# Modular e-Commerce Data Warehouse using Microservices

**Dr. Martin Bobák**

INSTITUTE OF INFORMATICS
SLOVAK ACADEMY OF SCIENCES

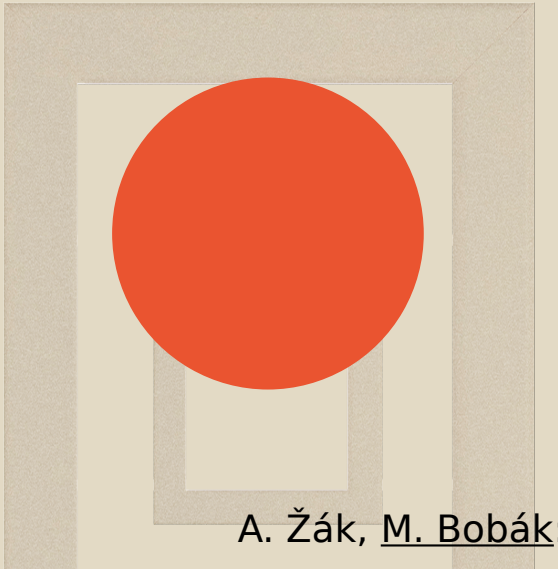
September 14, 2022


A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

01

## Introduction

Main aim and qFlow



02

## State of the art

Data warehouses and microservices

03

## Solution

Architecture and Implementation

04

## Experiments

qFlow comparison, load size testing

05

## Conclusion

Results and future work

# 01

# Introduction

Main aim and qFlow

# Aim

The main objective is to **compare** data warehouses implemented as **monolith** with data warehouses built in a **microservice** architecture.

# Objectives

**1. Design a platform for the data warehouse, which will be based on microservice architecture**

**2. Verify the proposed solution in the case study**

**3. Test and compare the proposed solution to the monolithic version of the data warehouse**

# Motivation

- **monolithic data warehouse**

  **- qFlow (SoftPoint)**
- **real data from Muziker e-shop**



A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# 02

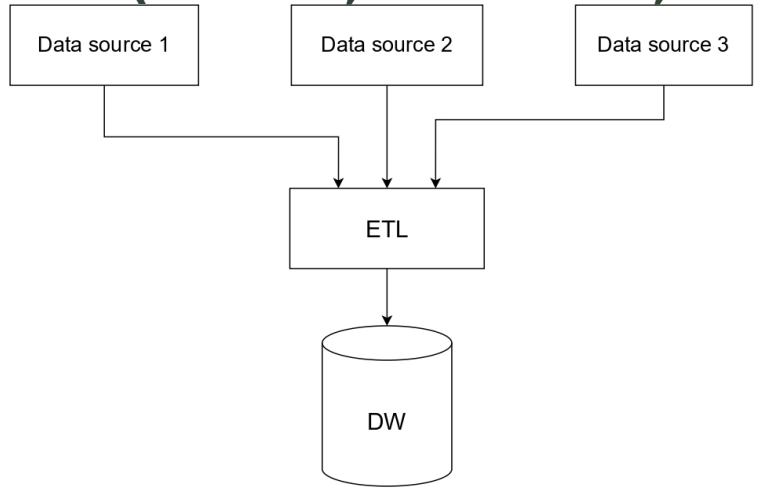
# State of the art

Data warehouses and microservices

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# Data warehouses

- **subject-oriented, integrated, time variant, and non-volatile collection of data in support of management decision making process**
- **ETL (Extract, Transform, Loa**

