI pipeline Travis CI

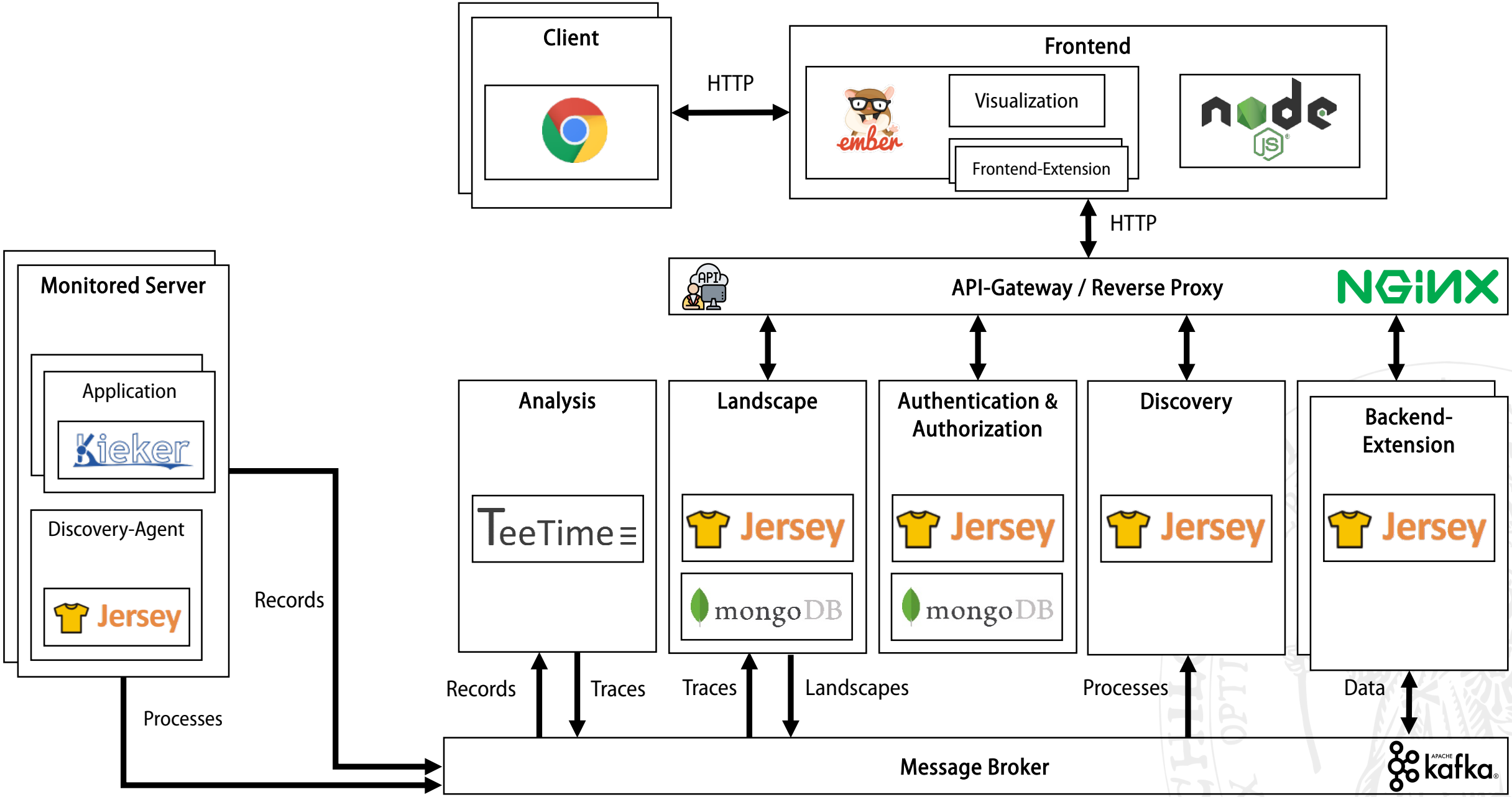
Software Configuration & Delivery

Still need to provide backend configurations for different use cases

- ⊗ First approach for an integration of extensions, but delivery was cumbersome
- ✓ We provide a jar file for each service with an embedded web server
- ✓ We offer ready-to-use Docker images for each part of our software

