

# Arquitetura de Microserviços: Um Estudo de Caso

Euller Henrique Bandeira Oliveira

Universidade Federal de Uberlândia  
Faculdade de Computação  
Graduação em Sistemas de Informação

1 de dezembro de 2023

# Sumário

- 1 Introdução
- 2 Fundamentação Teórica
- 3 Desenvolvimento
- 4 Resultados
- 5 Conclusão
- 6 Referências

## Motivação

- Monografia
  - Demonstrar
    - Aplicabilidade de meus conhecimentos adquiridos ao longo da graduação:
    - Lógica de Programação, Linguagens de Programação, Paradigmas de Programação, Estruturas De Dados, Banco de Dados, Sistemas Distribuídos, Engenharia de Software, etc.
  - Demonstrado
    - Estudo de caso de um ecossistema de microsserviços simplificado derivado de um ecossistema de microsserviço real desenvolvido no projeto e na empresa em que atuo atualmente.
- Arquitetura de Microsserviços
  - Sistema Interno da Empresa (Sistema Monolítico Legado)
    - Destinado ao controle de horas e pagamento
    - Funciona somente no Internet Explorer
    - Microsoft encerrou o seu suporte
    - Eventualmente o modo IE no Edge deixará de funcionar

## Objetivo

- Identificar e Exemplificar:
  - Vantagens e Desvantagens da Arquitetura de Microserviços após:
    - Implementar e Investigar um Ecossistema de Microserviços

## Justificativa

- Necessidade:
  - Criar uma referência que possa ser:
    - Consultada por qualquer desenvolvedor que queira entender ou obter razões para adotar a arquitetura de microserviços.

## Metodologia

- Metodologia Científica
  - Estudo de Caso
    - 1 Identificação de um problema de pesquisa
    - 2 Levantamento dos dados
    - 3 Análise do contexto
    - 4 Conclusões sobre o problema  
[Menezes, 2023]
  - Pesquisa Exploratória
    - Estudar profundamente tema antes de elaborar hipóteses
    - Objetivo: Atribuí-las embasamento teórico e científico  
[Significados, 2023]
  - Pesquisa Qualitativa
    - Universo de significados, motivos, aspirações, crenças, valores e atitudes  
[Coelho, 2017]

## Metodologia

- Metodologia de Desenvolvimento

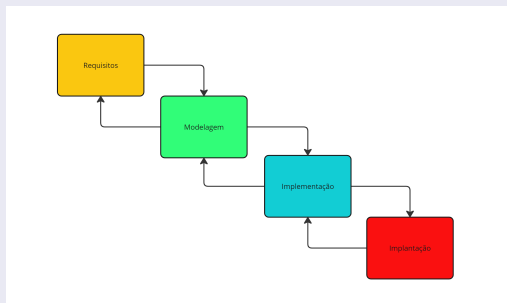


Figura: Modelo Iterativo de Desenvolvimento [Larman and Basili, 2003]

## Arquitetura Monolítica

Presente na aplicação na qual todas as suas funcionalidades estão presentes em um único sistema.

Tipos:

- Sistema monolítico de um só processo
  - Código é implantado em um único processo (Servidor)
  - Sistema distribuído simples (Sistema + Banco de Dados)
- Sistema monolítico modular de um só processo
  - Código é dividido em módulos na implementação
  - Código é unido na implantação

[Newman, 2020a]

## Arquitetura de Microsserviços

Presente na aplicação na qual cada microsserviço possui um conjunto de funcionalidades.

Pilares:

- Implantações independentes
  - Ao alterar um microsserviço, é necessário implantar somente ele.
- Design orientado a domínios (DDD)
  - Os requisitos, modelagem e implementação de cada microsserviço devem se basear em um domínio de negócio geral ou específico.
- Responsáveis pelos seus próprios dados
  - Cada microsserviço deve ser capaz de acessar e manipular apenas um determinado conjunto de tabelas.