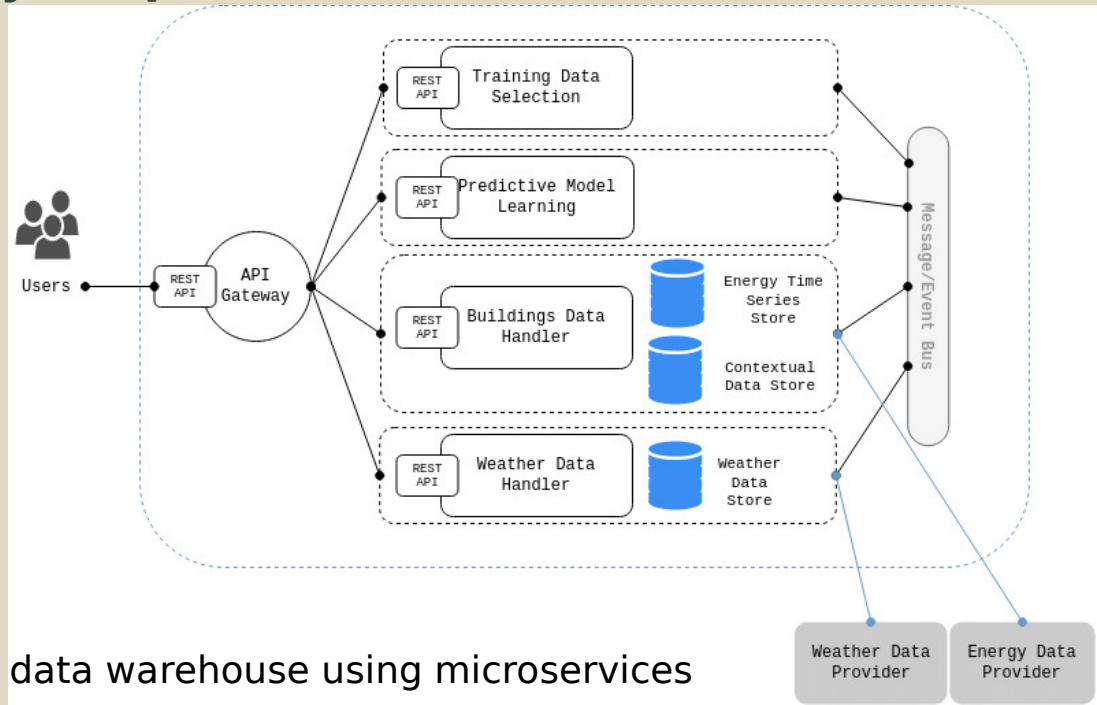



| Data warehouse | Pros | Cons |
|---|---|---|
| Amazon Redshift | Columnar storage<br>Querying language<br>Strong AWS integration | Uniqueness Not Enforced<br>Structured data<br>Not multi-cloud solution |
| Google BigQuery | Easier management of warehouse<br>Pay per use<br>Multi-cloud solution | Overkill for smaller data sets<br>Flat pricing<br>- |
| Snowflake | Easy implementation<br>Auto-scaling<br>Structured and semi-structured data | Customer is dependent on infrastructure<br>Big data load<br>No support for unstructured data. |

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# Microservices

- **architectural style that structures an application as a collection of loosely coupled services**
- **Amazon, Netflix, Spotify**



