University of Stuttgart
Germany

envite

Energieeffizienz verschiedener Backend-Architekturstile: Monolith vs. Microservices

EcoCompute Conference 2024

25.04.2024

David Kopp

Based on my master‘s thesis, conducted in cooperation with envite consulting GmbH

Overview

Approach

- Comparison of cloud-based architecture styles
- **Focus:** Modular Monolith vs. Microservices
- **Metric:** Energy Efficiency

Reference Application

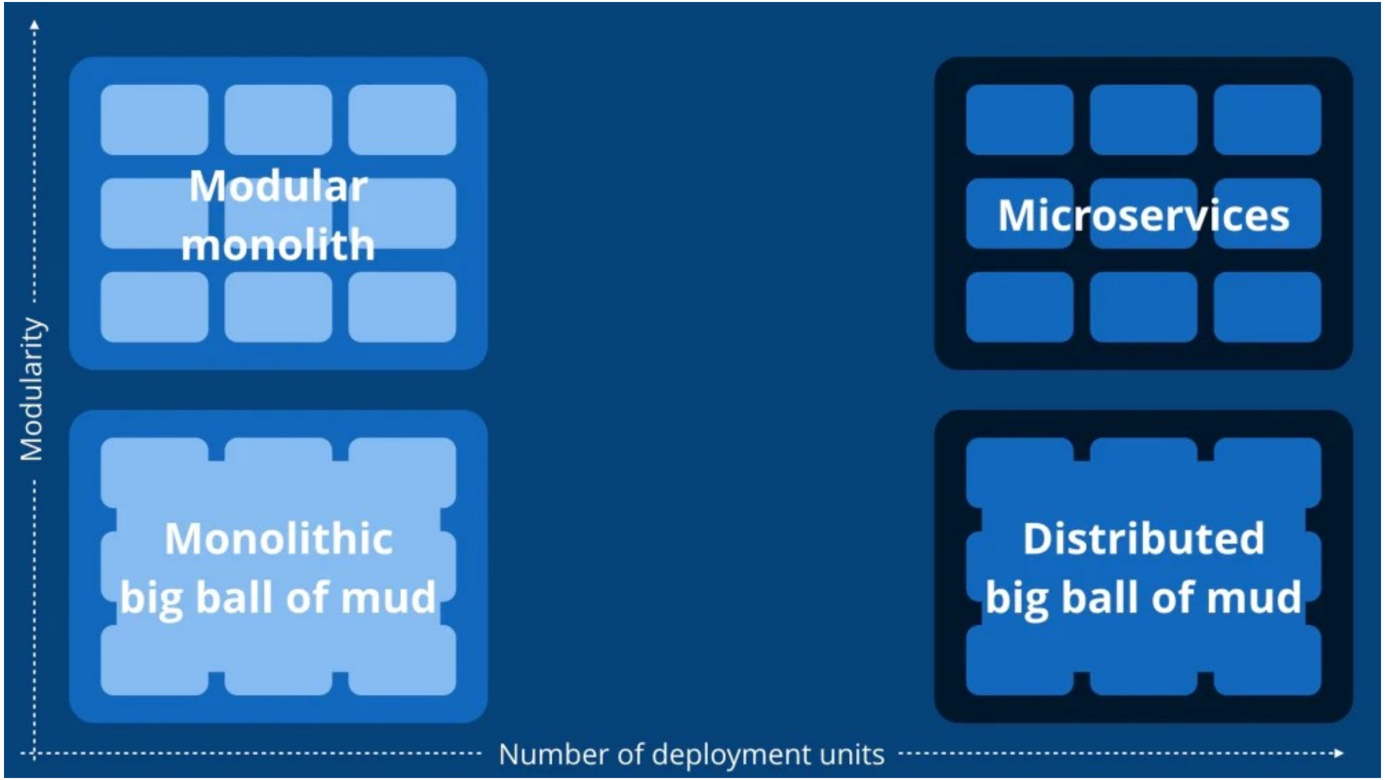
- **Microservices:** T2-Project (implementing saga pattern)
- **Modular Monolith:** Own implementation

Measurement & Analysis

- **Measurement Environments:**
  1. Green Metrics Tool
  2. Kubernetes & Kepler

Modular Monolith vs. Microservices

Why Comparing These Two Architecture Styles?



[Bro18]

Related Work

Monolith vs. Microservices in Terms of Performance

Monolith

- Better throughput under **low load**
- Lower response times
  (no overhead through network communication)

Microservices

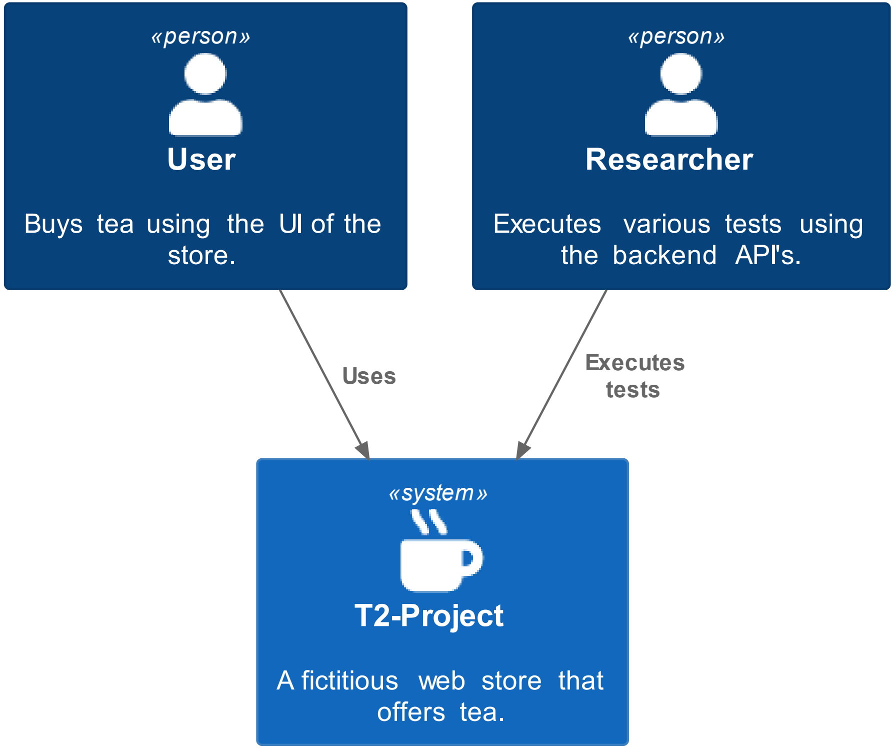
- Better throughput under **high load**
  (especially in compute-intensive scenarios)
- Better scalability
  (due to smaller components)

Papers are listed on slide References for Related Work

Reference Application

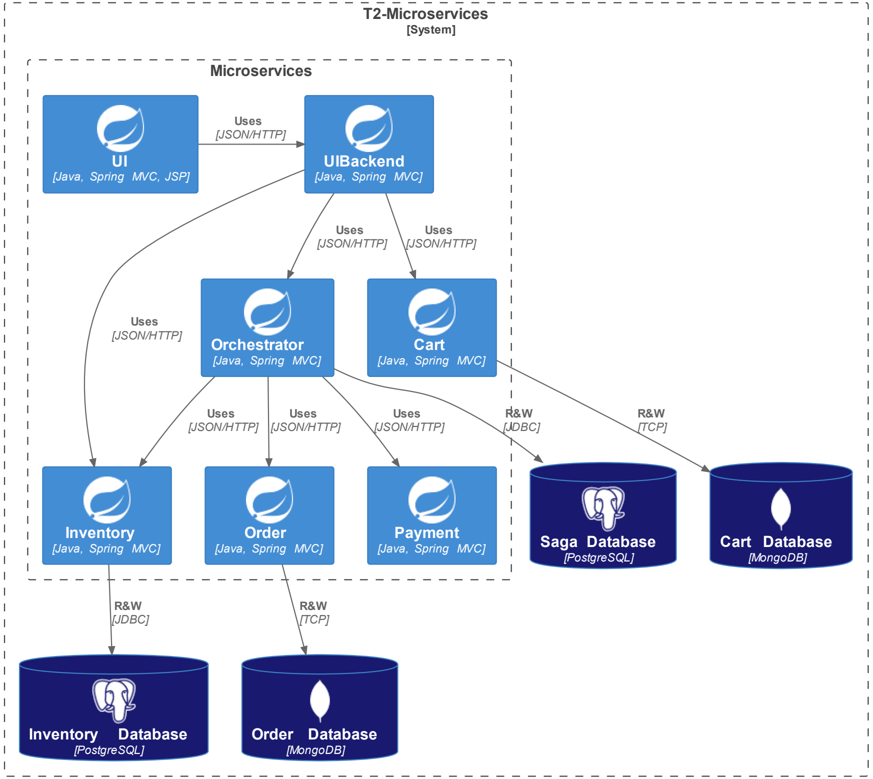
Reference Application – T2-Project

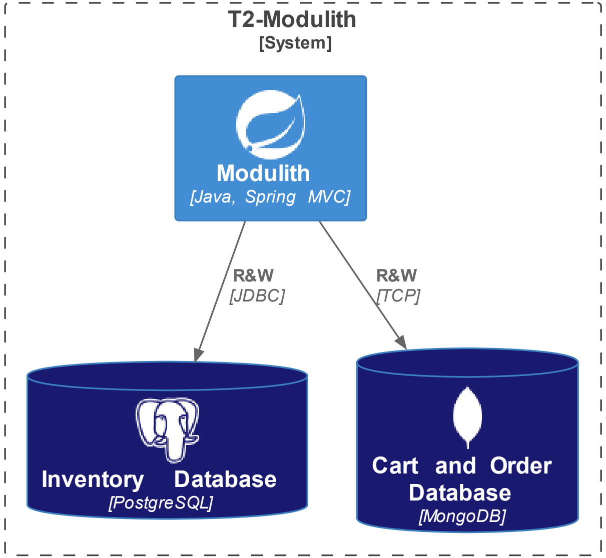
Business & System Context



Reference Application – T2-Project

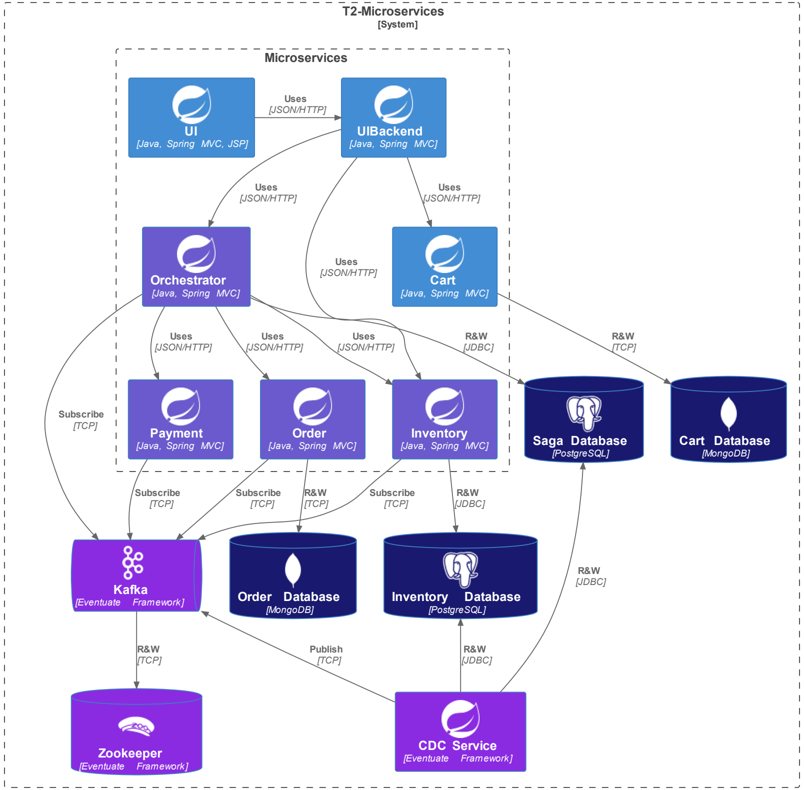
Variants: T2-Microservices & T2-Modulith

