**Projeto de Módulo**

**Arquitetura de Microsserviços e Mobile**

Sumário

[Introdução 4](#_Toc51520077)

[Etapa de Análise utilizando *UML* 5](#_Toc51520078)

[*Problem Statement* 5](#_Toc51520079)

[Mapeamento Problema x Solução 6](#_Toc51520080)

[Glossário 7](#_Toc51520081)

[Lista de Requisitos 9](#_Toc51520082)

[Casos de Uso 10](#_Toc51520083)

[Diagrama de Contexto 10](#_Toc51520084)

[Diagrama de Casos de Uso do Módulo de Gestão de Clientes 11](#_Toc51520085)

[Diagrama de Casos de Uso do Módulo de Gestão de Vagas 12](#_Toc51520086)

[Diagrama de Casos de Uso do Módulo de Gestão de Processos Seletivos 13](#_Toc51520087)

[Diagrama de Casos de Uso do Módulo de Busca de Oportunidades 14](#_Toc51520088)

[Diagrama de Casos de Uso do Módulo de Busca de Candidatos 15](#_Toc51520089)

[Diagrama de Classes de Domínio 16](#_Toc51520090)

[Etapa de Projeto utilizando ***C4 Model***, ***Domain-Driven Design*** e outras práticas ágeis 17](#_Toc51520091)

[*C4 Model* 17](#_Toc51520092)

[1º Nível – Contexto 17](#_Toc51520093)

[2º Nível – *Containers* 18](#_Toc51520094)

[3º Nível – Componentes 18](#_Toc51520095)

[4º Nível – Código 18](#_Toc51520096)

[Microsserviços Propostos 19](#_Toc51520097)

[Microsserviços necessários 19](#_Toc51520098)

[*Roadmap* 19](#_Toc51520099)

[Microsserviços Implementados 20](#_Toc51520100)

[Seleção de Microsserviços para implementação 20](#_Toc51520101)

[Justificativa dos Serviços Implementados 20](#_Toc51520102)

[Código 20](#_Toc51520103)

[Arquitetura e Design do Código 20](#_Toc51520104)

[Apresentação (Consumo dos Microsserviços) 20](#_Toc51520105)

[Implantação 20](#_Toc51520106)

[Comunicação 20](#_Toc51520107)

[I AM (autenticação e autorização) 21](#_Toc51520108)

[Publicação 21](#_Toc51520109)

[Apresentação (PPT) 21](#_Toc51520110)

[Conclusão 22](#_Toc51520111)

[Referências 23](#_Toc51520112)

# Introdução

A proposta deste trabalho é apresentar maneiras de se documentar softwares baseados em arquitetura de Microsserviços.

Para tal foi utilizada parte da *UML* na fase de Análise para entendimento das necessidades e delinear a próxima fase. Na fase de projeto, foi utilizada uma combinação de técnicas de modelagem como *C4 Model*, *Domain-Driven Design* e outras práticas ágeis como cartões *CRC*.

Para provar os conceitos será criada uma implementação em Java utilizando a documentação proposta, cujo código será disponibilizado na plataforma de repositórios de código [*GitHub*](https://github.com/cirilojunior/infnet-mit-arquitetura-bloco-microservicos).

# Etapa de Análise utilizando *UML*

## *Problem Statement*

A *IT Job Hunters* Ltda é uma empresa que presta serviços de Recursos Humanos terceirizados para diversas empresas do mercado da Região Sudeste e possui mais de 20 anos de experiência com foco na seleção de candidatos para vagas para empresas de Tecnologia da Informação, tais como Analistas de Sistemas, Desenvolvedores, Administradores de Banco de Dados, assim como perfis de gestão ou processos como Gerentes de Projetos, *Product Owners*, *Scrum Master* e Agile *Coaches*.

Com a crescente demanda por mão de obra qualificada, houve um aumento substancial de clientes, assim como algumas aquisições ou fechamento de parcerias com outras empresas do ramo, expandindo sua atuação para outras regiões do Brasil. Além disso, a empresa passou a realizar a intermediação de vagas para empresas estrangeiras, tanto para expatriação quanto para trabalho remoto.

Apesar de pioneira em informatização de seus processos, possui um grande legado de sistemas especializados e não integrados, além de sistemas criados para necessidades específicas de alguns clientes. Tais sistemas já não suportam a carga de trabalho da empresa e as necessárias mudanças de processos, pondo em risco seu protagonismo em seu nicho e preocupando todo o *board* de gestão.

Para superar esses desafios, a empresa utilizará tanto equipe de TI interna, mas também realizará algumas contratações temporárias de acordo com a etapa em questão, mas com possibilidade de contratação definitiva ao final do processo. Essa dinâmica será inclusive utilizada, sempre que possível, como piloto dos próprios módulos desenvolvidos.

Todo o processo será revisto. Os problemas começam na captação de bons candidatos, na prospecção de oportunidades nos clientes, seguido do *match* entre as necessidades dos clientes e a base de currículos que ainda é manual, demorado e depende muito da experiência do analista que atua em determinado cliente, e deverá ser um dos pontos de maior destaque nessa nova empreitada.

O novo processo que vem sendo discutido internamente passa por módulos de captação de candidatos com detalhe para seus skills e experiências técnicas, de área de negócio, empresas (conhecimento de cultura organizacional).

## Mapeamento Problema x Solução

|  |  |
| --- | --- |
| O problema | Processos de negócios lentos e difícil escala |
| Afeta | A velocidade da área fim da empresa |
| Cujo impacto é | Falha na identificação de oportunidades nos clientes e perda do *time* de alguns negócios. |
| Tendo como solução | A criação de uma nova solução de *software* baseada em arquitetura de microsserviços e utilizando uma série de novas práticas de desenvolvimento empiricamente comprovadas pelo mercado. |