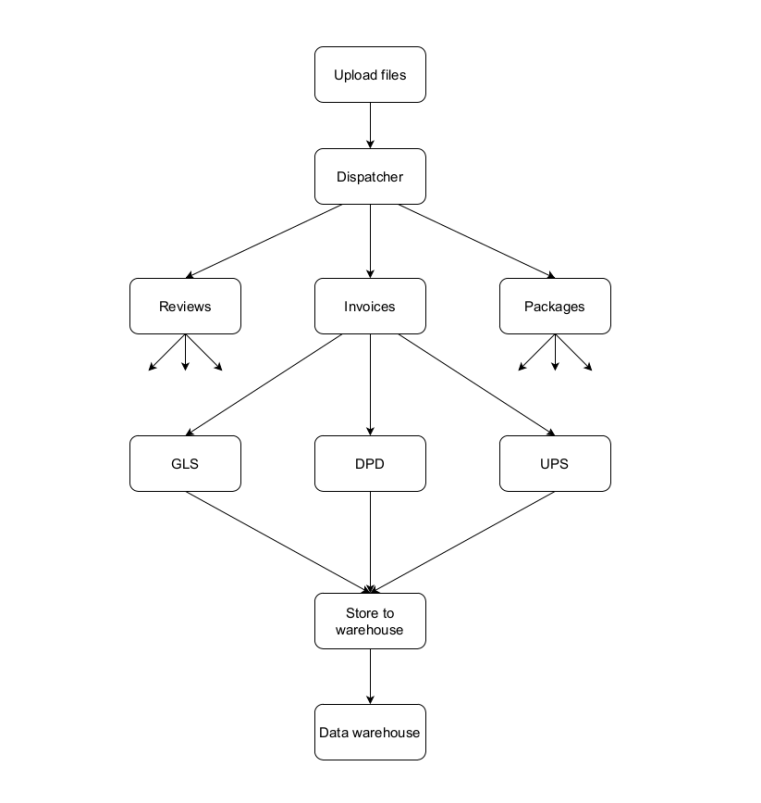

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

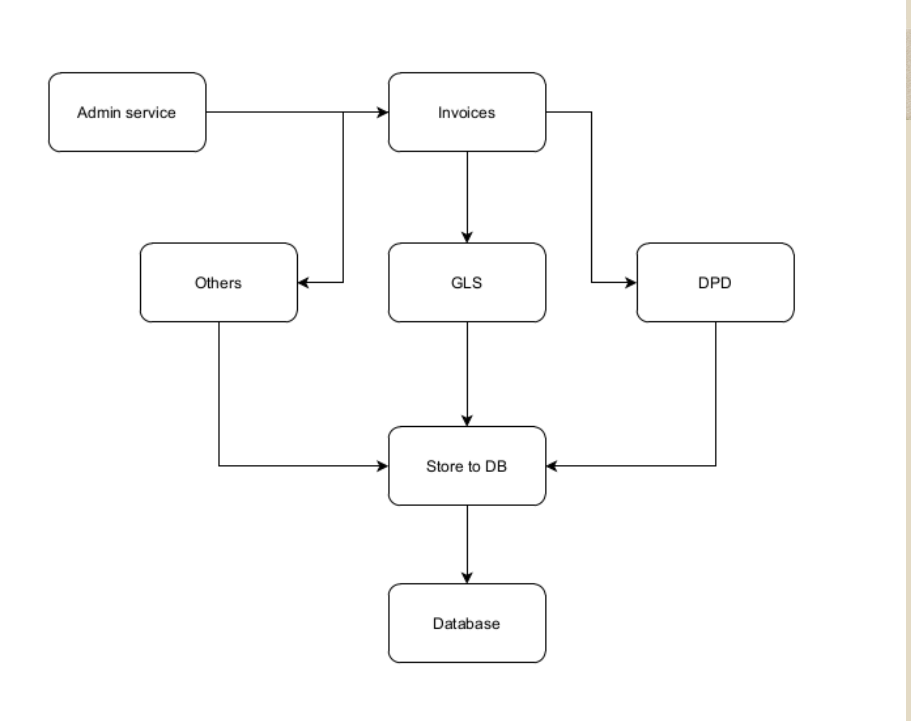
# 03

# Solution

Architecture and Implementation

# Architecture




A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# Docker-compose.yml

```yaml
version: '3.0'
services:
  admin_service:
    build: Admin-Service/.
    ports:
      - "4001:3000"
    environment:
      - POSTGRES_HOST=172.17.0.1
      - POSTGRES_DB=admin_development
      - POSTGRES_USER=adam_zak
      - POSTGRES_PASSWORD=postgres
      - POSTGRES_PORT=5432
      - REDIS_SIDEKIQ_URL=redis://172.17.0.1:6379

  processor:
    build: processors/.
    environment:
      - REDIS_SIDEKIQ_URL=redis://172.17.0.1:6379
    ports:
      - "4560-4570:4567"
    links:
      - "saver:saver"

  saver:
    build: savers/.
    ports:
        - "3950-4000:3000"
    privileged: true
    environment:
      - POSTGRES_HOST=172.17.0.1
      - POSTGRES_DB=savers_development
      - POSTGRES_USER=adam_zak
      - POSTGRES_PASSWORD=postgres
      - POSTGRES_PORT=5432
      - REDIS_SIDEKIQ_URL=redis://172.17.0.1:6379