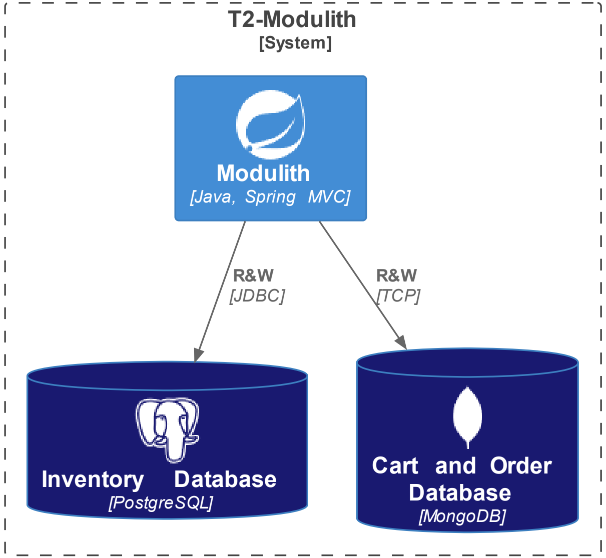




Reference Application – T2-Project

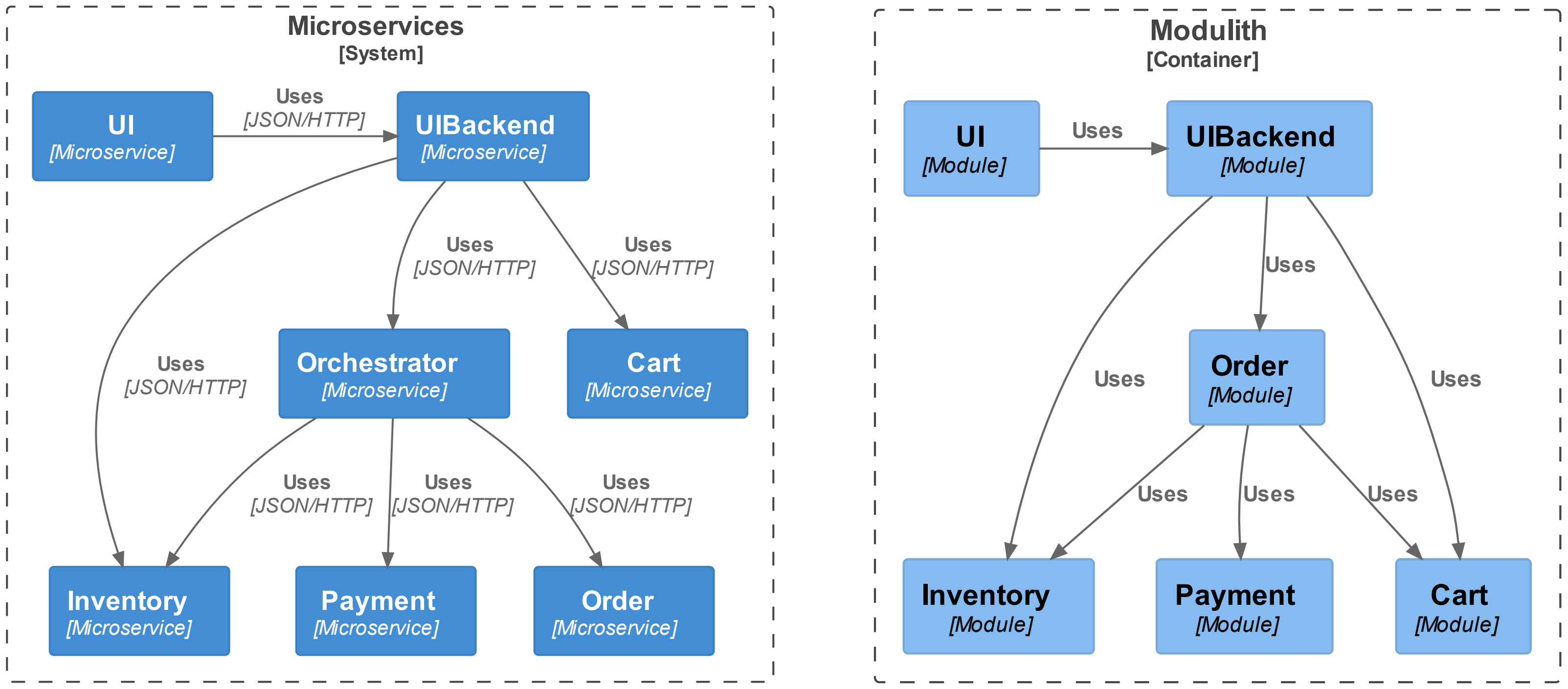
Implementation of the Saga Pattern & Messaging Middleware





Reference Application – T2-Project

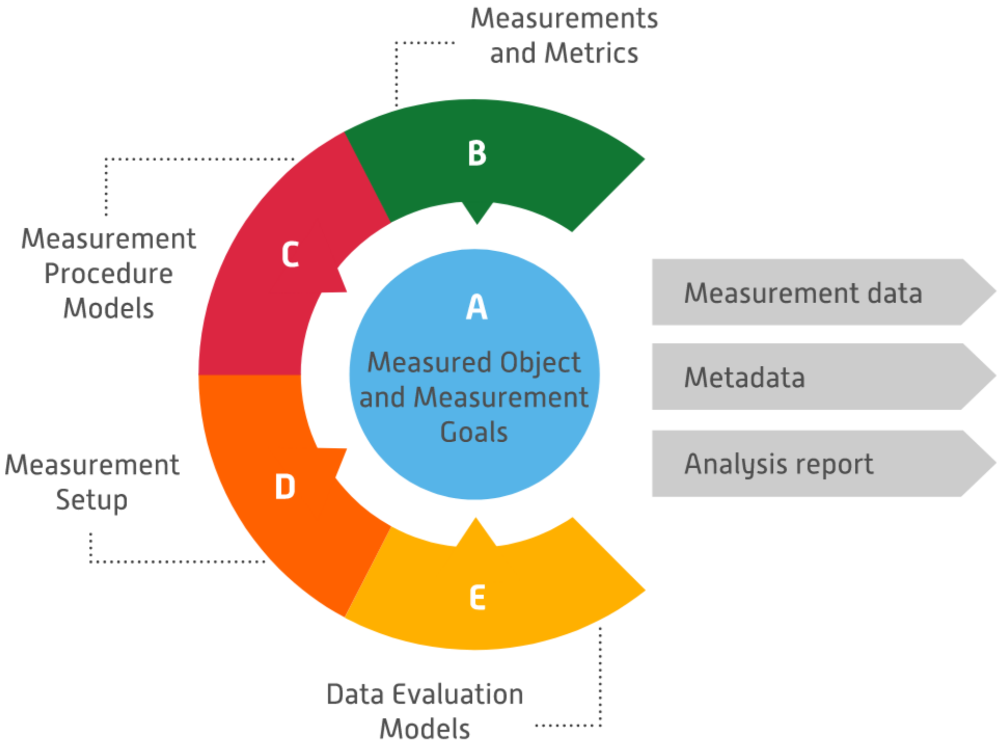
Modularity



Measurement Methodology

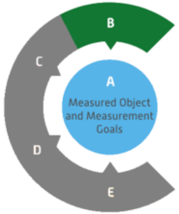
Measurement Methodology

Green Software Measurement Model (GSMM) by Guldner et al. (2024)



[GBC+24]

Measurement Model – Measurements & Metrics



- **Metric ("useful work"):** Energy consumption of one order
- **Measurement:** Processing of three sequential HTTP requests