[Newman, 2020b] [Newman, 2022]



## Arquitetura Cliente-Servidor

Cliente:

- Solicitar uma ação

Servidor:

- 1 Receber solitação
- 2 Realizar ação
- 3 Retornar Status ou Status e Dado

[Native, 2022]

## Arquitetura Orientada a Eventos

- Programação Assíncrona
  - Sistema que solicita a ação não precisa esperar pela resposta do outro sistema
  - Fluxo do sistema não é interrompido

[Tibco, 2023]

- Componentes:
  - Produtor: Notar evento, Enviar mensagem
  - Canal: Transmitir a mensagem
  - Consumidor: Receber a mensagem, Realizar ação

[RedHat, 2019]

- Modelos:
  - Pub/Sub: Mensagem é excluída do canal após ser recebida
  - Streaming: Mensagem não é excluída do canal após ser recebida

[Microsoft, 2023b]

## Tecnologias Utilizadas

- Banco de Dados
  - Sql
  - Microsoft Sql Server
- Ecossistema de Microserviços
  - Java, Maven, Spring Framework, Spring Boot, JPA, Hibernate, Rest Template, Apache Kafka, Spring Email, Lombok, Sonar Lint, Jasper Reports, Jaspersoft Studio, Microsoft Azure Key Vault, IntelliJ IDEA Ultimate e Postman.
- Sistema Web:
  - Html, Css, Javascript e Visual Studio Code

## Requisitos - Requisitos Funcionais

- Tela Dia Não Útil
  - Buscar usuários ativos
  - Buscar dias não úteis
  - Salvar dia não útil
  - Excluir dia não útil
- Tela Hora Extra
  - Buscar tarefas ativas
  - Buscar usuários de uma tarefa
  - Buscar horas extras
  - Salvar hora extra
  - Excluir hora extra

## Requisitos - Requisitos Funcionais

- Tela Dia Timesheet
  - Buscar tarefas ativas de um usuário
  - Buscar horas trabalhadas
  - Buscar primeiro dia aberto
  - Buscar dia anterior/posterior
  - Copiar dia anterior
  - Fechar dia
  - Salvar timesheet
  - Excluir timesheet
- Tela Faturamento do Consultor
  - Buscar usuários ativos
  - Buscar faturamento(s)
  - Enviar email(s) do(s) faturamento(s)
  - Gerar relatório do(s) faturamento(s)

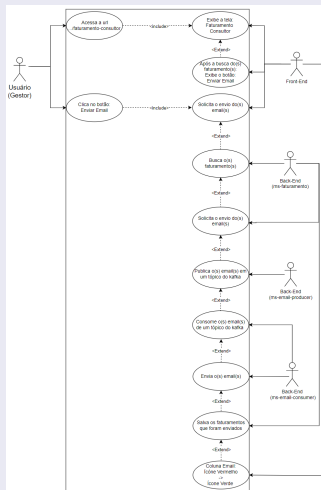
## Requisitos - Requisitos Não Funcionais

- Arquitetura: BE-BE ou BE-FE – Arquitetura Cliente-Servidor
- Arquitetura: Back-End – Arquitetura de microserviços
- Arquitetura: Mensageria – Arquitetura orientada eventos
- Data-Base – Banco de dados relacional

## Requisitos - Casos de Uso

- Copiar dia anterior
  - Fluxo normal/feliz:
    - 1. Usuário (Consultor): Acessa a url .../timesheet
    - 2. Front-End: Exibe o botão 'Copiar Dia Anterior'
    - 3. Usuário (Consultor): Clica no botão 'Copiar Dia Anterior'
    - 4. Front-End: Solicita ao ms-timesheet que o dia anterior ao escolhido seja copiado
    - ...
  - Fluxo alternativo/triste:
    - 7.1 Back-End (ms-faturamento): Se o faturamento do usuário logado no mês do dia em questão for encontrado, o status 406 (Não aceito) é retornado
    - 14.1.1 Back-End (ms-tarefa): Se o usuário associado ao dia escolhido não estiver associado a tarefa presente no timesheet em questão, o status 406 (Não aceito) é retornado
    - ...