  sidekiq-heureka:
    build: downloaders/.
    environment:
      - SINGLE=true
      - REDIS_SIDEKIQ_URL=redis://172.17.0.1:6379
      - START-JOB=false
      - SCHEDULE-JOB=true
      - CRON='*/5 * * * *'            # ak je scheduled job true, tak nastav CRON cas
    command: bundle exec sidekiq -r ./heureka_reviews_downloader.rb
```

# Directory

| | |
|---|---|
| Admin-Service | Add env for cron (dpd_rework) |
| downloaders | Last push :) |
| processors | Add env for cron (dpd_rework) |
| savers | Add env for cron (dpd_rework) |
| scripts | Commit asi po roku :( ) |
| workers | Add env for cron (dpd_rework) |
| .gitignore | Add readme and gitignore (main) |
| README.md | Add readme and gitignore (main) |
| docker-compose.yml | Version after suggestion (dpd_rework) |
| docker-compose_old.yml | Merge processors (add_sinatra) |

# 04

# Experiments

qFlow comparison, load size testing

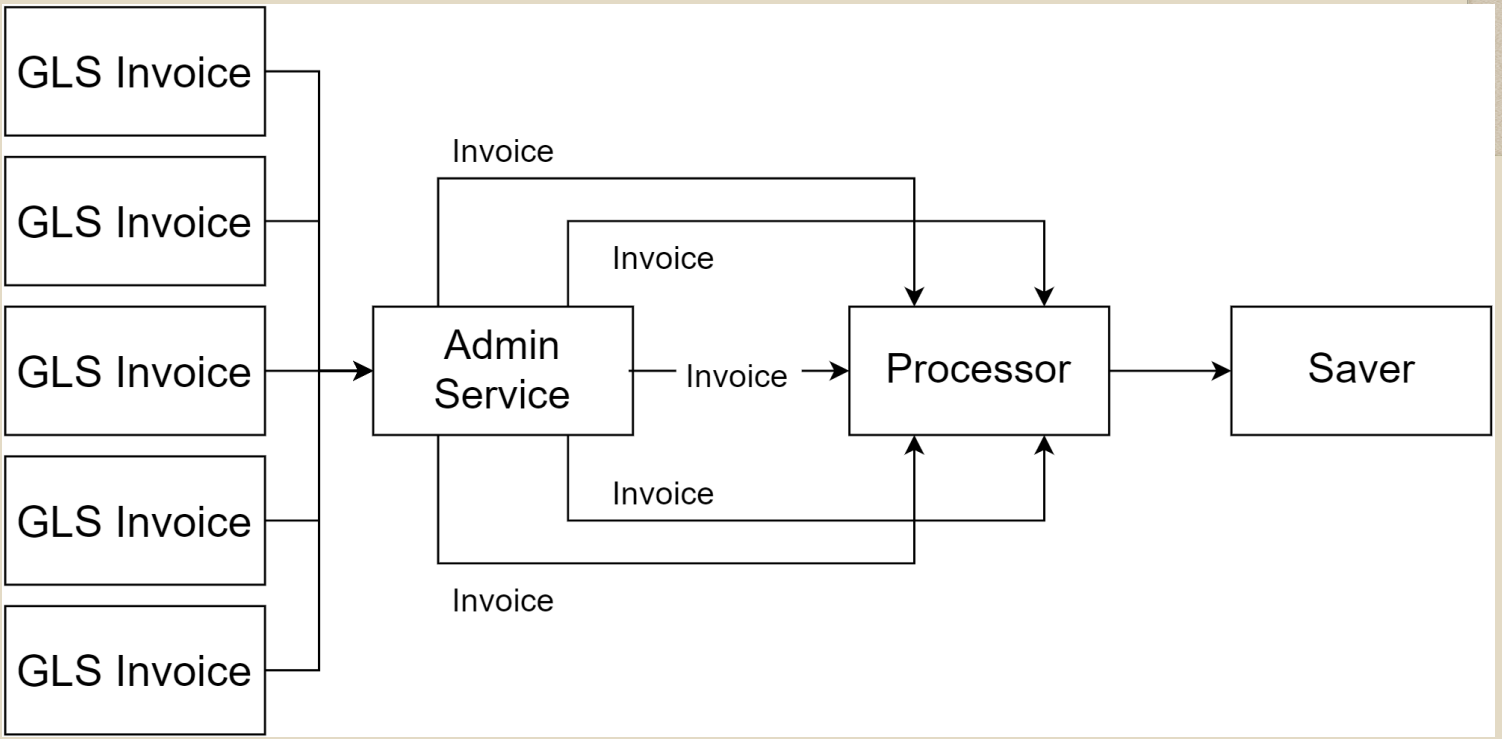
# Hypothesis

1) Microservice-based data warehouse will be bit faster than a monolithic data warehouse, but the difference will be small.