GET – product list

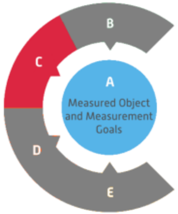
↓

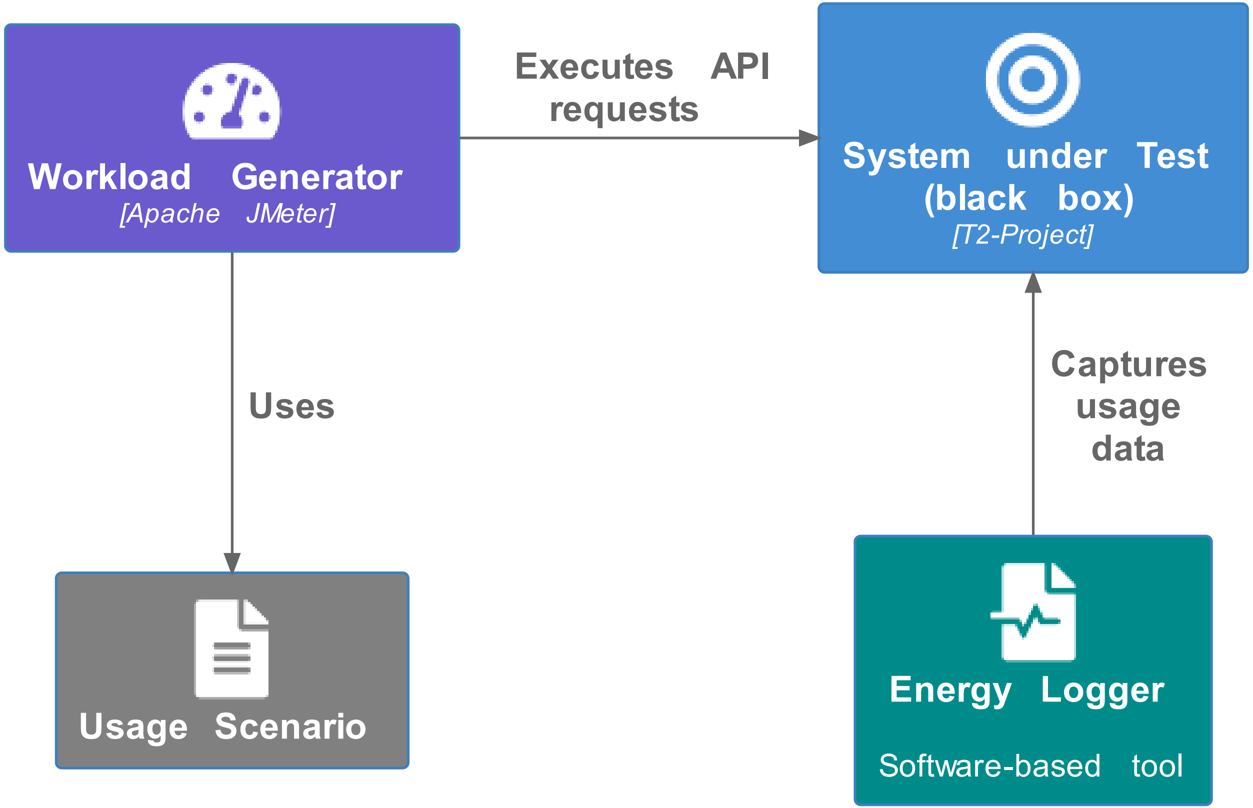
POST – add item to cart

↓

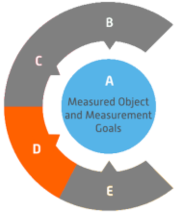
POST – confirm order

Measurement Model – Procedure





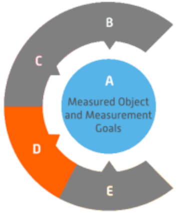
Measurement Model – Setups

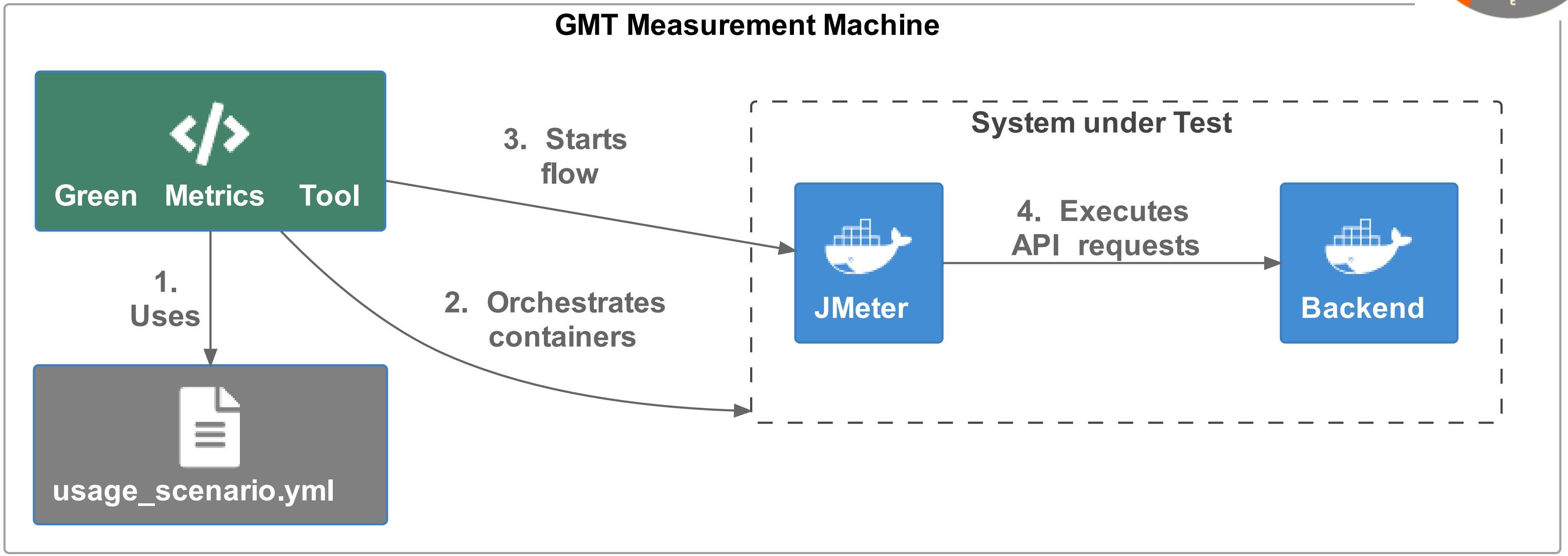


Green Metrics Tool

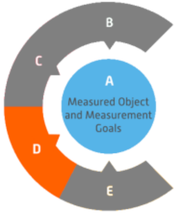
Kubernetes & Kepler

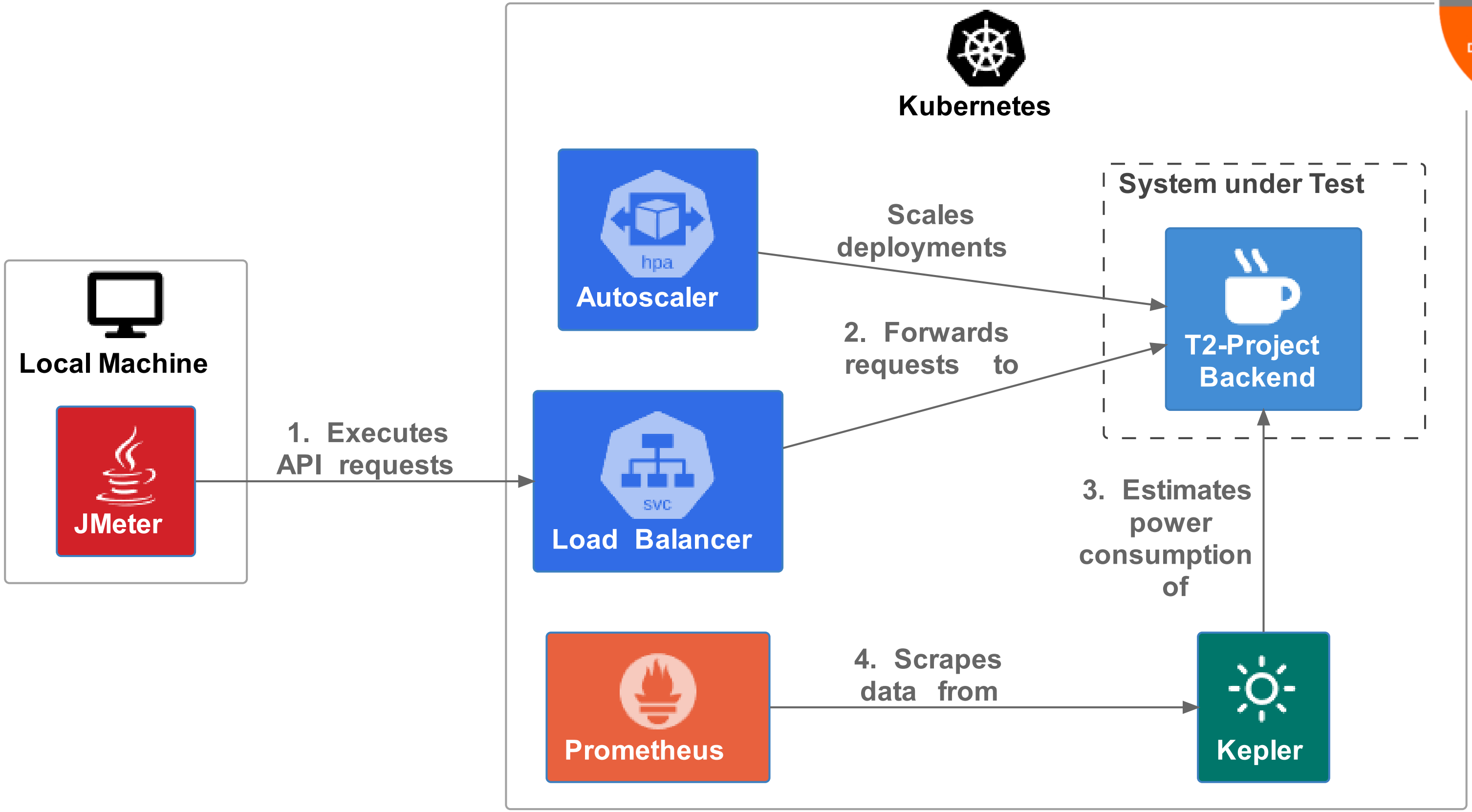
Measurement Setup – Green Metrics Tool (simplified)





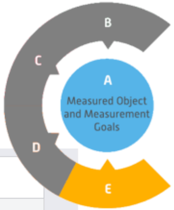
Measurement Setup – K8s & Kepler (simplified)

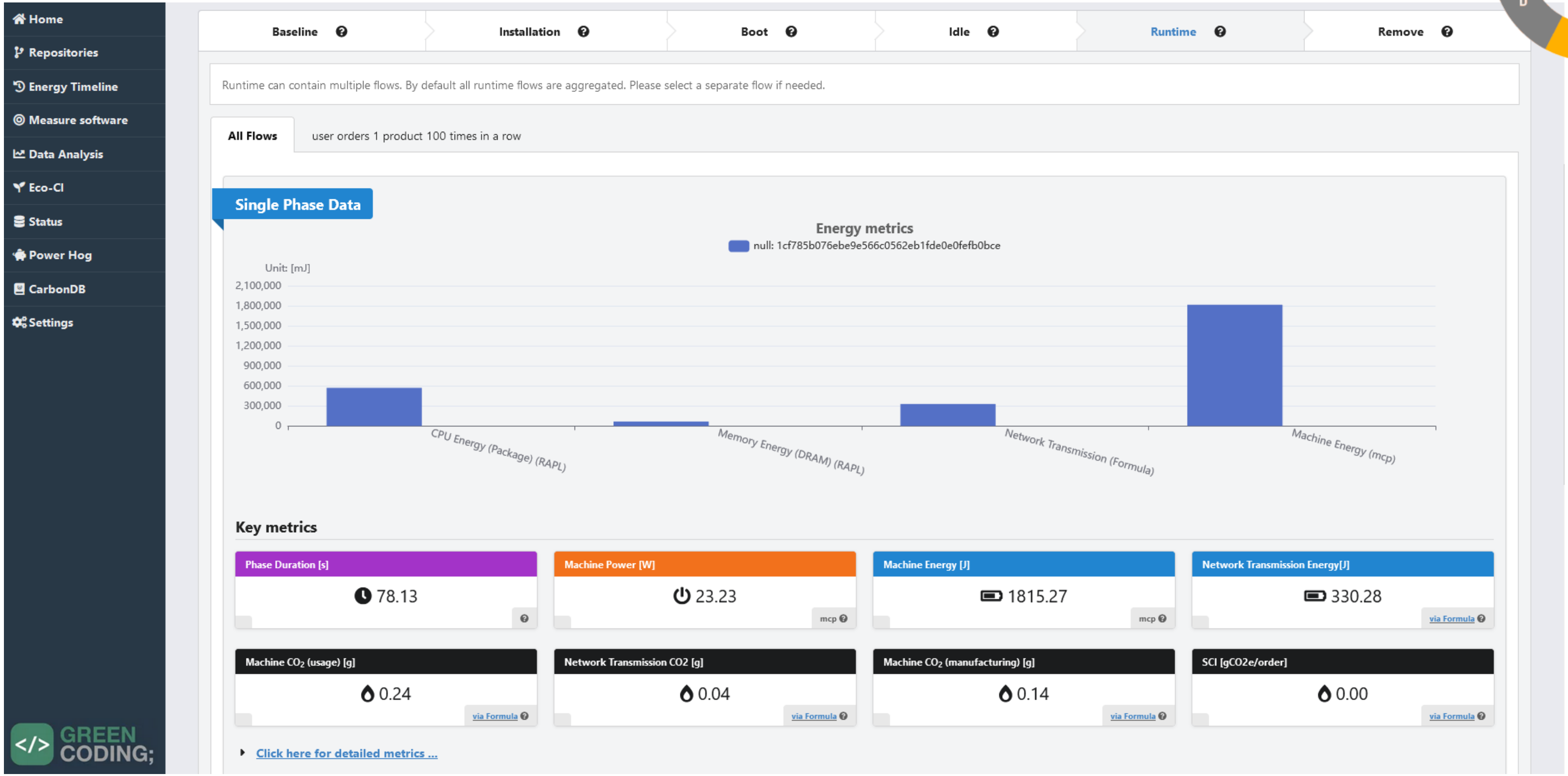




Measurement Model – Data Evaluation (Green Metrics Tool)