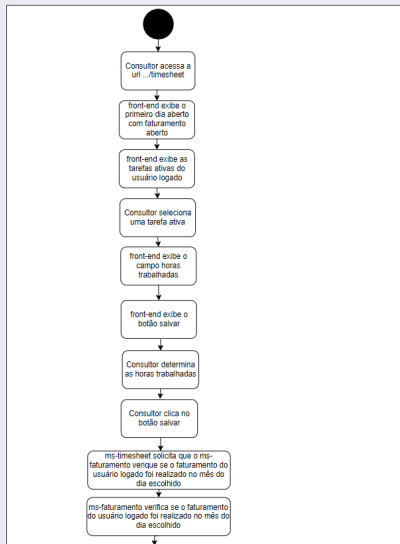
## Modelagem - Diagramas de Caso de Uso



**Figura:** Tela: Faturamento do Consultor - Enviar faturamento(s)



## Modelagem - Diagramas de Atividade



## Modelagem

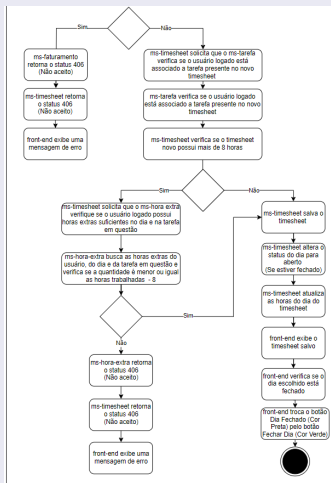


Figura: Salvar Timesheet

## Modelagem - Diagramas de Arquitetura

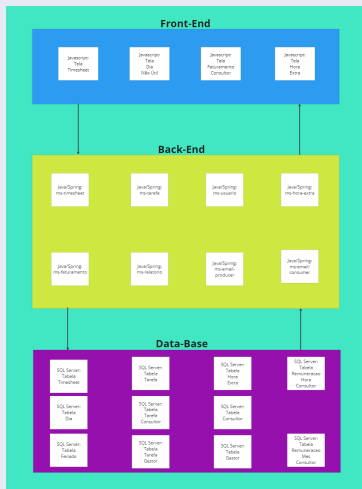


Figura: Arquitetura em Camadas

## Modelagem - Diagramas de Arquitetura

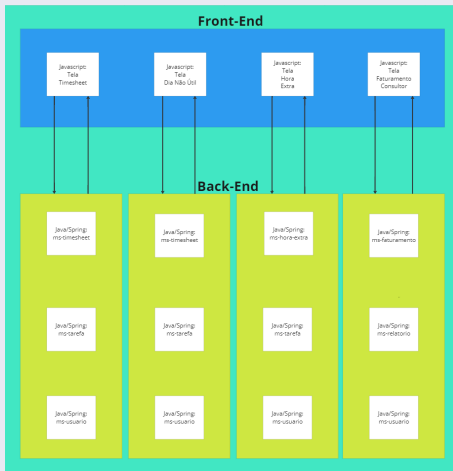


Figura: Arquitetura Cliente-Servidor

## Modelagem - Diagramas de Arquitetura

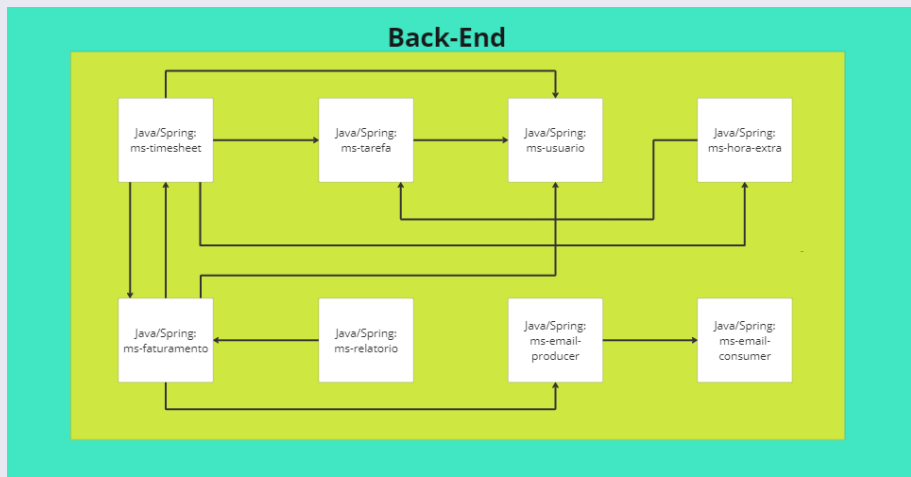


Figura: Arquitetura de Microserviços

## Modelagem - Diagramas de Arquitetura

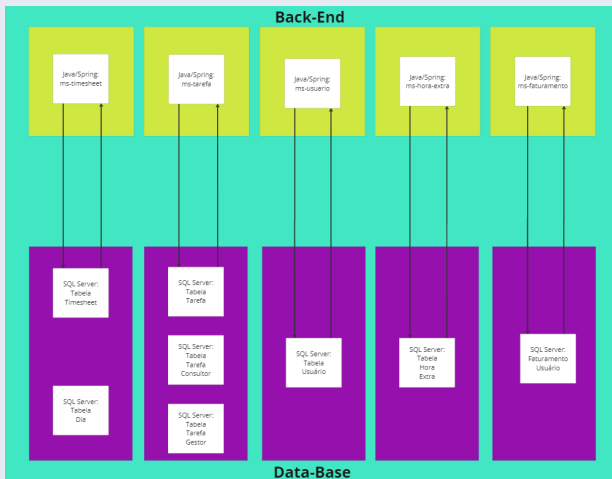


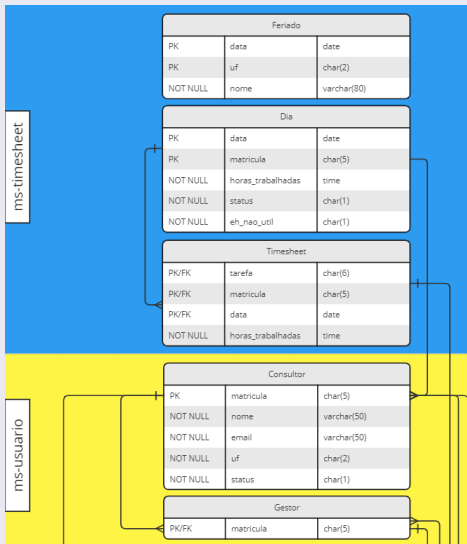
Figura: Arquitetura de Dados