## Glossário

1. Mercado – Região do cliente ou vaga.
2. Seleção de Candidatos – Processo de seleção de candidatos.
3. Vaga – Posto de Trabalho requerido por um cliente. Pode ser presencial ou remota.
4. Oportunidade – Necessidade identificada pelos nossos analistas em um determinado cliente que poderá se tornar uma vaga ou encontrada pelos processos automáticos.
5. Cliente – cliente da ***IT Job Hunters***.
6. Perfil – perfil de vaga e perfil de candidato. Usado nas rotinas de match.
7. Processo – Processo de Negócio e Processo Seletivo.
8. Processo de Negócio – Processos internos da empresa.
9. Processo Seletivo – Processos de busca de candidatos para vagas.
10. Analista – colaborador que atua na identificação de necessidades dos clientes, busca e avaliação por candidatos e processo de contratação.
11. Gestão – colaborador que ocupa cargo de gestão e interage em funções de aprovação e acompanhamento por relatórios.
12. Aquisição – Aquisição de empresas do ramo que passam a fazer parte do grupo.
13. Fechamento de Parcerias – Parcerias com outras empresas do ramo em mercados ainda não atingidos, aplicando o processo interno da empresa (marca).
14. Expatriação – Processo de seleção que envolve a mudança do candidato para o país da vaga. A empresa auxilia nesse processo como um produto extra pago pelo cliente.
15. Sistemas especializados – Sistemas especializados em determinados processos de negócio, que agora seriam módulos de um novo sistema.
16. Integrados – Os sistemas antigos não eram integrados ou possuíam algum nível de integração rudimentar, como integração direto pela base de dados. A ideia agora é existir uma integração mais bem orquestrada e aderente aos novos padrões arquiteturais praticados no mercado de desenvolvimento de software. A empresa é focada em profissionais da área e precisa dar exemplo.
17. Contratação Temporária – Vaga com tempo definido para execução de um projeto específico.
18. Contratação Permanente – Vaga cujo perfil interessa ao cliente como parte definitiva da equipe (tempo indeterminado).
19. Piloto – Execução do processo no próprio sistema sendo desenvolvido (módulos existentes) e que pode evoluir para o conceito de demanda / projeto interno de seleção de vagas para um cliente.
20. Captação – Processo de busca por candidatos.
21. Candidato – Aspirante a uma vaga.
22. Prospecção – Processo de análise constante de necessidades internas dos clientes que demandem contratações.
23. Match – evento de descoberta de candidatos para vagas.
24. Base de Currículos – Base com informações de candidatos e seus perfis (habilidades, experiências e senioridade).
25. Negócio – *Core* da empresa que são os Processos Seletivos.
26. Colaborador – Profissional que participa do processo seletivo intermediando contatos entre Candidato e Cliente.

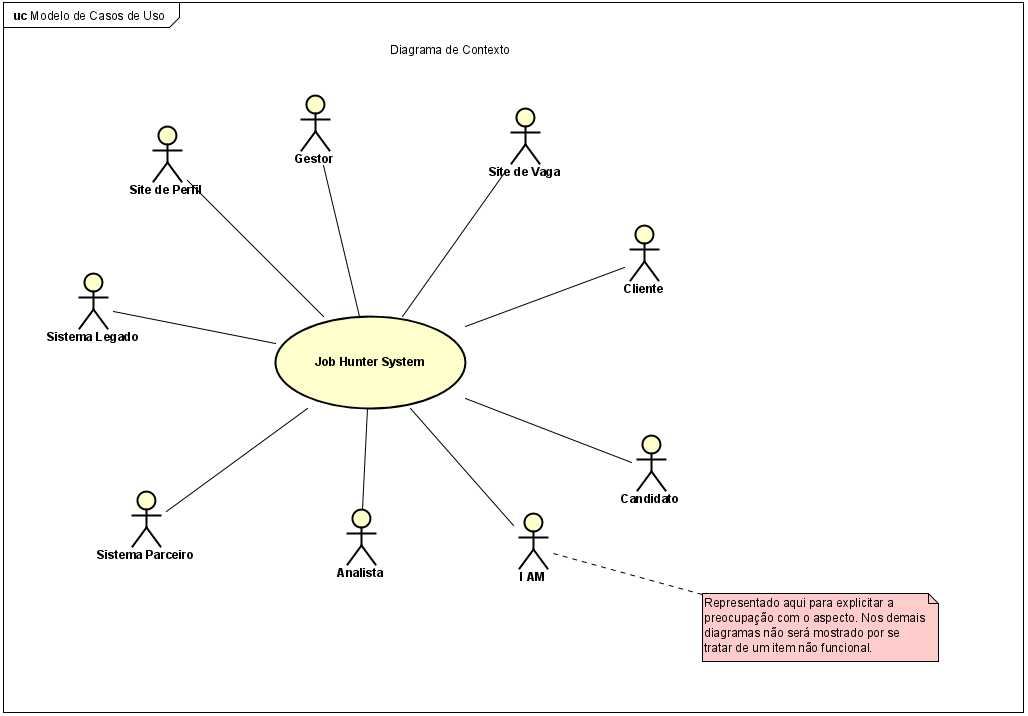
## Lista de Requisitos

* Gestão de Clientes
  + Mercados
* Gestão de Vagas
  + Perfis
    - Cargo
    - Nível
    - Habilidades
    - Experiências
  + Vagas
    - Presencial (padrão)
    - Remota
    - Aprovação pelo Cliente
* Gestão de Processos Seletivos
  + Candidatos
    - Cargo
    - Nível
    - Habilidades
    - Experiências
  + Contratação
    - Confirmação do Candidato
* Busca de Oportunidades
  + Sites de vagas conhecidos (LinkedIn, APInfo, Vagas.com.br, etc)
  + Sites e bases de clientes
  + Identificadas por Analista
* Busca Candidatos
  + Sites de perfis conhecidos (LinkedIn, APInfo, etc)
* Gestão de Parcerias \*
  + Integração com empresas de mercados onde a *IT Job Hunters* não possui escritório e atua em parceria com outras empresas da área
* Gestão de Aquisições \*
  + Integração rápida pós-aquisição e adaptação de processos com os sistemas de empresa adquirida

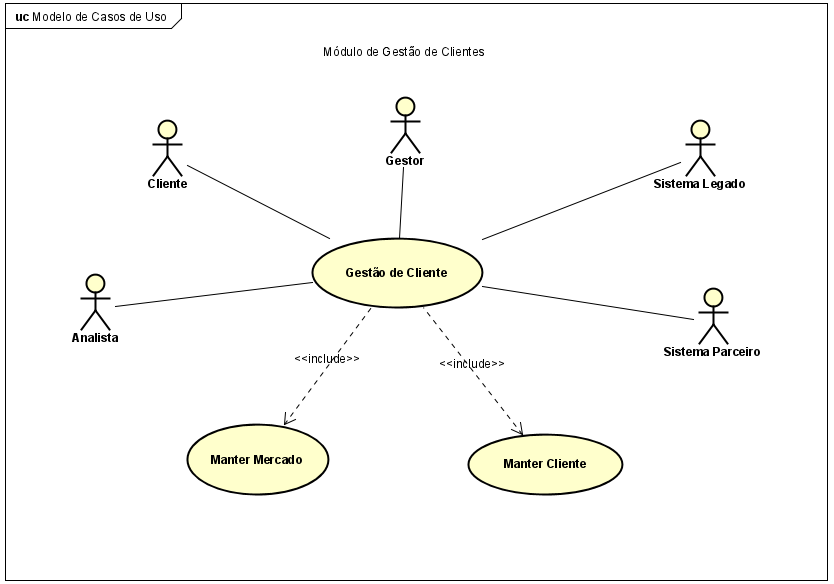
\* Descritos porque fazem parte da contextualização do trabalho, mas como são áreas muito específicas e fogem ao senso comum de atividades de um processo seletivo não serão abordados em detalhes e demais seções do trabalho.