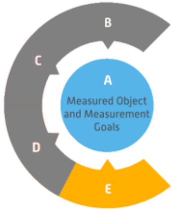
GMT Dashboard: Single Run View (Key Metrics)

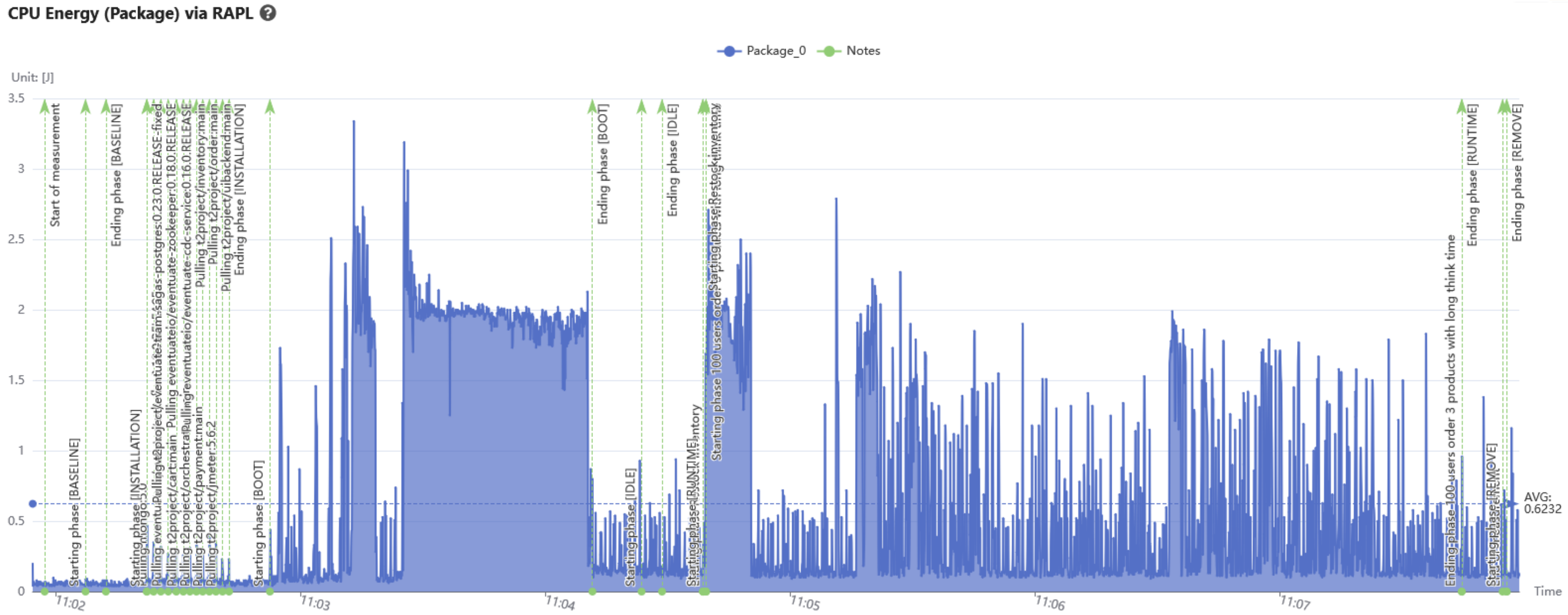




Measurement Model – Data Evaluation (Green Metrics Tool)

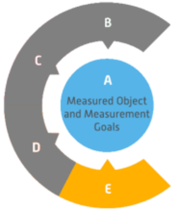
GMT Dashboard: CPU Energy Consumption

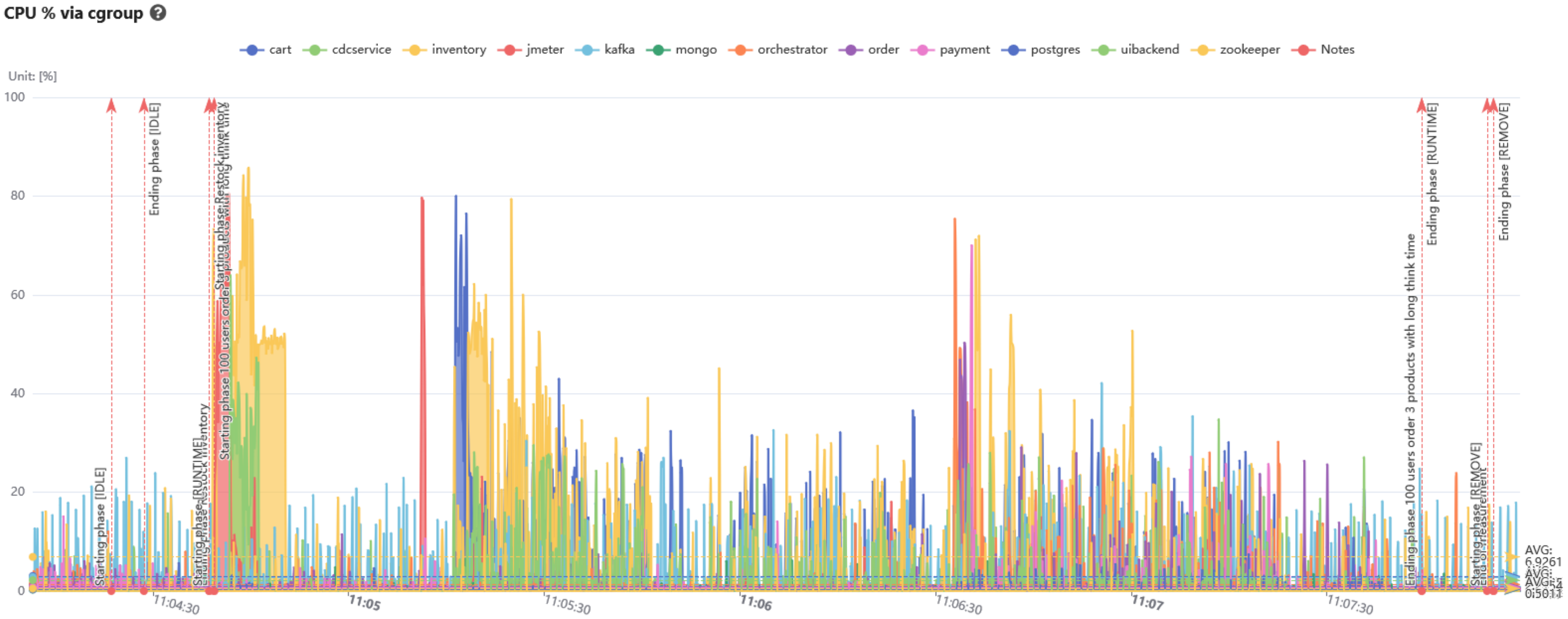




Measurement Model – Data Evaluation (Green Metrics Tool)

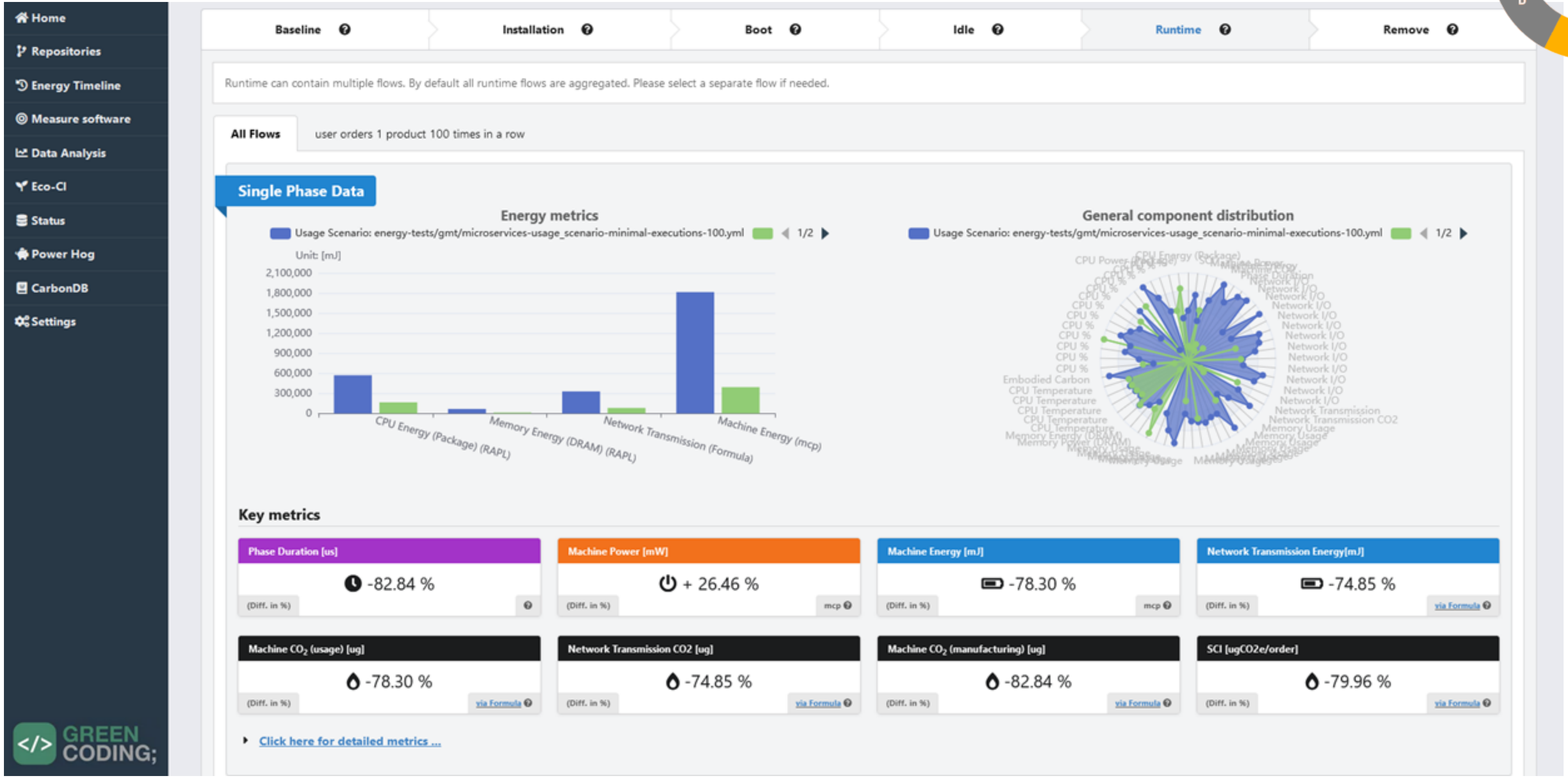
GMT Dashboard: CPU Utilization





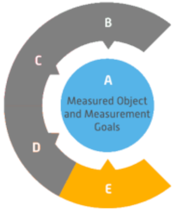
Measurement Model – Data Evaluation (Green Metrics Tool)