Frontend requires a different approach

- ⊗ Not possible to install an Ember addon inside of a deployed Ember application
- ⊗ Currently developing a build service for that ships ready-to-use, pre-built configurations
- ⊗ Alternatively these will be available as Docker images


Client
HTTP
Frontend
Visualization
Frontend-Extension
HTTP
API-Gateway / Reverse Proxy
Monitored Server
Application
Kieker
Discovery-Agent
Jersey
Records
Processes
Analysis
TeeTime
Landscape
Jersey
mongoDB
Authentication & Authorization
Jersey
mongoDB
Discovery
Jersey
Backend-Extension
Jersey
Records
Traces
Traces
Landscapes
Processes
Data
Message Broker
kafka

Conclusion and Future Work

ExplorViz

https://www.explorviz.net
https://github.com/ExplorViz

- ✓ Modularized and migrated our architecture from a Monolith to Microservices
- ✓ Significant simplification on the development of features
- ✗ Ongoing development
  - Further refactoring and modularization
  - Migrating missing features
  - Adding new features

References

[1] C. Goble, “Better Software, Better Research,” IEEE Internet Computing, vol. 18, no. 5, pp. 4–8, Sep. 2014.

[2] A. Johanson and W. Hasselbring, “Software engineering for computational science: Past, present, future,” Computing in Science & Engineering, vol. 20, no. 2, pp. 90–109, Mar. 2018.

[3] W. Hasselbring and G. Steinacker, “Microservice Architectures for Scalability, Agility and Reliability in ECommerce,” in Proceedings of the IEEE International Conference on Software Architecture Workshops (ICSAW), Apr. 2017.

[5] F. Fittkau, A. Krause, and W. Hasselbring, “Software landscape and application visualization for system comprehension with ExplorViz,” Information and Software Technology, vol. 87, pp. 259–277, Jul. 2017.

[6] F. Fittkau, S. Roth, and W. Hasselbring, “ExplorViz: Visual runtime behavior analysis of enterprise application landscapes,” in 23rd European Conference on Information Systems (ECIS 2015 Completed Research Papers), AIS Electronic Library, May 2015, pp. 1–13.

[18] C. Zirkelbach, A. Krause, and W. Hasselbring, “On the Modernization of ExplorViz towards a Microservice Architecture,” in Combined Proceedings of the Workshops of the German Software Engineering Conference 2018, vol. Online Proceedings for Scientific Conferences and Workshops, Ulm, Germany: CEUR Workshop Proceedings, Feb. 2018.

[31] A. Krause, C. Zirkelbach, and W. Hasselbring, “Simplifying Software System Monitoring through Application Discovery with ExplorViz,” in Proceedings of the Symposium on Software Performance 2018: Joint Developer and Community Meeting of Descartes/Kieker/Palladio, Nov. 2018.

Image References

- Share Icon made by Hadrien
- Correct, Cancel, Right arrow Icon made by Lucy G
- Right arrow Icon made by Lyolya
- Circular outlined Button Icon made by Catalin Fertu
- Planning Icon made by Prosymbols
- Puzzle Icon made by DinosoftLabs
- Coliseum Icon made by zlatko-najdenovski
- Network Icon made by itim2101
- Jigsaw, Purpose Icon made by geptatah
- Technical Support Icon made by srip
- Settings Icon made by Gregor Cresnar
- OSS Icon made by Pixel perfect
- Blueprint, Puzzle, Evolution Icon made by Freepik
- Problem, Merging Arrow, Medical shape, Data, Binary,
  Application, API, Demand, HR, Projector, Product Icon made by Eucalyp

All icons are from www.flaticon.com