## Casos de Uso

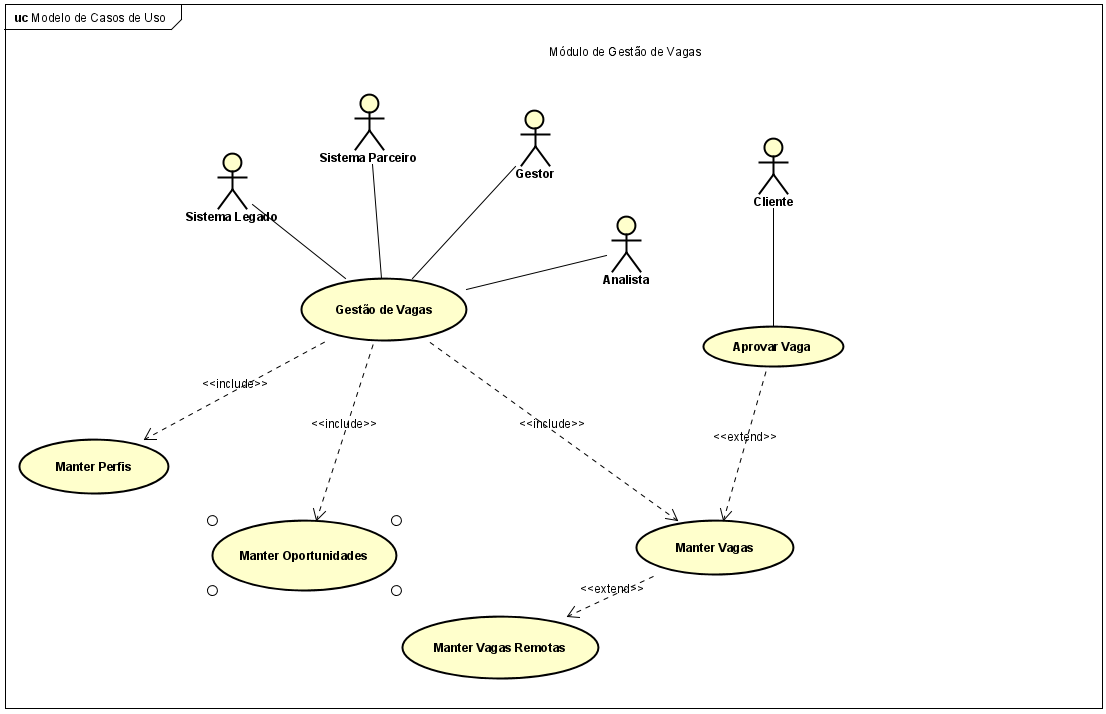
### Diagrama de Contexto



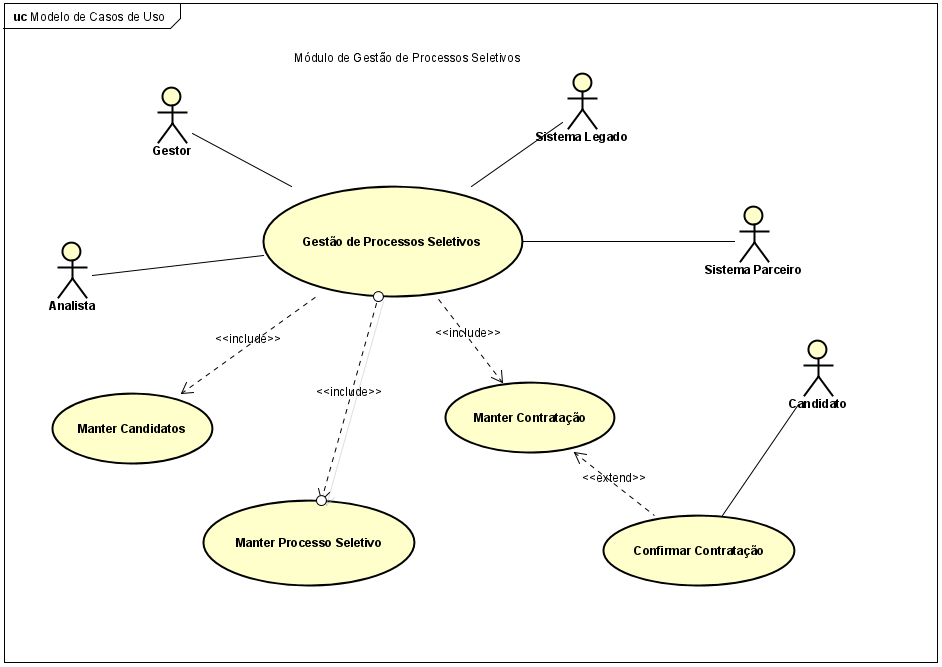
### Diagrama de Casos de Uso do Módulo de Gestão de Clientes



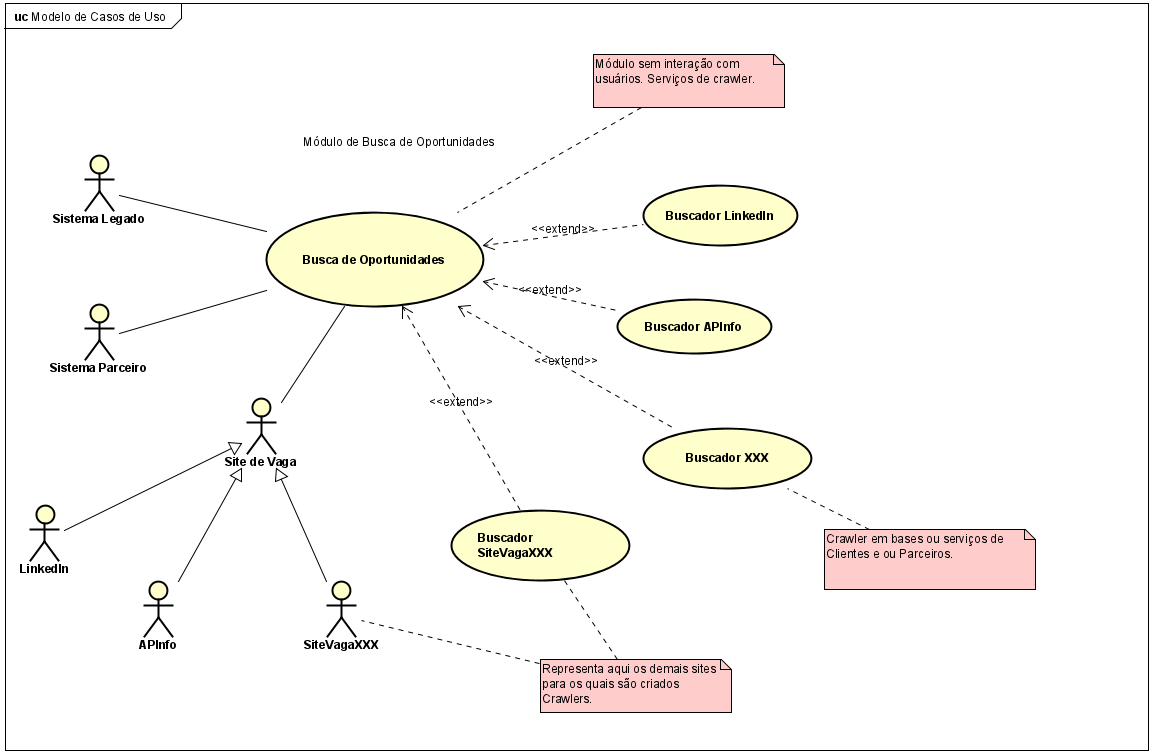
### Diagrama de Casos de Uso do Módulo de Gestão de Vagas