GMT Dashboard: Comparison View



Measurement Model – Data Evaluation (Kepler)

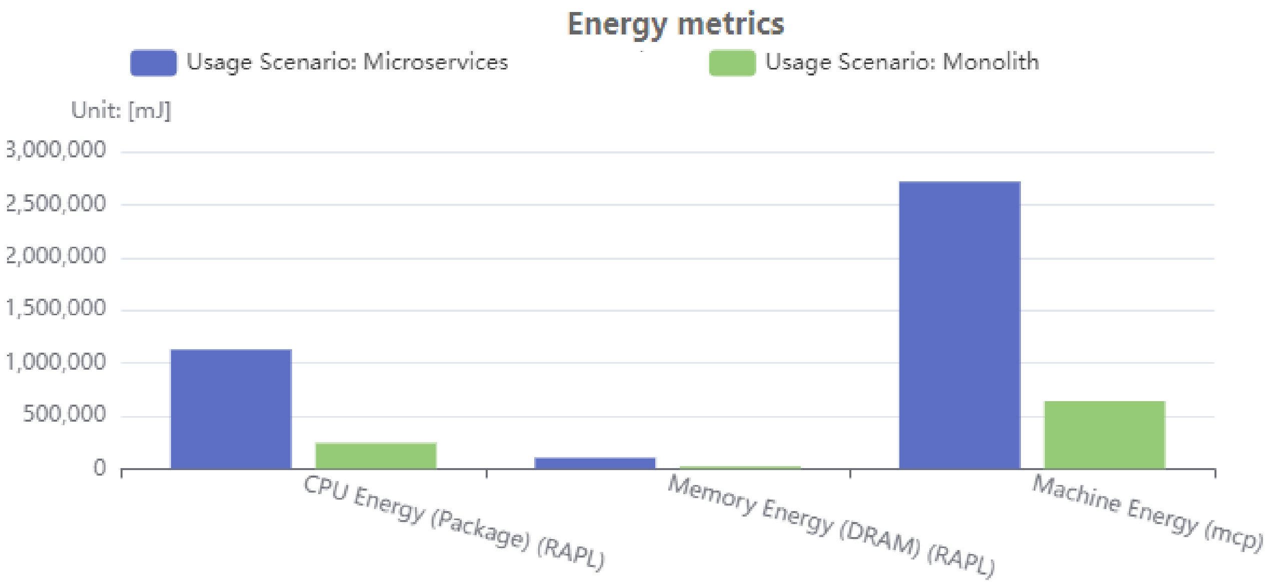
Grafana Dashboard

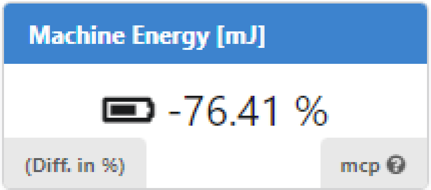


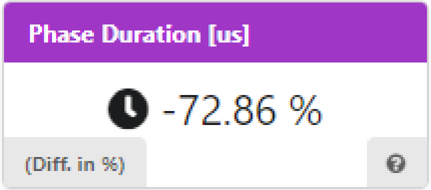
Results & Discussion

Results – Green Metrics Tool

Energy Consumption of Boot Phase

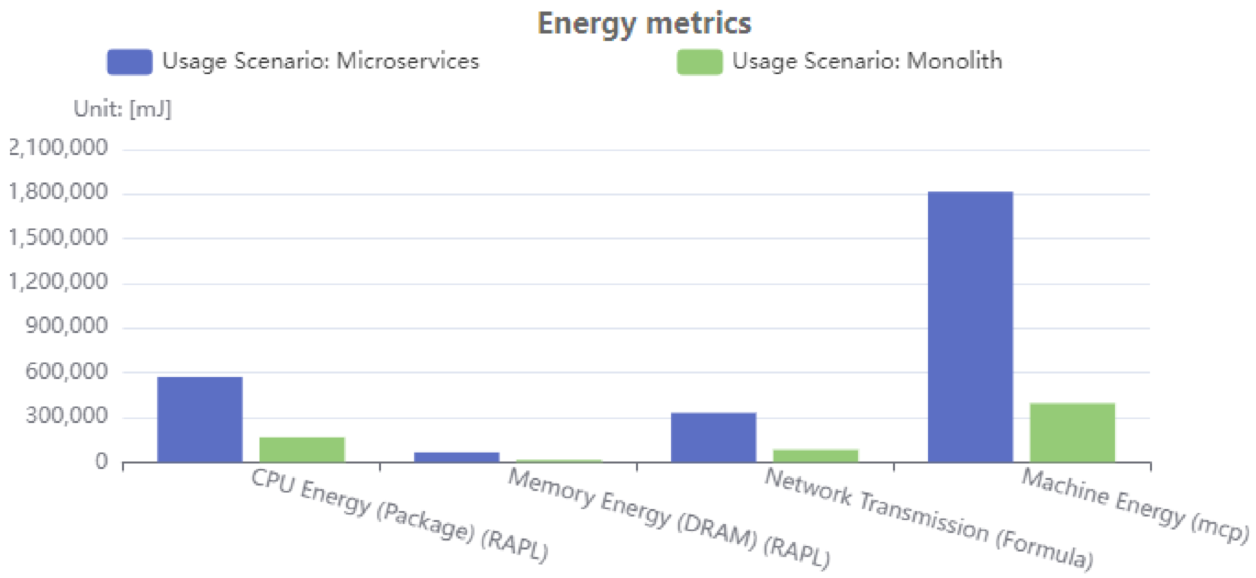


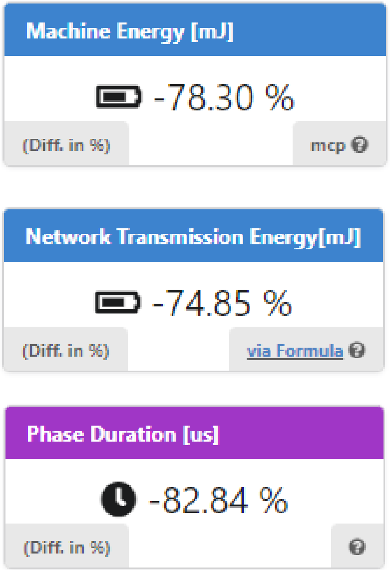




Results – Green Metrics Tool

Scenario: 100 Sequential Executions

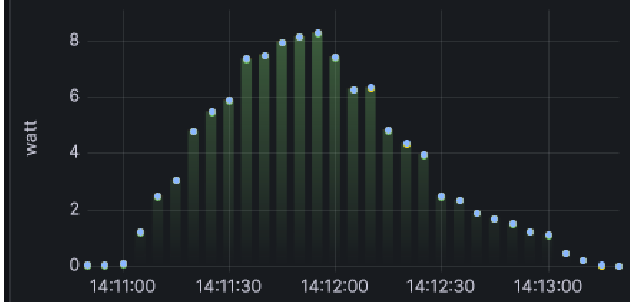




Results – K8s & Kepler

Scenario: 1 user, 100 executions (no scaling)

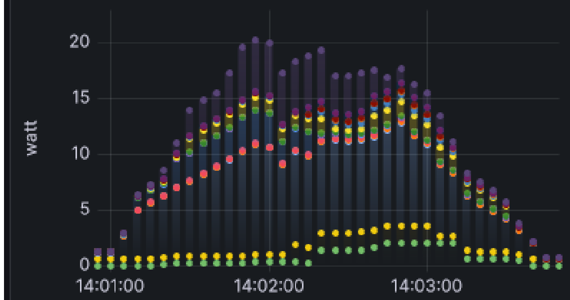
"t2-modulith" – Pod Power Consumption (W)



Name	Mean	Max	Total
backend-5798b44c5c-d2pqp	3.47	8.24	108
mongo-mongodb-6d878fcbdc-pjcdj	0.0216	0.0436	0.671
postgres-0	0.00581	0.0109	0.180

Energy Consumption (Ws / J)

"t2-microservices" – Pod Power Consumption (W)



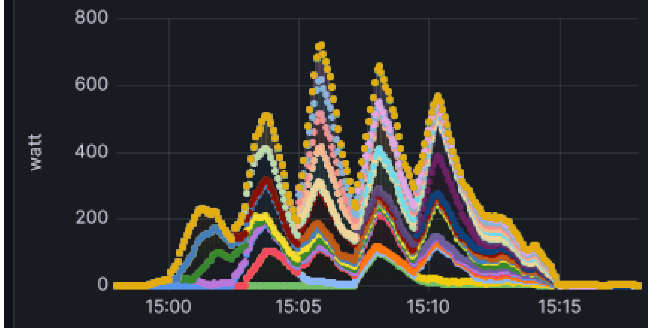
Name	Mean	Max	Total
inventory-744d446954-8cm42	5.96	9.93	215
uibackend-6896c76b89-s68bt	1.81	4.66	65.2
kafka-2	0.935	2.99	33.7
cdcservice-7dc8cd8df7-gss9d	0.899	1.57	32.4
orchestrator-65f9f5c5f7-vm886	0.761	1.39	27.4
cart-5f945cf68-j9gv2	0.737	2.08	26.5
order-6df8b6654f-dgfqs	0.280	0.656	10.1
postgres-orchestrator-0	0.234	0.624	8.43

Energy Consumption (Ws / J)

Results – K8s & Kepler

Scenario: 50 users, 10 min ramp-up, 5 min hold

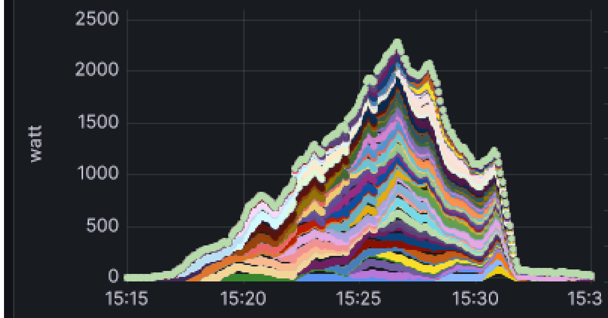
"t2-modulith" – Pod Power Consumption (W)