2) Microservice-driven applications can solve problems with overloaded parts of the data pipeline.

3) Microservices are most efficient when working with large data.

# Workflow

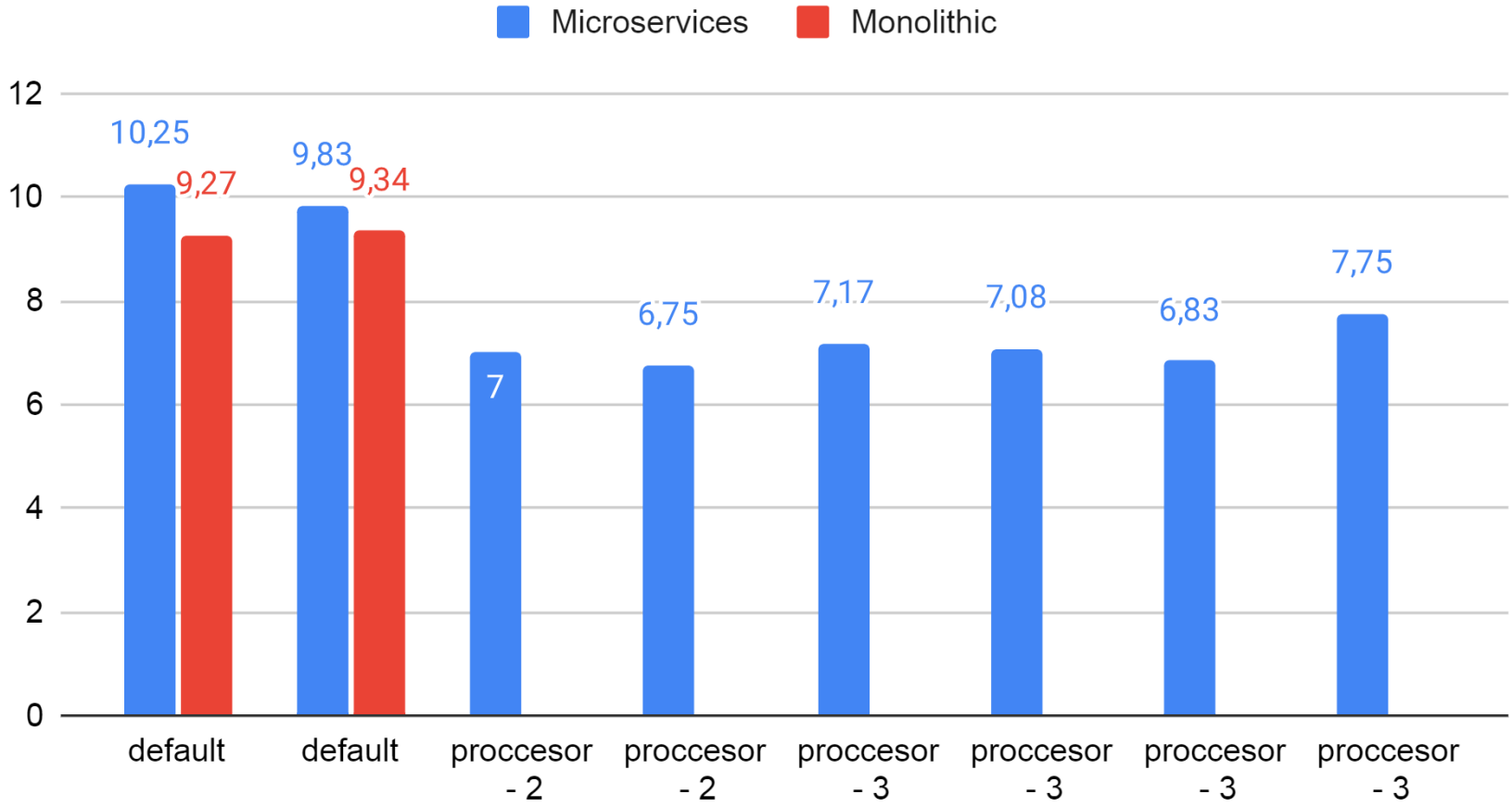



A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

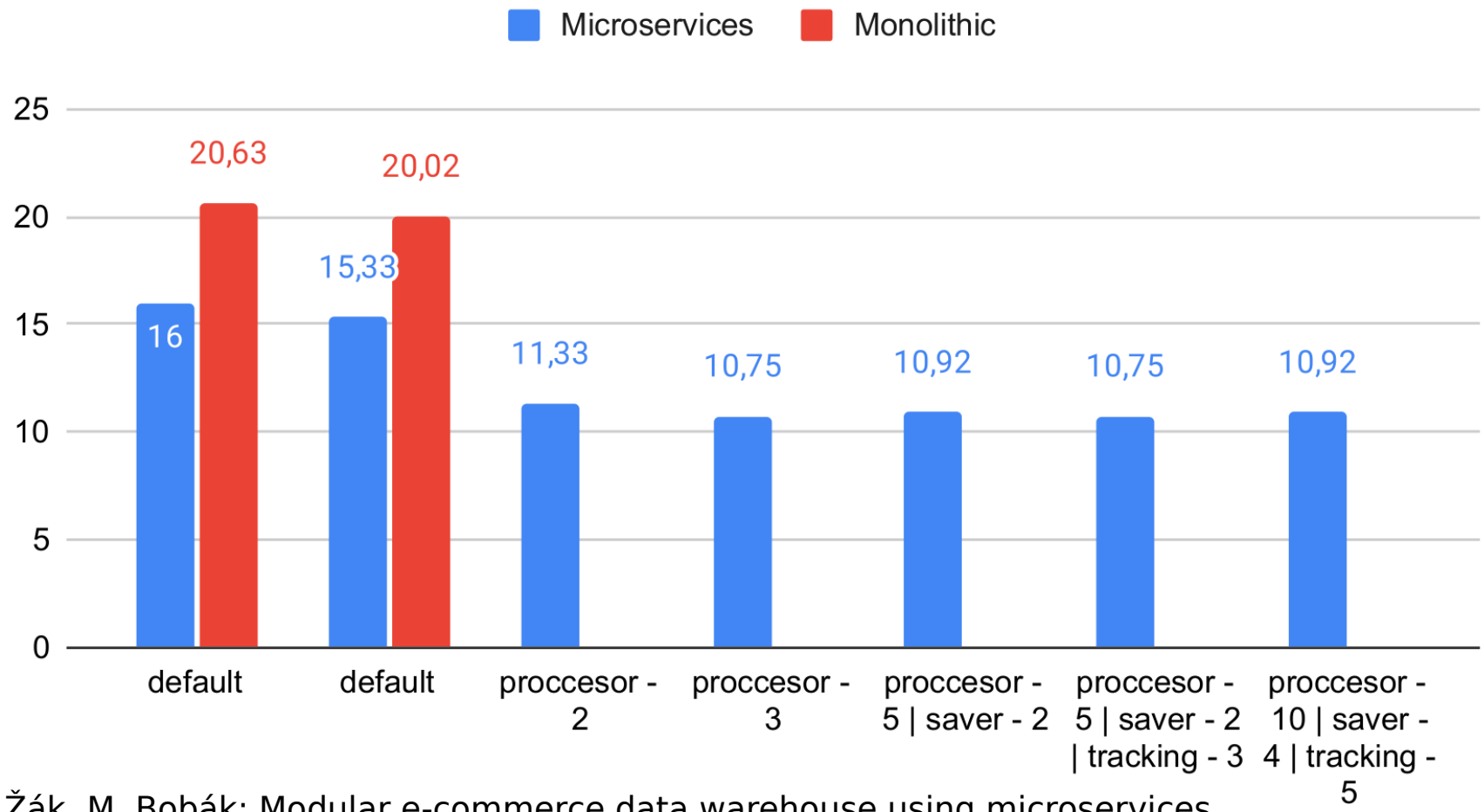
# Monolithics vs microservice-based data warehouse (First hypothesis)

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# 10 Invoices - time division