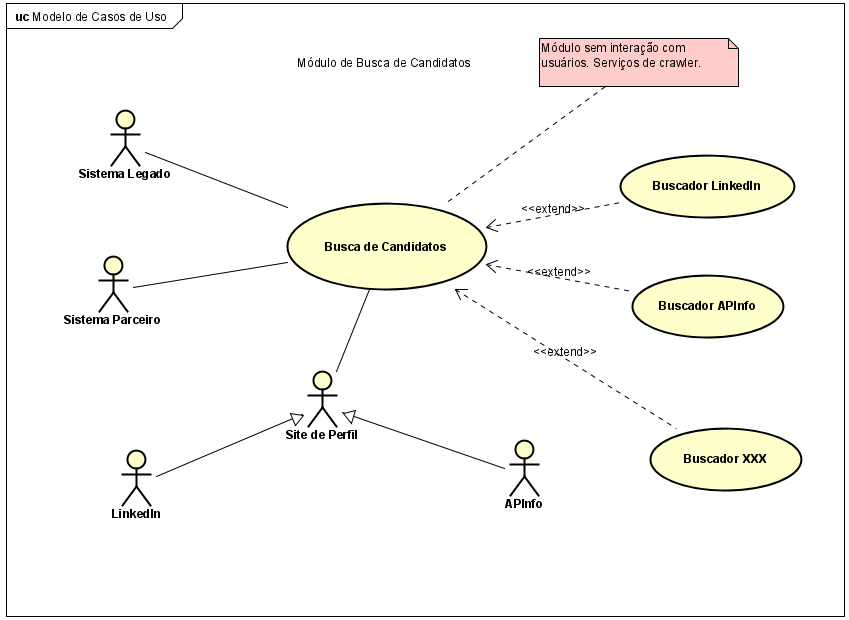
### Diagrama de Casos de Uso do Módulo de Gestão de Processos Seletivos



### Diagrama de Casos de Uso do Módulo de Busca de Oportunidades



### Diagrama de Casos de Uso do Módulo de Busca de Candidatos

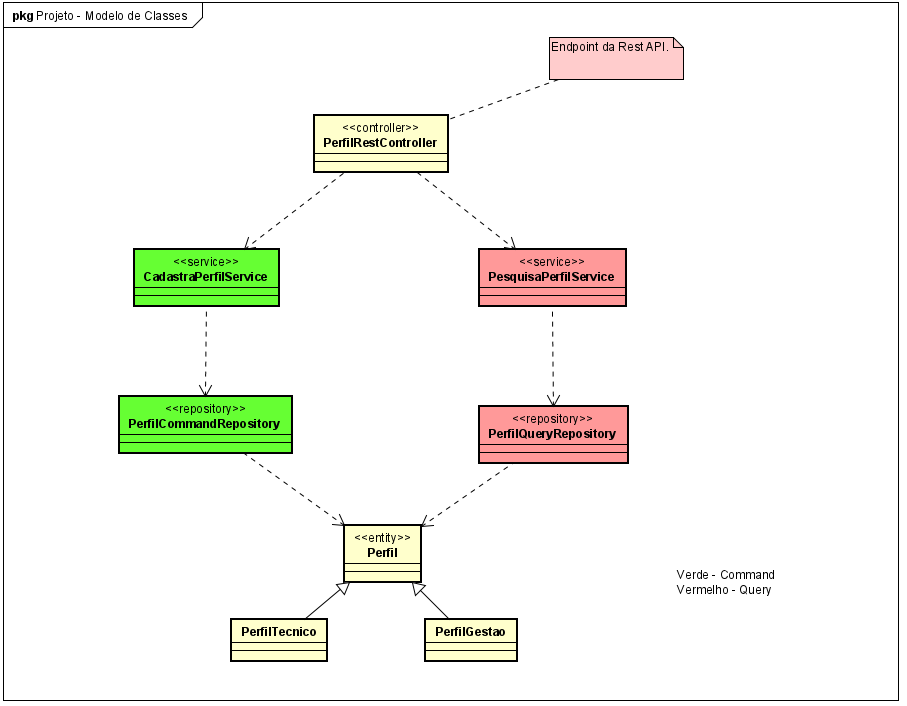


## Diagrama de Classes de Domínio

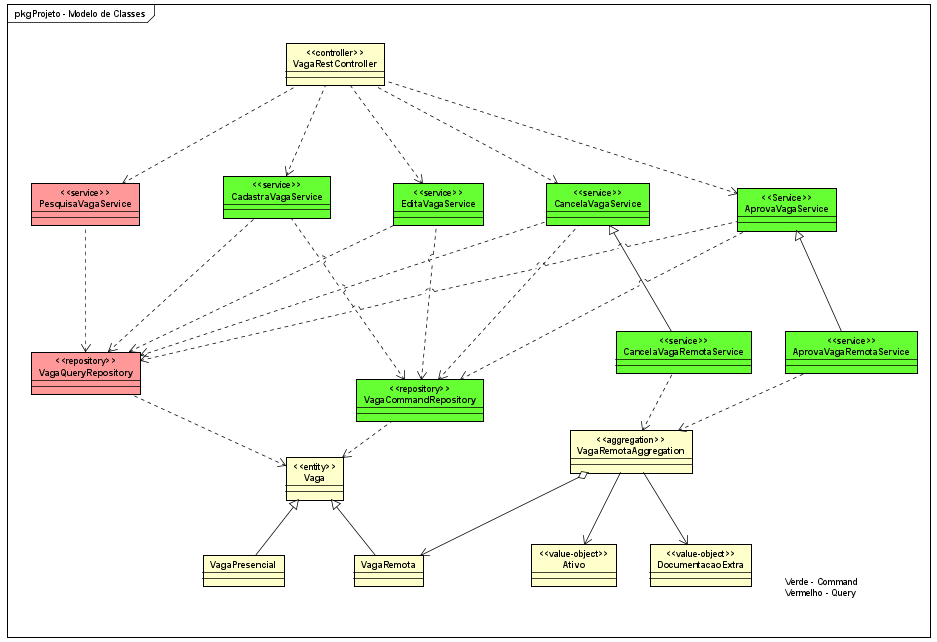
# Etapa de Projeto utilizando ***UML***, ***C4 Model***, ***Domain-Driven Design*** e outras práticas ágeis