Name	Mean ⌄	Max	Total
backend-76674b8bb7-r8tnq	23.0	108	2345
backend-76674b8bb7-rc7n2	22.6	105	2240
backend-76674b8bb7-xjzv4	19.9	109	2588
backend-76674b8bb7-6hwpg	19.8	99.6	2040
backend-76674b8bb7-s5csp	19.7	105	2560
backend-76674b8bb7-ckchw	19.6	99.3	1936
backend-76674b8bb7-l8ct8	18.5	107	2911
backend-76674b8bb7-v4fd8	18.0	107	2827

Energy Consumption (Ws / J)

"t2-microservices" – Pod Power Consumption (W)



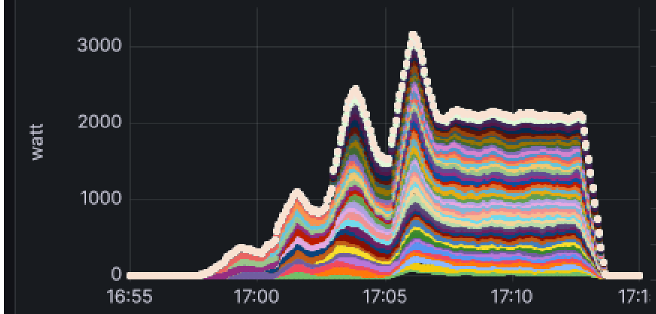
Name	Mean ⌄	Max	Total
inventory-7d9b5b5478-f7xcr	46.1	107	7558
inventory-7d9b5b5478-scntr	36.8	107	4126
inventory-7d9b5b5478-fr62r	35.8	106	3470
inventory-7d9b5b5478-v5dhb	33.9	109	6128
inventory-7d9b5b5478-l4jpz	33.5	98.9	3254
inventory-7d9b5b5478-98sfw	32.4	107	3631
inventory-7d9b5b5478-4w996	32.4	106	4116
inventory-7d9b5b5478-vwsdr	32.4	105	3142

Energy Consumption (Ws / J)

Results – K8s & Kepler

Scenario: Compute-intensive (~1 sec per order), 50 users, 10 min ramp-up, 5 min hold

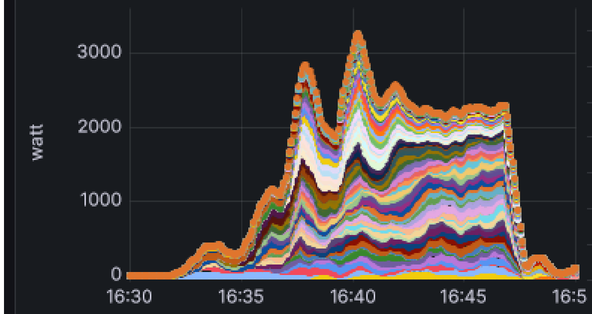
"t2-modulith" – Pod Power Consumption (W)



Name	Mean	Max	Total
backend-5c76ddbcf8-nlf2k	47.9	92.3	9386
backend-5c76ddbcf8-n9bjx	45.6	104	5380
backend-5c76ddbcf8-vkl5l	45.3	105	5207
backend-5c76ddbcf8-lr84d	45.3	105	5340
backend-5c76ddbcf8-gcbzx	45.1	103	5324
backend-5c76ddbcf8-6fm5x	44.7	101	6885
backend-5c76ddbcf8-6fz9t	44.4	104	7640
backend-5c76ddbcf8-wlzz9	44.4	98.2	6348

Energy Consumption (Ws / J)

"t2-microservices" – Pod Power Consumption (W)



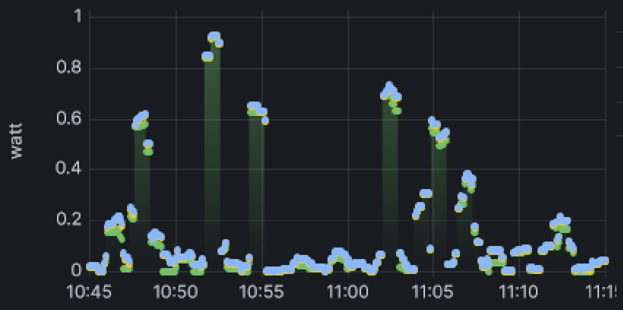
Name	Mean	Max	Total
computation-simulator-754444d565-g5wrg	53.1	111	8132
computation-simulator-754444d565-9d7bw	48.9	109	7631
computation-simulator-754444d565-4l4b4	45.9	110	6608
computation-simulator-754444d565-6v67f	45.7	111	6587
computation-simulator-754444d565-56pbm	45.5	112	10965
computation-simulator-754444d565-k2kb4	43.9	110	7639
computation-simulator-754444d565-crgcv	43.0	103	7488
computation-simulator-754444d565-tvjwv	43.0	101	7524

Energy Consumption (Ws / J)

Results – K8s & Kepler

Energy Consumption in Idle (30 min)

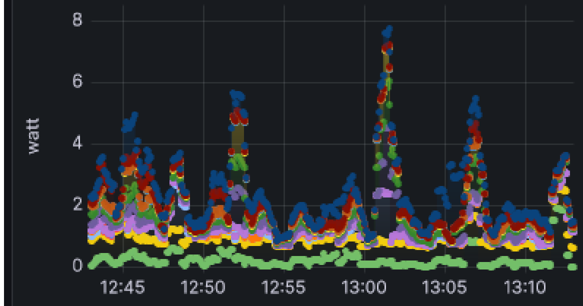
"t2-modulith" – Pod Power Consumption (W)



Name	Mean	Max	Total
backend-55dfcc4469-nvfvf	0.150	0.921	54.0
mongo-mongodb-6d878fcbdc-hzx27	0.0143	0.0665	5.17
postgres-0	0.00533	0.0109	1.93

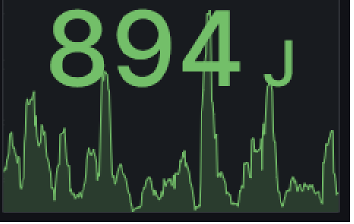
Energy Consumption (Ws / J)

"t2-microservices" – Pod Power Consumption (W)



Name	Mean	Max	Total
cdcservice-667db4d459-br6bj	0.677	1.11	245
uibackend-6f5fff8787-4xmk7	0.341	2.47	123
inventory-6b86d657d-bnrbw	0.261	1.67	94.3
cart-67f676cc8-kdf29	0.261	1.90	94.2
order-59644875d7-jbxl8	0.248	1.52	89.4
payment-68cd944745-dwwdw	0.240	1.49	86.5
orchestrator-79869c9d55-z8wfr	0.226	2.12	81.5
ui-cc87d879b-bkb4l	0.136	0.818	49.0

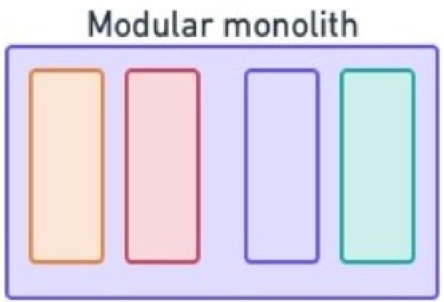
Energy Consumption (Ws / J)



Limitations

- One small reference application (with saga) is not representative
- Autoscaling and measurement of scaling behavior could be improved
- Costs of infrastructure components are not taken into account
- Optimizing for energy efficiency is not everything

Discussion
Energy Efficiency – Modular Monolith vs. Microservices



No overhead through network communication

Only one runtime environment

Local transactions

Fain-grained scalability

Potential for better utilization of resources due to smaller size

Hibernate of individual components (not tested)

Symbols by [Fer22]

Learnings

GMT: Easy usage & utilizes best practices for measuring usage scenarios

Kepler: Good for monitoring workloads over longer time in K8s environments

Different philosophies of measurement tools:
Per-process measurement? Usage scenarios? Metrics provider? Virtualized environments?

Energy consumption of network transfer? Biggest issue is data processing

Saga is quite slow and inefficient (but ensures data integrity and scalability)

Conclusions

- Modular Monolith is easier to handle than a microservices system
- Monoliths are more energy efficient in scenarios with low or constant load
- Microservices have advantages in compute-intensive scenarios with high load
- There is no general recommendation, it depends on your scalability needs

University of Stuttgart
Germany

Thank you!

David Kopp

e-mail st160056@stud.uni-stuttgart.de

University of Stuttgart

Institute of Software Engineering
Software Quality and Architecture Group

Energieeffizienz verschiedener Backend-Architekturstile: Monolith vs. Microservices

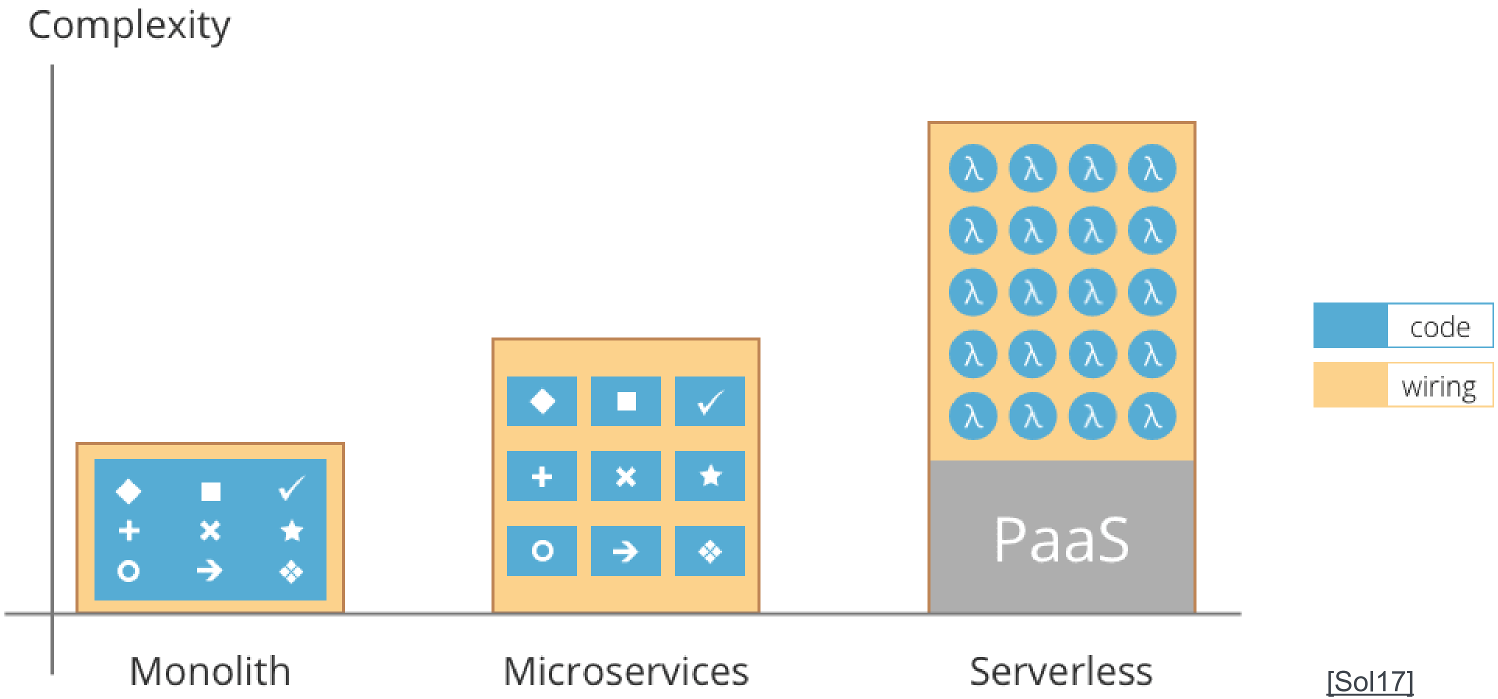
EcoCompute Conference 2024

25.04.2024

Additional Slides

Foundations

Architecture & deployment styles in the cloud



[Sol17]

Foundations
Spring Modulith

- Set of libraries that support developers implementing logical modules in Spring Boot applications
- Features:
  - Apply structural validation
  - Document the module arrangement
  - Run integration tests for individual modules
  - Observe the modules interaction at runtime
  - Implement module interaction in a loosely-coupled way

Foundations

Performance of Saga Frameworks

Table 2: Performance test, time in mm:ss – scenario 1 (1 000 requests, 10 threads).