## Modelagem - Diagramas de Arquitetura



Figura: Arquitetura Orientada a Eventos

## Modelagem





## Modelagem

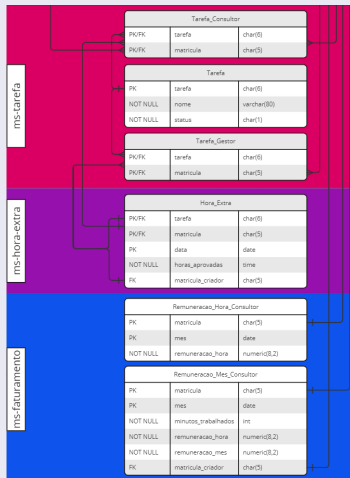


Figura: Diagrama Entidade Relacionamento

## Implementação – Back-End

- Padrão Arquitetural
  - MVC (Model View Controller)
    - Model – Service, Repository
    - View – DTO (Data Transfer Object)
    - Controller
- Padrões de Projeto
  - Service
  - Repository
  - Entity
  - Próprios (Ex: DtoService, ModelService e UtilService)

## Implementação – Front-End

### TimeSheet

< 01/09/2023 >

COPIAR DIA ANTERIOR

FECHAR DIA

00004 - Realizar Implantação

01:00

SALVAR

Dia: 04:00 | Semana: 04:00 | Mês: 04:00

TimeSheets

Tarefa	Horas	Ação
00001 - Definir Requisitos	01:00	X
00002 - Fazer Modelagem	01:00	X
00003 - Realizar Implementação	01:00	X
00004 - Realizar Implantação	01:00	X