## *UML*

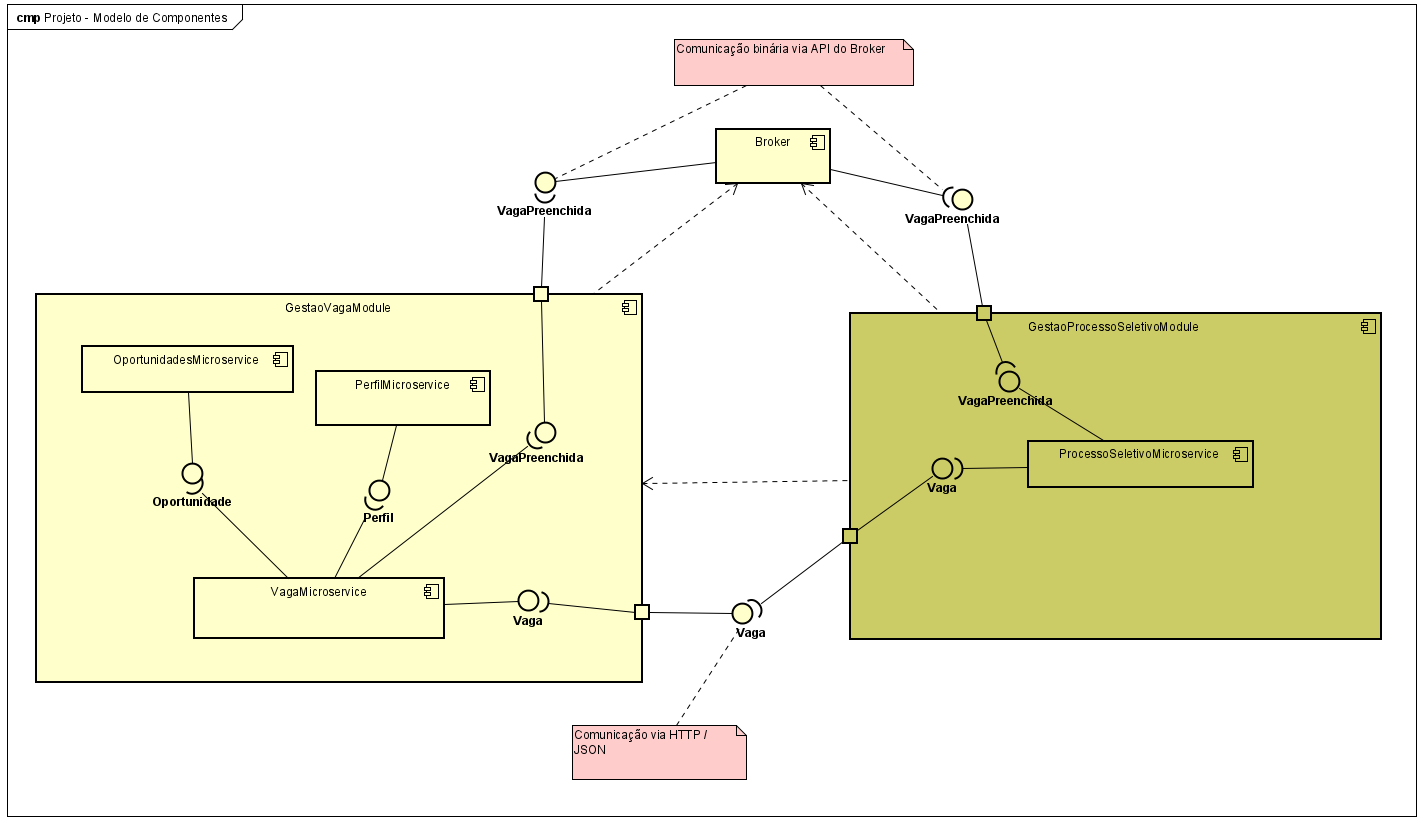
### Classes do PerfilMicroservice



### Classes do VagaMicroservice



### Comunicação entre Microsserviços

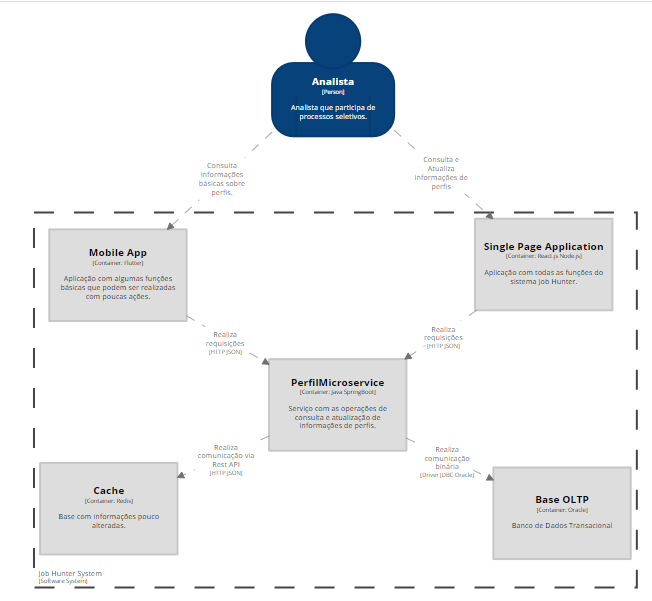


## *C4 Model*

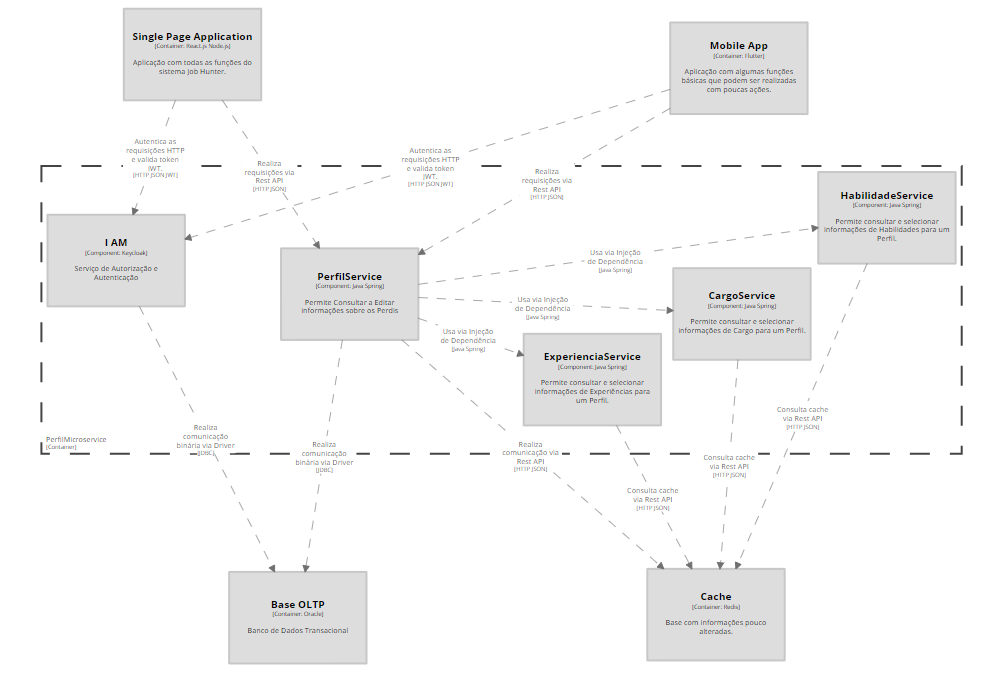
Foi utilizado o modelo C4 para visualização dos 4 aspectos principais de um sistema e seus componentes (incluindo seus Microsserviços).

### 1º Nível – Contexto

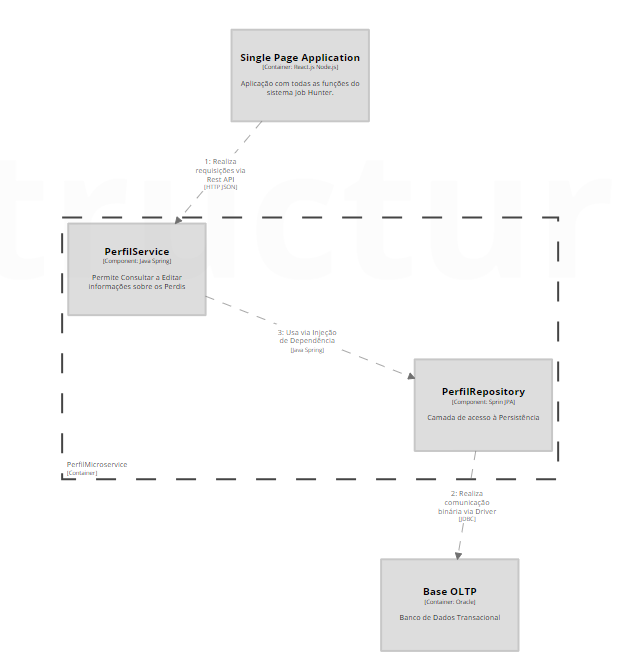
### 2º Nível – *Containers*



### 3º Nível – Componentes



### 4º Nível – Código



## Cartão CRC

|  |  |
| --- | --- |
| PesquisaPerfilService | |
| * Recuperar Perfis desejados pelos clientes * Listar Perfis para criação de Vagas | * Cliente * Vaga |

# Microsserviços Propostos

## Microsserviços necessários

Serão necessários os seguintes microsserviços para implementar o novo sistema:

* Módulo de Gestão de Clientes
  + MercadoMicroservice
  + ClienteMicroservice
* Módulo de Gestão de Vagas
  + PerfilMicroservice
  + OportunidadeMicroservice
  + VagaMicroservice
* Módulo de Gestão de Processos Seletivos
  + ProcessoSeletivoMicroservice
  + CandidatoMicroservice
  + ContrataçãoMicroservice
* Módulo de Busca de Oportunidades
  + BuscadorOportunidadesMicroservice
* Módulo de Busca de Candidatos
  + BuscaCandidatosMicroservice

A representação dos Microsserviços no processo de análise reflete-se nos diagramas de Casos de Uso por módulo. Para todo Caso de Uso do tipo *include* será criado um Microsserviço. Já para os Casos de Uso do tipo *extend* entende-se que, ou estão contemplados como endpoints do Microsserviço principal ou são implementações de estratégias (*Strategy pattern - GoF*) para a estrutura principal de uma determinada rotina (no caso, os buscadores).

## *Roadmap*

Com foco em *ROI\** e análise das pessoas que já foram integradas à equipe para execução do projeto piloto (com possível contratação) foi pensado o seguinte *Roadmap*:

Módulo de Gestão de Vagas >> Módulo de Gestão de Processos Seletivos >> Módulo de Busca de Candidatos >> Módulo de Busca de Oportunidades >> Módulo de Gestão de Clientes.