Project	**Processing delay**	**Total time**	**Completed requests**
Axon service	00:46	00:51	1 000
Eventuate service	00:49	00:58	2 000
Eventuate Tram service	00:27	00:34	1 000
LRA service	00:04	01:10	1 000

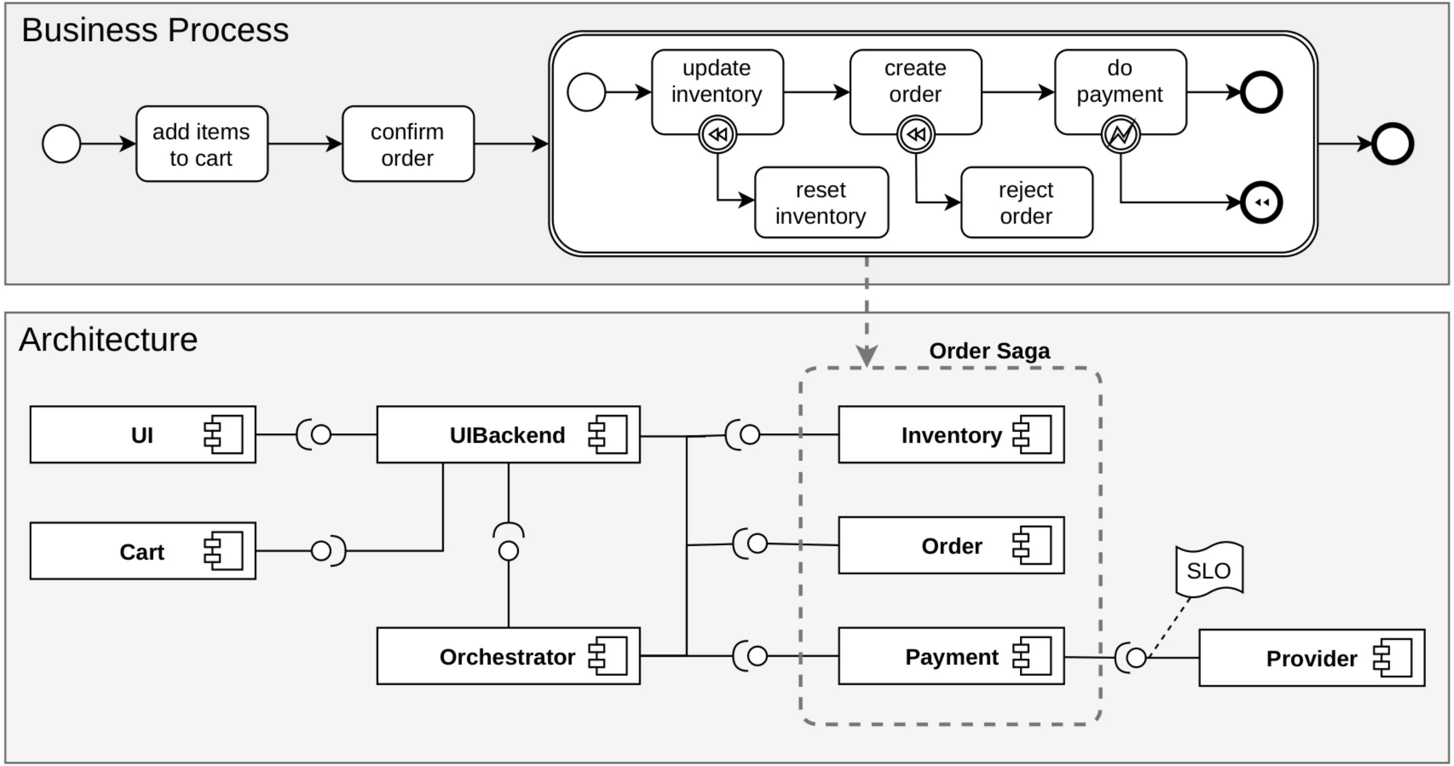
Table 3: Performance test, time in mm:ss – scenario 2 (10 000 requests, 100 threads).

Project	**Processing delay**	**Total time**	**Completed requests**
Axon service	06:53	07:44	5 657
Eventuate service	14:05	14:46	19 791
Eventuate Tram service	03:20	03:56	10 000
LRA service	00:22	08:58	10 000

Source: Štefanko, M., Chaloupka, O., & Rossi, B. (2019). The saga pattern in a reactive microservices environment. https://doi.org/10.5220/0007918704830490

Reference Application – T2-Project

Business Process – Implemented with the Saga Pattern



[Sti21]

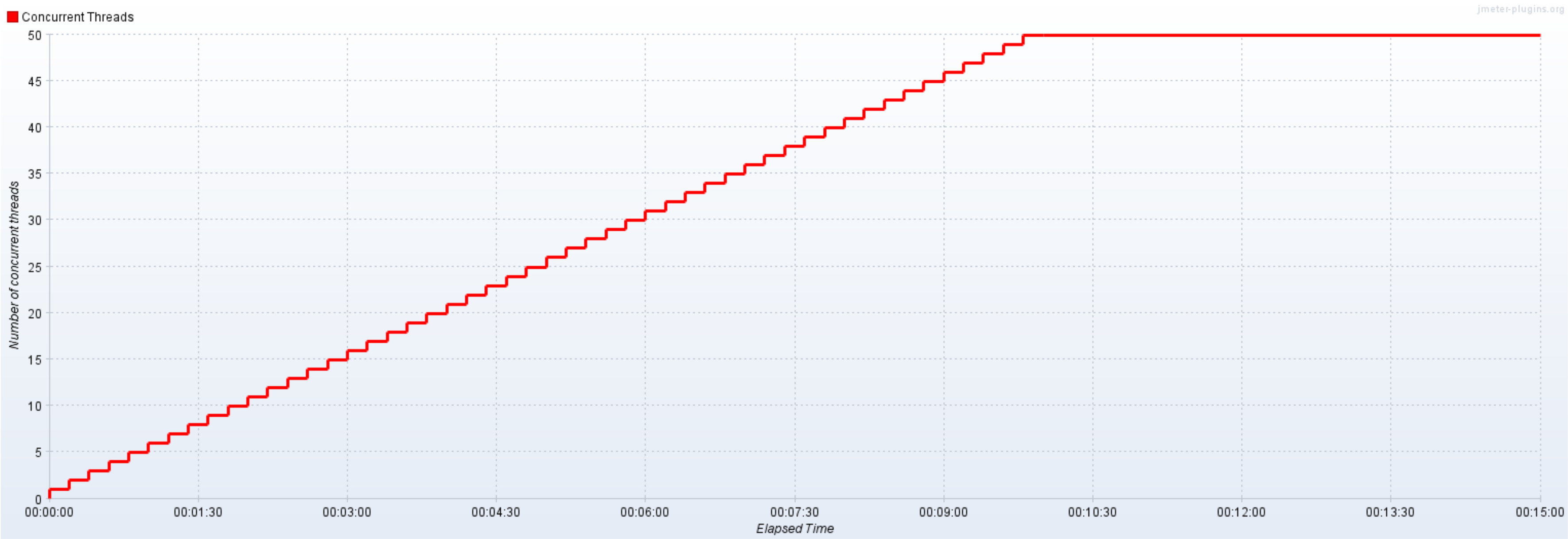
Measurement Setup – Green Metrics Tool

usage_scenario.yml

```yaml
name: T2-Modulith Basic Usage Scenario
author: David Kopp
description: One user gets the inventory, adds one product to cart and confirms the order.

compose-file: !include monolith-compose.yml

sci:
  R_d: order

flow:
 - name: single user orders one product
   container: jmeter
   commands:
   - type: console
     command: jmeter -Jhostname=backend -Jport=8080 -JnumExecutions=1 -JnumUser=1 -JnumProducts=1
              -JthinkTimeMin=0 -JloggingEnabled=true -n -t /tmp/repo/t2-project-flexible.jmx
     log-stdout: true
     read-notes-stdout: true
     read-sci-stdout: true
```