Figura: Tela Timesheet

## Implementação – Front-End

### Dia Não Útil

00001 - Euler Henrique Bandeira Oliveira

setembro de 2023

BUSCAR

Dias

Dia	Ação
02	+
03	+
07	+
09	+
10	+
16	+
17	+
23	+
24	+
30	+

Figura: Tela Dia Não Útil

## Implementação – Front-End

### Hora Extra

00001 - Definir Requisitos

00001 - Euler Henrique Bandeira Oliveira

01/09/2023

02:00

SALVAR

setembro de 2023

Horas Extras

Tarefa	Consultor	Data	Horas	Ação
00001 - Definir Requisitos	00001 - Euler Henrique Bandeira Oliveira	01/09/2023	02:00	X

Figura: Tela Hora Extra

## Implementação – Front-End

### Faturamento Consultor

Selecione um consultor

setembro de 2023

BUSCAR

Faturamentos

GERAR RELATÓRIO

Consultor	Horas Trabalhadas	Remuneração Hora	Remuneração Mês	Email Enviado
00001 - Euler Henrique Bandeira Oliveira	163:35	R\$ 200.00	R\$ 32716.67	Não
00002 - João da Silva Pereira	85:46	R\$ 180.00	R\$ 15438.00	Não
00003 - Pedro de Souza Lima	114:12	R\$ 160.00	R\$ 18272.00	Não
00004 - José dos Santos Machado	84:55	R\$ 120.00	R\$ 10190.00	Não
00005 - Maria da Almeida Lopes	110:58	R\$ 100.00	R\$ 11096.66	Não
00006 - Ana das Lopes Martins	100:05	R\$ 80.00	R\$ 8006.67	Não
00007 - Joana dos Martins Pereira	89:45	R\$ 60.00	R\$ 5385.00	Não
00008 - Beatriz da Pereira Costa	147:37	R\$ 40.00	R\$ 5904.66	Não

ENVIAR EMAIL

Figura: Tela Faturamento Consultor

## Implementação – Front-End

Relatório: Faturamento Consultor  
Mês/Ano: 09/2023  
Remuneração Mês Total: R\$ 105.809,64

Página: 1 de 1  
Data: 25/09/2023  
Hora: 08:18:45

Matrícula	Nome	Horas Trabalhadas	Remuneração Hora	Remuneração Mês	Email Enviado
00001	Euller Henrique Bandeira Oliveira	157:35	R\$ 200,00	R\$ 31.516,66	Sim
00002	João da Silva Pereira	85:46	R\$ 180,00	R\$ 15.438,00	Sim
00003	Pedro de Souza Lima	114:12	R\$ 160,00	R\$ 18.272,00	Sim
00004	José dos Santos Machado	84:55	R\$ 120,00	R\$ 10.190,00	Sim
00005	Maria da Almeida Lopes	110:58	R\$ 100,00	R\$ 11.096,66	Não
00006	Ana das Lopes Martins	100:05	R\$ 80,00	R\$ 8.006,66	Não
00007	Joana dos Martins Pereira	89:45	R\$ 60,00	R\$ 5.385,00	Não
00008	Beatriz da Pereira Costa	147:37	R\$ 40,00	R\$ 5.904,66	Não

Figura: Tela Faturamento Consultor - Relatório

## Implementação – Front-End

Caro Consultor, Euler Henrique Bandeira Oliveira

A data do seu crédito será dia 05/10/23

Valor: R\$ 31.516,66

Atenciosamente,  
Equipe do Financeiro

Figura: Tela Faturamento Consultor - Email



## Implantação

- Servidor
  - Prazo curto
  - Custo alto
  - Conhecimento insuficiente
- Local
  - Docker
    - Java
    - Apache Kafka
    - Kafka Drop
    - PostgreSQL
    - MailHog