Todas as informações necessárias e já existentes em alguma outra fonte deverão carregadas no novo banco ou lidas, quando for a melhor solução de momento, nos bancos legados.

\* *Return of Investment*.

# Microsserviços Implementados

## Justificativa dos Serviços Implementados

O microsserviço selecionado para implementação foi o *PerfilMicroservice* do Módulo de Gestão de Vagas que, apesar de simples, permitem representar claramente a proposta de solução arquitetural, considerando todo o código base que guia o desenvolvimento dentro dos padrões definidos.

### Código

O código implementado para validar a proposta desse trabalho encontra-se no GitHub do autor: <https://github.com/cirilojunior/infnet-mit-arquitetura-bloco-microservicos>.

O repositório conta com a toda a documentação do projeto, código para geração de Diagramas no padrão *C4 Model* e código do(s) microsserviço(s) implementado(s).

### Arquitetura e Design do Código

Os detalhes técnicos foram pensados levando-se em consideração os padrões mais modernos de desenvolvimento.

Para facilitar e criar um nível de padronização coerente, o desenvolvimento de todos os módulos será baseado em uma pequena *lib* corporativa que entrega códigos criados recorrentemente. Ela possui em sua implementação os conceitos abaixo abordados.

#### CQRS

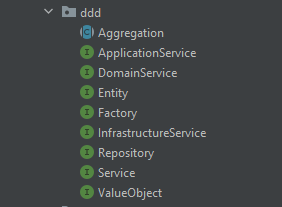
Baseado na proposta do CQS implementado na linguagem Eiffel.

Utilização no *design* das classes e nos *endpoints* dos microsserviços.



#### Domain-Driven Design

Utilização dos *building blocks* do *DDD* para separação de responsabilidade das classes influenciando todo o design. As classes herdam ou implementam\* abstrações que marcam sua responsabilidade influenciando em comportamentos e na própria organização e documentação do código.



\* Outra opção seria a criação de Anotações.

#### S.O.L.I.D.

Conjunto de princípios que guia o *design* do código buscando alta qualidade.