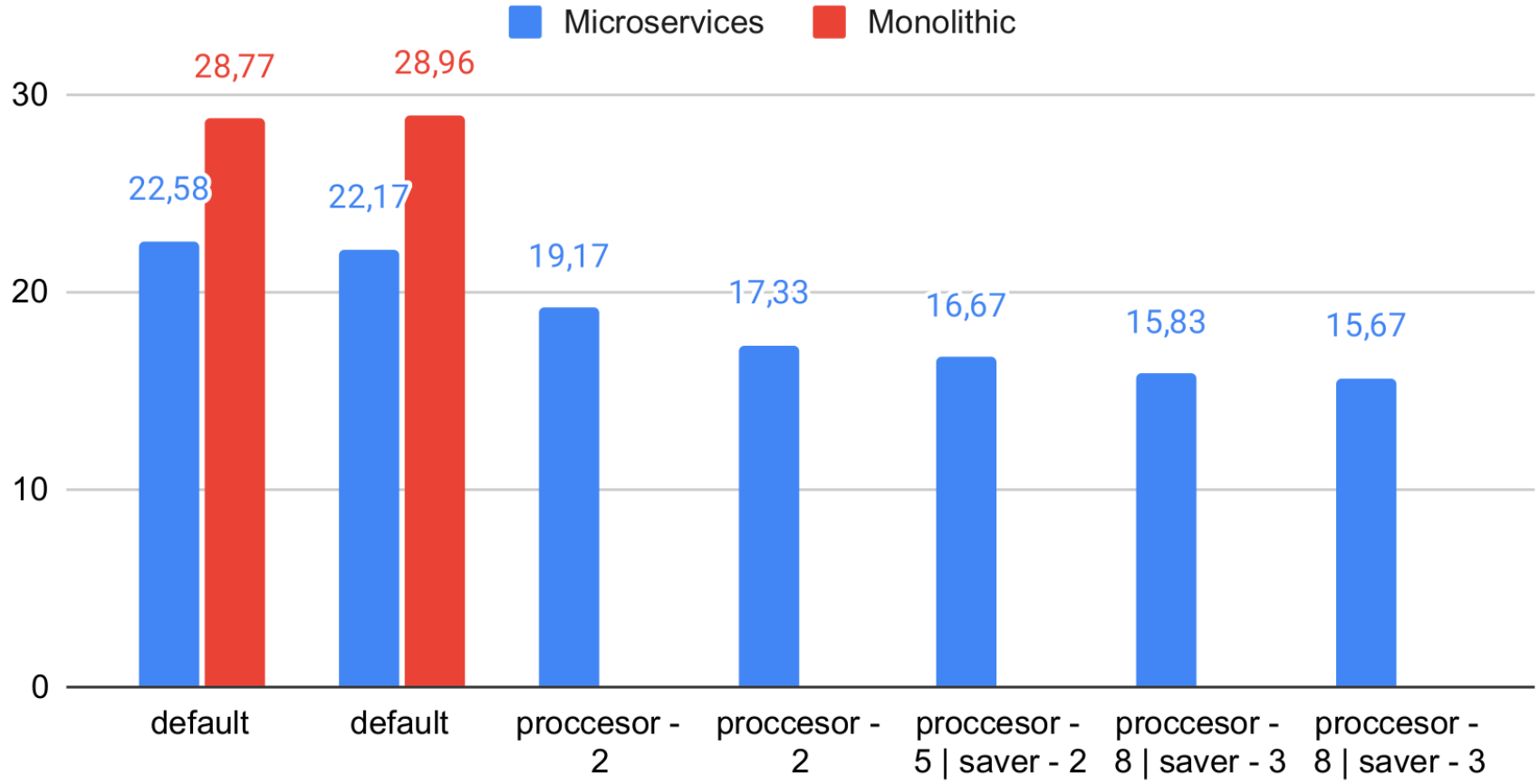


# 20 Invoices - time division




A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# 30 Invoices - time division



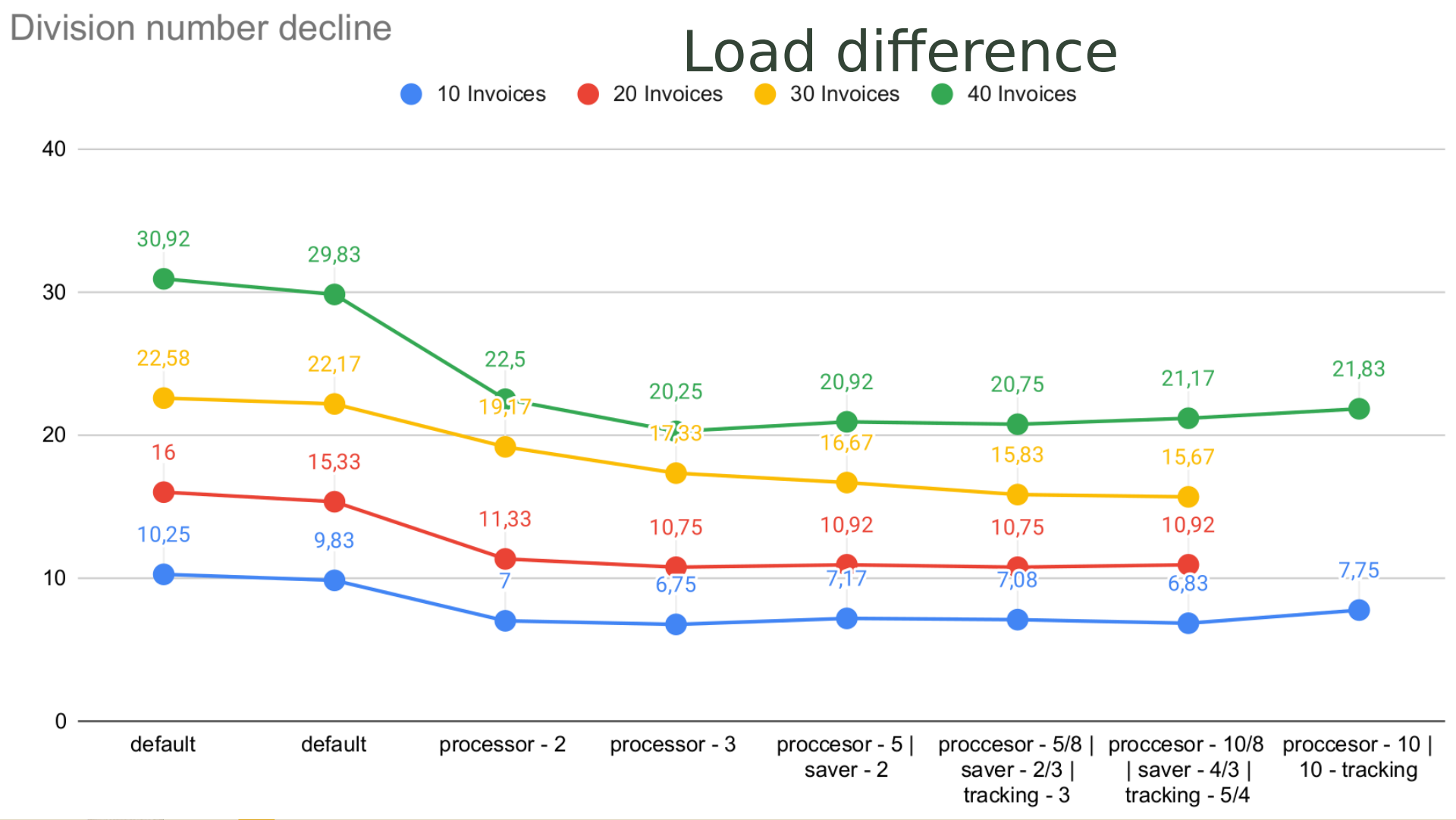
# Effectiveness evaluation (Second hypothesis)