## Vantagens

- Atribuição mais fácil da implementação
  - Motivos:
    - Design Orientado a Domínios (DDD)
    - Responsável por seus próprios dados  
[Newman, 2020b] [Newman, 2020c] [Newman, 2022]
- Implantação mais fácil
  - Motivos:
    - Configuração própria de implantação e servidor próprio  
[Newman, 2020b] [Newman, 2022]
    - Conexão individual a uma pipeline (Oferece suporte a integração, entrega e implantação contínua (CI/CD))  
[JetBrains, 2023][Microsoft, 2023c]

## Vantagens

### - Manutenção mais fácil

- Motivos:

- Padrão arquitetural e padrões de projeto iguais
- Design Orientado a Domínios (DDD)[Newman, 2020b][Newman, 2022]

### - Escalabilidade mais fácil

- Motivos:

- Escalabilidade Vertical Individual (Scale-Up):
  - Redução/Aumento da quantidade de recursos (Processamento, armazenamento, memória RAM) (Por meio do Azure, Aws, ...)
- Escalabilidade Horizontal Individual (Scale-Out):
  - Redução/Aumento da quantidade de servidores do cluster de acordo com a demanda (Por meio do Kubernetes)

[Microsoft, 2023a] [Bouguezzi, 2023]

## Desvantagens

- Utilização limitada do hibernate
  - Motivos:
    - Responsável por seus próprios dados [Newman, 2020b] [Newman, 2022]
    - Realizar relacionamentos por meio do hibernate se torna um recurso limitado
- Inadequado para geração de relatórios extensos
  - Motivo:
    - Responsável por seus próprios dados [Newman, 2020b] [Newman, 2022]
    - Extrair muitos dados por meio da arquitetura cliente-servidor se torna inviável

## Desvantagens

- Implementação mais complexa
  - Motivo:
    - Decomposição de um microserviço não é uma atividade simples e trivial
    - Necessidade de ter que criar vários métodos, classes e pastas para que um microserviço possa se conectar totalmente a outro
    - Possibilidade de ter que desenvolver cada parte de uma funcionalidade em microserviços diferentes
    - Necessidade de ter que obter determinados dados de outros microserviços
- Utilização limitada de Transactions
  - Motivo:
    - Responsável por seus próprios dados [Newman, 2020b] [Newman, 2022]
    - Utilização limitada da lógica das transações

## Consideração Final

- Cumpriu:
  - Objetivo
  - Justificativa
- Proporcionou:
  - Conhecimento Teórico
  - Conhecimento Prático
- Inspirou
  - Adquirir mais conhecimentos da Arquitetura de Microserviços
  - Adquirir mais conhecimentos do Desenvolvimento Dev Ops

# Referências I



Bouguezzi, S. (2023).

Differences between scaling horizontally and vertically.



Coelho, B. (2017).

Pesquisa qualitativa: entenda como utilizar essa abordagem de pesquisa.



JetBrains (2023).

Integração contínua vs. entrega vs. implantação.



Larman, C. and Basili, V. R. (2003).

Iterative and incremental development: A brief history.



Menezes, P. (2023).


Estudo de caso.





Microsoft (2023a).


Criar aplicativos para dimensionamento.


# Referências II

 [Microsoft \(2023b\)](#).  
Estilo de arquitetura controlada por evento.

 [Microsoft \(2023c\)](#).  
O que é o azure pipelines?

 [Native, C. \(2022\)](#).  
Arquitetura cliente servidor.

 [Newman, S. \(2020a\)](#).  
*Migrando sistemas monolíticos para microserviços*.  
Novatec Editora.

 [Newman, S. \(2020b\)](#).  
*Migrando sistemas monolíticos para microserviços*.  
Novatec Editora.



# Referências III



Newman, S. (2020c).

*Migrando sistemas monolíticos para microserviços.*

Novatec Editora.



Newman, S. (2022).

*Criando Microserviços.*

Novatec Editora.



RedHat (2019).

O que é arquitetura orientada a eventos?



Significados (2023).

Pesquisa exploratória (estudo exploratório).



Tibco (2023).

O que é arquitetura orientada a eventos?