Um exemplo de utilização é a inversão de dependência entre as camadas da aplicação que, uma vez utilizando características de *DDD*, temos a injeção de implementações de repositórios nas abstrações definidas no domínio e a segregação de interfaces de consulta e alteração do *CQRS*.

#### Tiny Types

Estilo de codificação buscando principalmente expressividade, mas também reaproveitamento de código, como por exemplo: comportamentos e validações comuns e normalmente repetidas pela base de código.



### Apresentação (Consumo dos Microsserviços)

### Implantação (*deploy*)

Será utilizada a plataforma *Openshift* para *deploy* das aplicações, inicialmente *on-premise* na versão *Community*. Após a finalização de todos os módulos inicialmente planejados, existe um plano de contratação do serviço em um dos planos de nuvem\*. A migração das aplicações nesse contexto fica facilitada e reduz o *vendor lock-in* de provedores de nuvem. A dependência do próprio *Openshift* se justifica por todas as facilidades que ele oferece sobre a administração e gestão das aplicações por uma equipe com cultura *DevOps\*\**.

\* nas referências existe um *link* para as modalidades utilizando AWS, Google, Azure ou IBM.

\*\* A plataforma simplifica bastante todo o processo de *Continuous Integration*, *Continuous Delivery* através de um modelo de *build pipeline* obtido diretamente do repositório de código e criada através um catálogo opções de *containers* previamente configurados.

### Comunicação

A comunicação assíncrona entre os microsserviços se dará pelo *broker* padrão oferecido pela plataforma *Openshift*: *JBossMQ* (baseado no Apache Artemis).

Num primeiro momento, conforme já informado, algumas informações serão obtidas das bases corporativas atuais e na medida em que o novo sistema for sendo incrementado, essa comunicação será desligada.

Não será feita a implementação, apenas representação do componente arquitetural.

### I AM (autenticação e autorização)

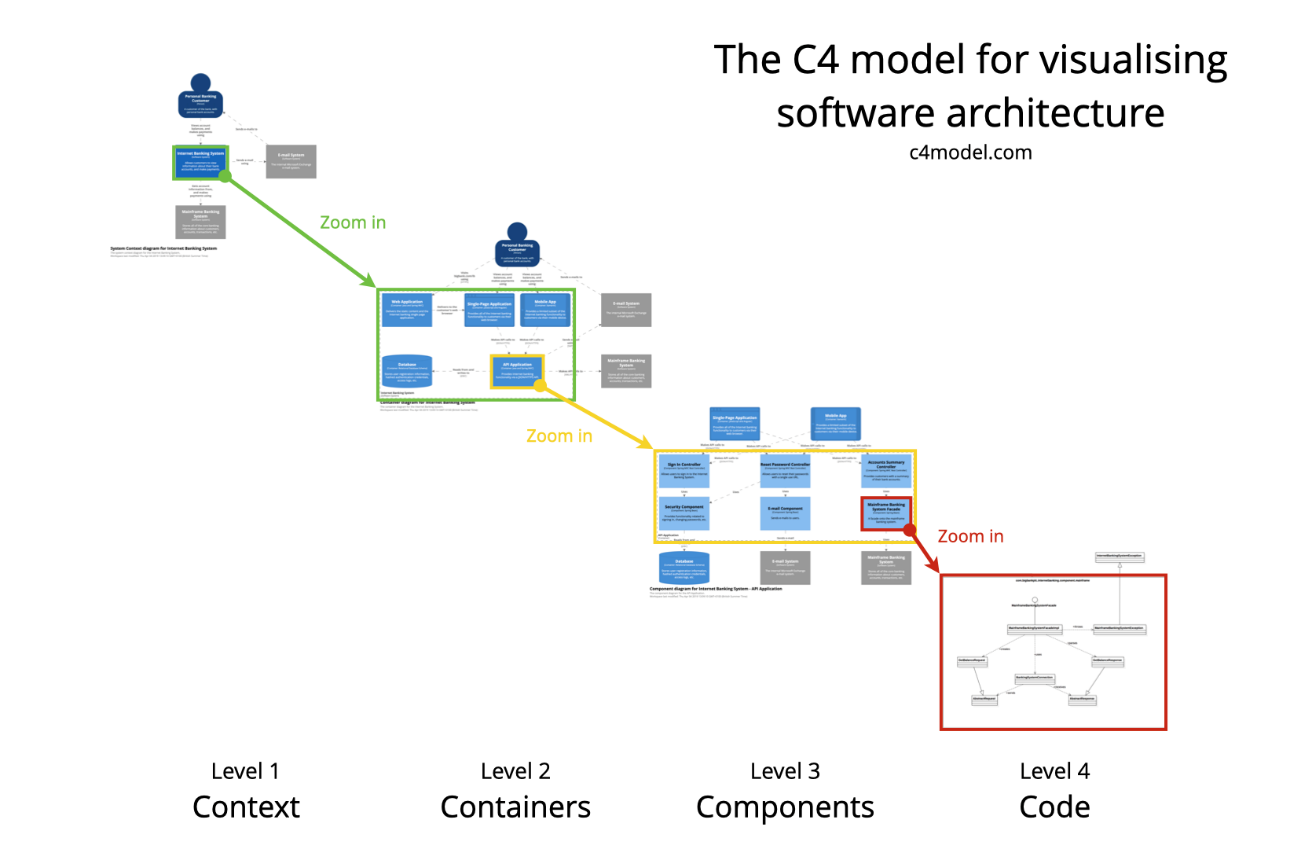
Como solução de *Identity Server*, foi selecionado o produto Keycloak também fornecido pela plataforma *Openshift*.

Não será feita a implementação, apenas representação do componente arquitetural.