| Test type | Execution time | Scale |
|---|---|---|
| 10 GLS invoices | 02:05 | default |
| 10 GLS invoices | 02:03 | default |
| 10 GLS invoices | 01:58 | default |
| 10 GLS invoices | 01:38 | processor - 2 |
| 10 GLS invoices | 01:31 | processor - 2 |
| 10 GLS invoices | 01:24 | processor - 2 |
| 10 GLS invoices | 01:21 | processor - 2 |
| 10 GLS invoices | 01:26 | processor - 3 |
| 10 GLS invoices | 01:25 | processor - 3 |
| 10 GLS invoices | 01:22 | processor - 3 |
| 10 GLS invoices | 01:33 | processor - 3 |
| 10 GLS invoices | 02:09 | saver - 2 |

| Test type | Execution time | Scale |
|---|---|---|
| 10 Heureka reviews | 11:21 | default |
| 10 Heureka reviews | 09:41 | processor - 2 |
| 10 Heureka reviews | 09:37 | single_review - 2 |
| 10 Heureka reviews | 07:10 | single_review - 2 — saver - 2 |
| 10 Heureka reviews | 05:29 | single_review - 5 — saver - 5 |
| 10 Heureka reviews | 03:31 | single_review - 10 — saver - 10 |
| 10 Heureka reviews | 03:20 | single_review - 20 — saver - 20 |
| 10 Heureka reviews | 03:17 | single_review - 20 — saver - 20 |
| 30 Heureka reviews | 02:05 | single_review - 30 — saver - 30 |
| 30 Heureka reviews | 01:20 | single_review - 30 — saver - 30 |
| 30 Heureka reviews | 01:19 | single_review - 30 — saver - 30 |