Measurement Setup – K8s & Kepler

Workload Generation for Scaling: 50 users, 10 min ramp-up, 5 min hold

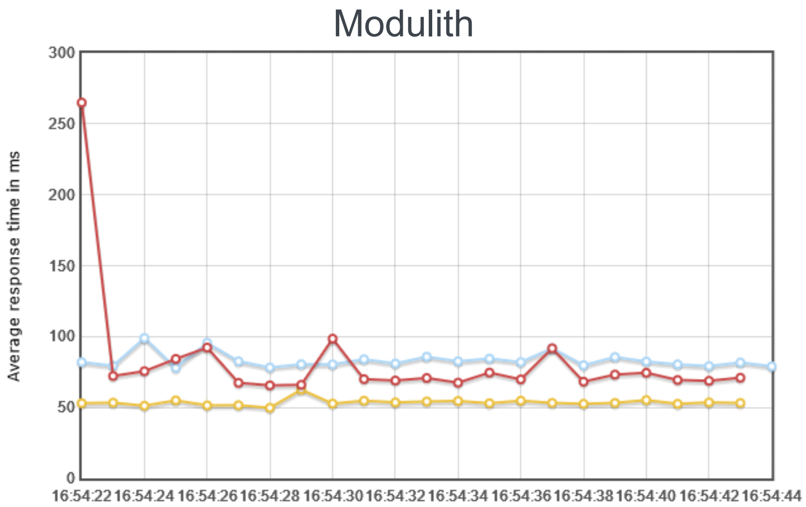


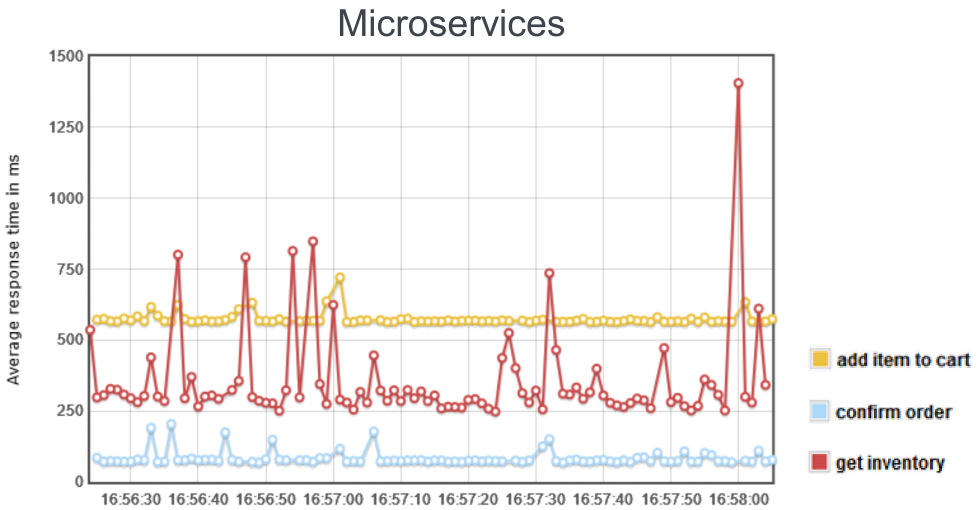
Results – Green Metrics Tool

	Number of Executions	Think Time [s]	Duration [s]	Machine Energy [J]	CPU Energy [J]	Memory Energy [J]	Network Energy [J]	SCI [mgCO2e / order]
Modulith	0	0	3.8	113.3	53.2	3.0	0.0	N/A
Microservices	0	0	4.0	125.6	55.9	3.4	0.5	N/A
Modulith	1	0	5.8	181.5	85.8	5.4	1.0	34.2
Microservices	1	0	16.6	441.3	164.3	14.0	4.7	87.3
Modulith	100	0	13.4	393.9	166.5	13.5	83.1	0.8
Microservices	100	0	78.1	1815.3	572.1	67.2	330.3	3.8
Modulith	100	1	113.5	1809.8	254.3	58.1	84.9	4.4
Microservices	100	1	175.2	3457.7	808.7	115.5	345.6	7.6

Results – K8s & JMeter

Scenario: 1 user, 100 executions (equal total CPU resources)



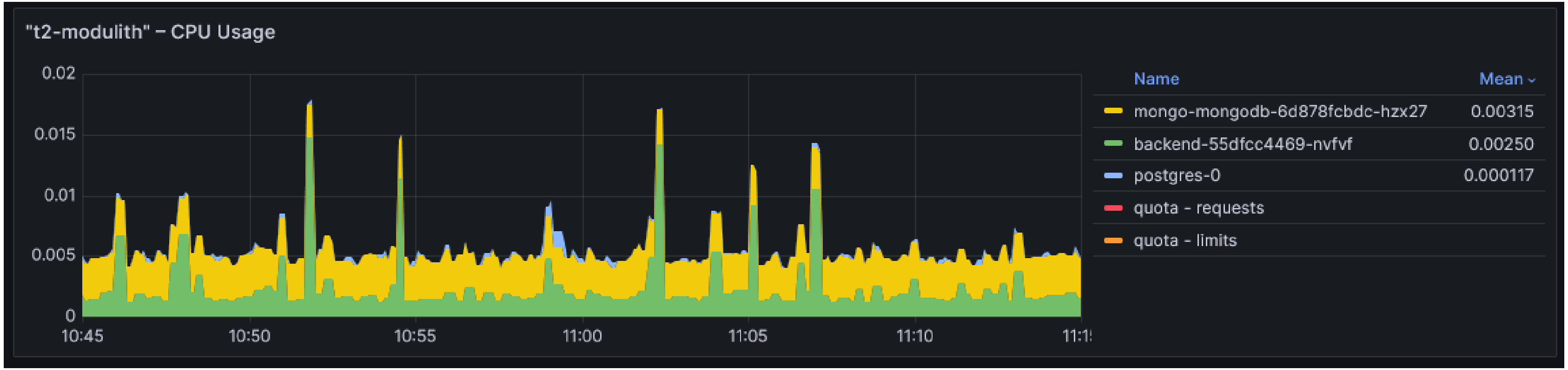


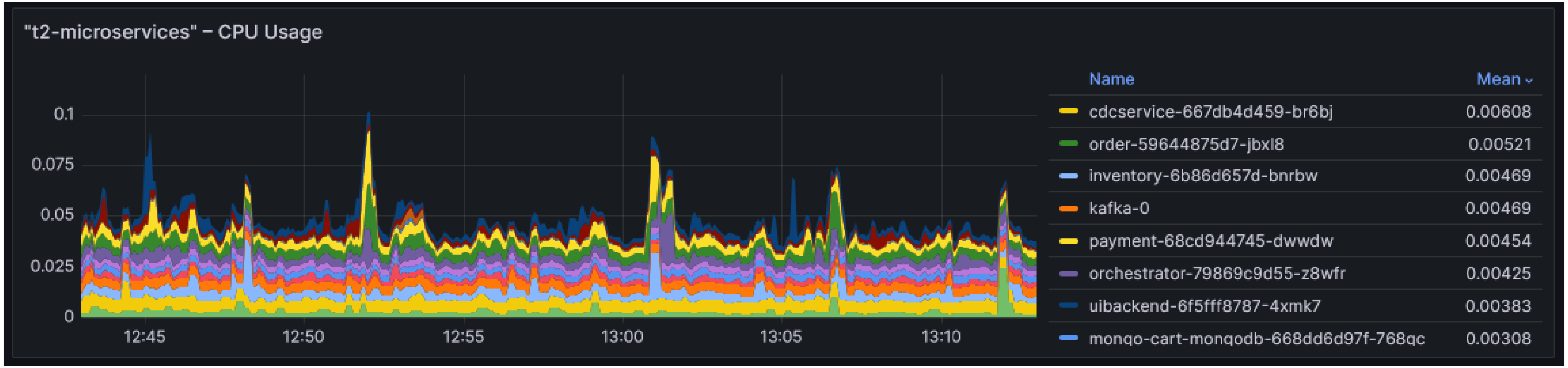
Kubernetes Resource Requests & Limits:

- Modulith: 1.5 Cores
- Microservices: 0.25 Cores

Results – K8s & Kepler

CPU Usage in Idle (30 min) (one replica each)





References for Related Work

- Ueda, T., Nakaike, T., & Ohara, M. (2016). Workload characterization for microservices. *2016 IEEE International Symposium on Workload Characterization (IISWC)*, 1–10. https://doi.org/10.1109/IISWC.2016.7581269
- Villamizar, M., Garcés, O., Ochoa, L., Castro, H., Salamanca, L., Verano, M., Casallas, R., Gil, S., Valencia, C., Zambrano, A., & Lang, M. (2017). Cost comparison of running web applications in the cloud using monolithic, microservice, and AWS Lambda architectures. *Service Oriented Computing and Applications*, *11*(2), 233–247. https://doi.org/10.1007/s11761-017-0208-y
- Al-Debagy, O., & Martinek, P. (2018). A Comparative Review of Microservices and Monolithic Architectures. *2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI)*, 000149–000154. https://doi.org/10.1109/CINTI.2018.8928192
- Bjørndal, N., Bucchiarone, A., Mazzara, M., Dragoni, N., & Dustdar, S. (2020). *Migration from monolith to microservices: Benchmarking a case study*. https://doi.org/10.13140/RG.2.2.27715.14883
- Blinowski, G., Ojdowska, A., & Przybylek, A. (2022). Monolithic vs. Microservice Architecture: A Performance and Scalability Evaluation. *IEEE Access*, *10*, 20357–20374. https://doi.org/10.1109/ACCESS.2022.3152803
- Okrój, S., & Jatkiewicz, P. (2023). Differences in performance, scalability, and cost of using microservice and monolithic architecture. *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, 1038–1041. https://doi.org/10.1145/3555776.3578725
- Faustino, D., Gonçalves, N., Portela, M., & Rito Silva, A. (2024). Stepwise migration of a monolith to a microservice architecture: Performance and migration effort evaluation. *Performance Evaluation*, *164*, 102411. https://doi.org/10.1016/j.peva.2024.102411

Summary of results: Related Work

References

- [BHR19] Barroso, L. A., Hölzle, U., & Ranganathan, P. (2019). *The Datacenter as a Computer: Designing Warehouse-Scale Machines*. Springer International Publishing. https://doi.org/10.1007/978-3-031-01761-2
- [Bro18] S. Brown. “Modular Monoliths”. Conference Talk. GOTO 2018 (Berlin). Nov. 22, 2018. url: https://youtu.be/5OjqD-ow8GE
- [FBW+21] Freitag, C., Berners-Lee, M., Widdicks, K., Knowles, B., Blair, G. S., & Friday, A. (2021). The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations. Patterns, 2(9), 100340. https://doi.org/10.1016/j.patter.2021.100340
- [Fer22] Tomas Fernandez (2022). When Microservices Are a Bad Idea. https://semaphoreci.com/blog/bad-microservices
- [GBC+24] A. Guldner, R. Bender, C. Calero, G. S. Fernando, M. Funke, J. Gröger, L. M. Hilty, J. Hörnschemeyer, G.-D. Hoffmann, D. Junger, T. Kennes, S. Kreten, P. Lago, F. Mai, I. Malavolta, J. Murach, K. Obergöker, B. Schmidt, A. Tarara, J. P. De Veaugh-Geiss, S. Weber, M. Westing, V. Wohlgemuth, S. Naumann. “Development and Evaluation of a Reference Measurement Model for Assessing the Resource and Energy Efficiency of Software Products and Components—Green Software Measurement Model (GSMM)”. In: Future Generation Computer Systems 155 (June 1, 2024), pp. 402–418. doi: 10.1016/j.future.2024.01.033.
- [GSF23] Green Software Principles: Green Software Foundation (2023). https://learn.greensoftware.foundation
- [Haw22] Visualization “Warming Stripes”: Ed Hawkins (2022). CC BY 2.0. https://showyourstripes.info/s/europe/germany/badenwuerttemberg
- [Loc24] DPA/The Local (2024-04-02). Germany logs warmest March on record. https://www.thelocal.de/20240402/germany-logs-warmest-march-on-record
- [Sol17] Visualization “Monolith vs. Microservices vs. Serverless”: Jignesh Solanki (2017). https://dev.to/jignesh_simform/evolution-of-serverless-monolithic-microservices-faas-3hdp
- [Sti21] Stieß, S. S. (2021). Tracing the impact of SLO violations on business processes across a microservice architecture with the saga pattern [Master thesis]. https://doi.org/10.18419/opus-12038