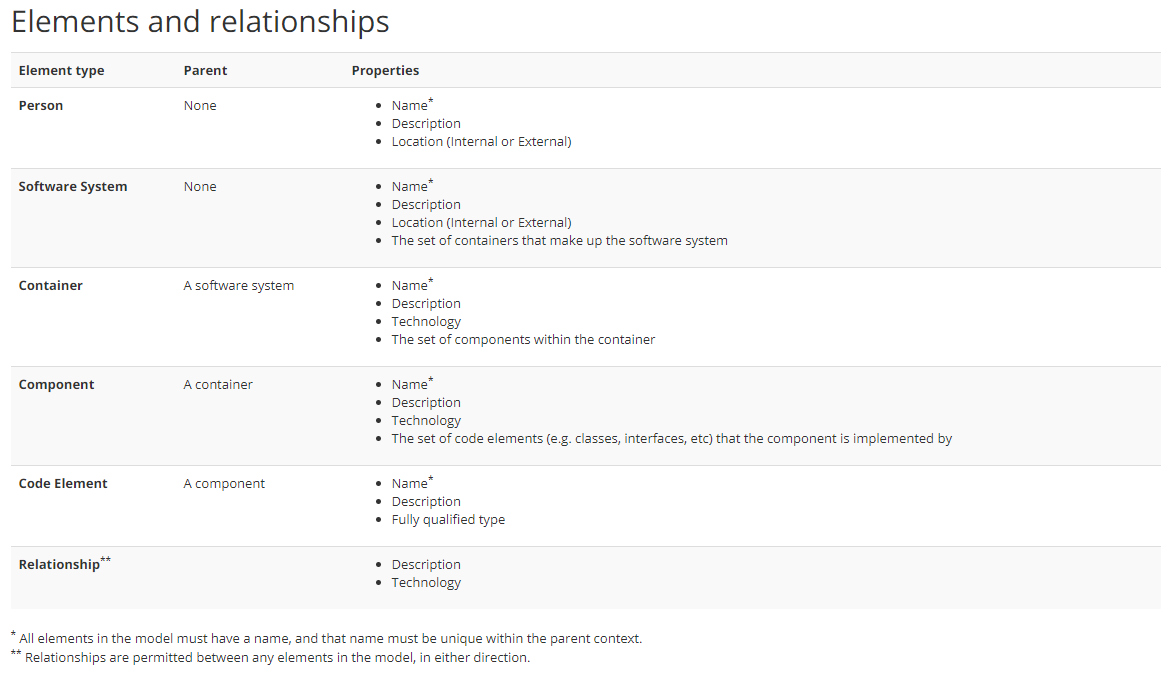
# Conclusão

# Referências

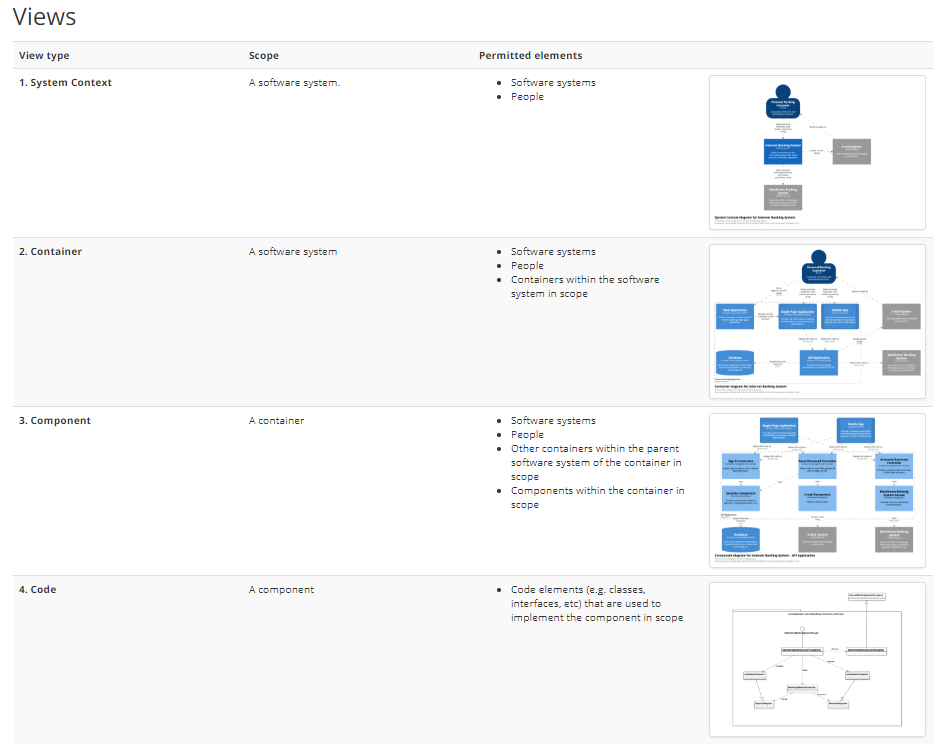
* Modelagem de *Software* com *RUP* e *UML*.
  + RUP
    - Entendimento das fases de um processo de desenvolvimento de *software*.
    - Fonte: <https://www.devmedia.com.br/rup-rational-unified-process/4574>
  + Documento de Visão e *Problem Statement*:
    - <https://www.ibm.com/support/knowledgecenter/en/SSWMEQ_4.0.6/com.ibm.rational.rrm.help.doc/topics/r_vision_doc.html>
  + UML as Sketch
    - Uso de *UML* de forma menos burocrática e mais produtiva com foco na solução e não na documentação em si.
    - Fontes:
      * <https://martinfowler.com/bliki/UmlMode.html>
      * <https://martinfowler.com/bliki/UmlAsSketch.html>
* *C4 Model*
  + Site oficial: <https://c4model.com/> (fonte das imagens)
    - Resumo
      * Visualização do Modelo:



* + - * *Metamodel*:



* + - * *View*:



* + *Structurizr*
    - <https://structurizr.com/>
    - Ferramenta para a criação de modelos no padrão *C4 Model*
* *GoF Patterns*
  + <https://www.devmedia.com.br/design-patterns-padroes-gof/16781>
  + Utilizado o *pattern Strategy* como guia para a solução dos buscadores.
* I AM
  + <https://www.keycloak.org/>
* Comunicação assíncrona entre serviços
  + <https://www.redhat.com/pt-br/technologies/jboss-middleware/amq>
* Openshift
  + <https://www.openshift.com/>
  + <https://www.openshift.com/products>
  + Plataforma de *containers* de aplicação *opensource* da [*Red Hat*](https://www.redhat.com/en).
* Lombok
  + <https://projectlombok.org/>
  + *Lib* utilizada na geração de código comum que permite que o fonte fique menos poluído.
* *CQS*
  + <https://martinfowler.com/bliki/CommandQuerySeparation.html>
  + Padrão de *design* que define a separação de métodos de Consulta e Alteração de forma clara.
* *CQRS*
  + <https://martinfowler.com/bliki/CQRS.html>
  + Padrão arquitetural baseado no *CQS*, mas agora pensando na questão arquitetural, ou seja, relacionado à requisições via rede.
* *S.O.L.I.D*
  + <https://blog.caelum.com.br/principios-do-codigo-solido-na-orientacao-a-objetos/>
  + Acrônimo de características desejadas para que o código obtenha um bom padrão de qualidade, sendo mais fácil de passar por evoluções.
* *Tiny Types*
  + <https://blog.caelum.com.br/pequenos-objetos-imutaveis-e-tiny-types/>
  + Estilos de codificação que visa expressividade e onde tentar-se criar os tipos o mais próximo possível do conceito representado, através de classes *wrappers* para os tipos comuns da linguagem, como por exemplo: String e List.
* *CRC Cards*
  + <http://agilemodeling.com/artifacts/crcModel.htm>
  + Estilos de documentação de classes (*Class*, *Resposabilities* e *Collaborators*).