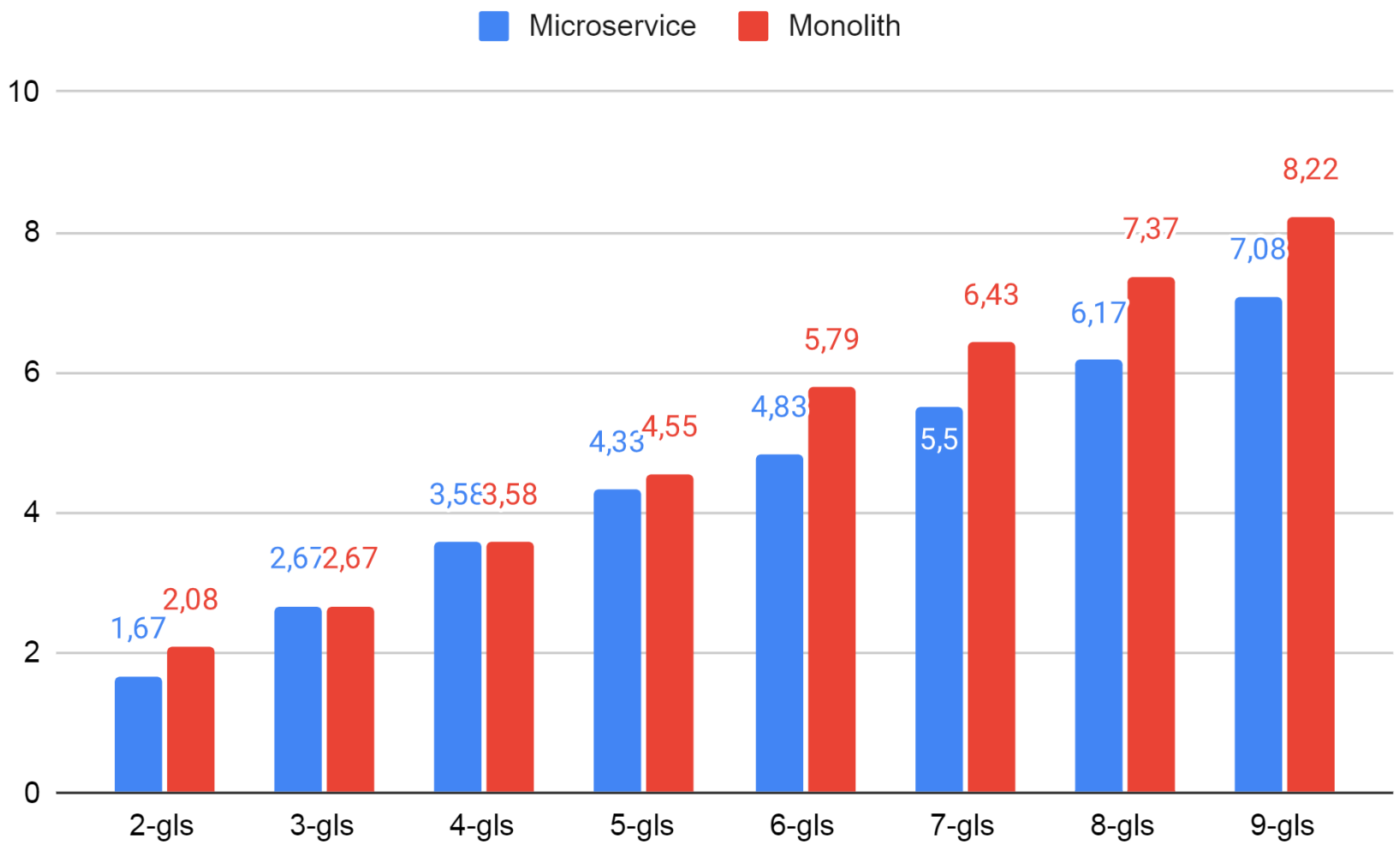
Division number decline

# Load difference

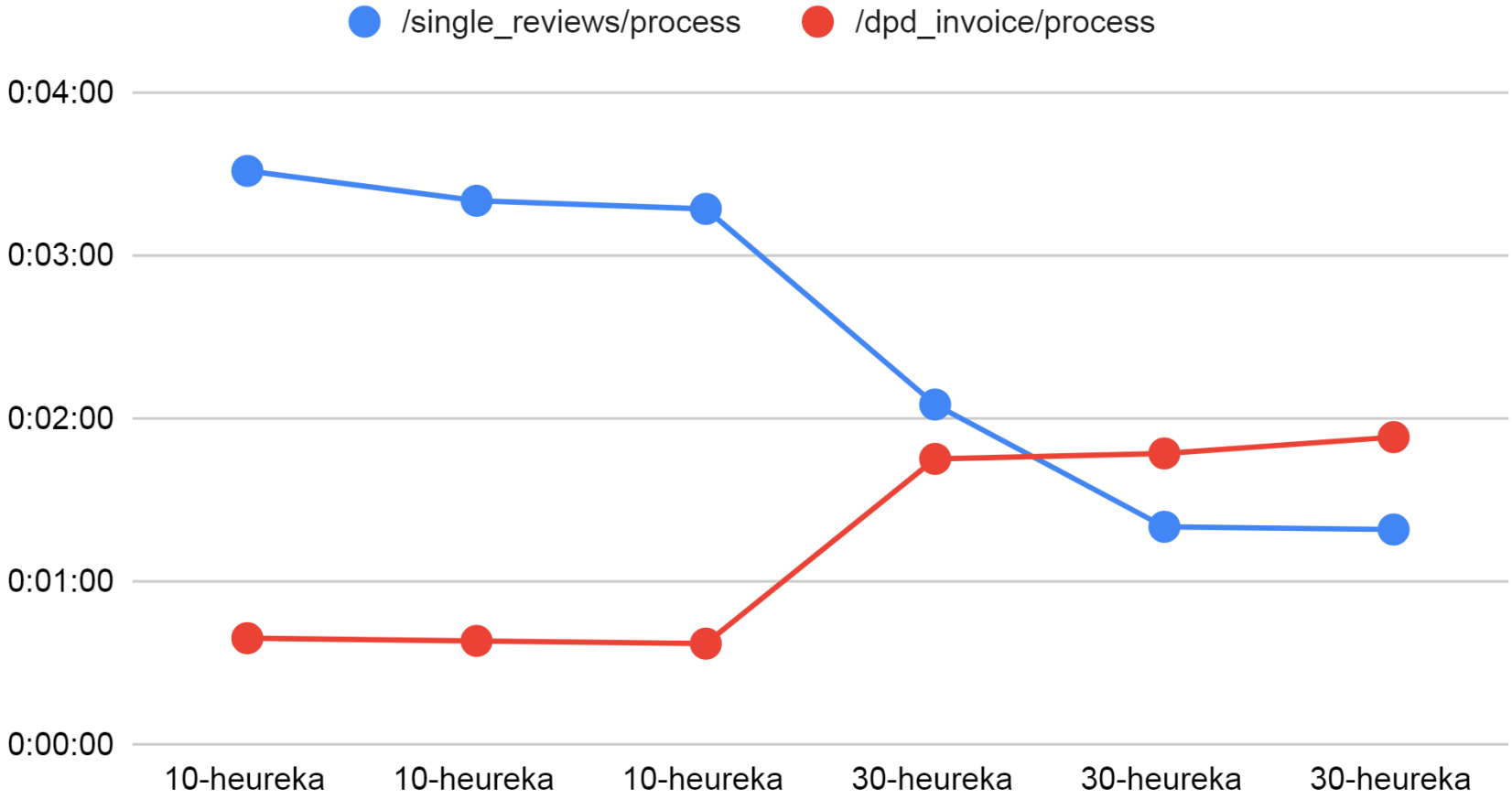


# Large data usability (Third hypothesis)

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices


Microservice
Monolith
10
8
6
4
2
0
1,67
2,08
2,67
2,67
3,58
3,58
4,33
4,55
4,83
5,79
5,5
6,43
6,17
7,37
7,08
8,22
2-gls
3-gls
4-gls
5-gls
6-gls
7-gls
8-gls
9-gls

# Time shift on endpoints with different load



# 05

# Conclusion

Results and future work

# Results

- **Scaling offers a very functional and easy way to remove bottlenecks from the program**
- **Experiments have shown that microservices are performing faster in many cases (not all cases)**
- **Microservices offer a good option for systems that work with large loads of data**

# Future work

- **Can machine learning help or even fully automate the process of scaling and deploying containers?**

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices

# Thank you for your attention!

martin.bobak@savba.sk

INSTITUTE OF INFORMATICS
SLOVAK ACADEMY OF SCIENCES

A. Žák, M. Bobák: Modular e-commerce data warehouse using microservices