**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**PUC Minas Virtual**

**Pós-graduação *Lato Sensu* em Arquitetura de *Software* Distribuído**

Projeto Integrado

Relatório Técnico (Parcial Etapa 1)

Software de Gestão – Módulo de Gestão Financeira

José Ricardo Serathiuk da Silveira

Belo Horizonte

fevereiro/2022

# Projeto Integrado – Arquitetura de Software Distribuído

**Sumário**

[Projeto Integrado – Arquitetura de Software Distribuído 2](#_Toc99198546)

[1. Introdução 3](#_Toc99198547)

[2. Cronograma do Trabalho 5](#_Toc99198548)

[3. Especificação Arquitetural da solução 6](#_Toc99198549)

[3.1 Restrições Arquiteturais 6](#_Toc99198550)

[3.2 Requisitos Funcionais 6](#_Toc99198551)

[3.3 Requisitos Não-funcionais 8](#_Toc99198552)

[3.4 Mecanismos Arquiteturais 8](#_Toc99198553)

[4. Modelagem Arquitetural 10](#_Toc99198554)

[4.1 Diagrama de Contexto 10](#_Toc99198555)

[Referências 12](#_Toc99198556)

## Introdução

O objetivo geral deste projeto é apresentar a descrição do projeto de arquitetura distribuída para um software de gestão (ERP). Essa solução será baseada em um software existente que utiliza uma arquitetura monolítica e vendido no modelo de SaaS (Software as a Service). Esse projeto irá propor um protótipo e versão inicial de um módulo de um sistema ERP utilizando uma arquitetura distribuída. Foi escolhido o módulo de Gestão Financeira, por ser um dos núcleos da gestão das empresas clientes.

Como esse projeto esperasse validar uma arquitetura para a modernização de um software ERP que é vendido como serviço e está no mercado há mais de 15 anos, atendendo os segmentos de mineradoras, concreteiras e construção pesada. Esse ERP monolito foi desenvolvido pensando em uma infraestrutura dedicada de servidores, não se aproveitando de todas as vantagens que Cloud Computing trouxe por exemplo. E a empresa que o desenvolveu está crescendo na faixa de 30% ao ano, e com isso cada vez mais está se tornando um problema escalar comercialmente o ERP atual e em manter o software e mudanças como um todo.

Uma das necessidades atuais é começar a separar a equipe de desenvolvimento em times por módulo e contexto de negócios, para especializar cada vez o desenvolvimento. Citando a Lei de Conway (WIKIPEDIA, 2022) que diz que “organizações que desenvolvem sistemas tendem a produzir sistemas que são cópias de suas estruturas de comunicação’, entendemos que essa mudança cultural dentro da empresa irá afetar a forma com que o software será desenvolvido. Cada time de cada módulo deverá ter que ter certa autonomia e independência, para que atenda de melhor forma o contexto de negócio que está inserido.

Para isso, se espera-se começar a utilizar uma arquitetura distribuída, para lidar com a questão organizacional, e como consequência melhorar a qualidade do software como um todo, desde o desenvolvimento até a sua entrega e a sua melhor adaptação a plataformas de cloud computing.

Foi escolhido apenas 1 módulo, e de forma simplicada para validar a arquitetura inicial, pois hoje o ERP monolito possui 15 módulos, cada um com a média de 10-15 telas. Não seria viável para fim de validação neste trabalho desenvolver uma solução completa de ERP. Será utilizado o Strangler Fig pattern (FOWLER, 2004) para fazer uma migração gradual do monolito para um modelo distribuído.

O módulo escolhido, o de Gestão de Finanças, deverá possibilitar a empresa controlar as finanças, de forma que ele consiga consolidar várias contas bancárias em um único local, possibilitando um melhor acompanhamento de sua saúde financeira e conferir o fluxo de caixa dos gastos. Também irá possibilitar controlar as Contas a Pagar e Receber. O usuário poderá acompanhar e atualizar suas informações financeiras via navegador de internet e via smartphone. Também possibilitará a integração com o ERP monolito (legado) e com os futuros módulos a serem migrados ou desenvolvidos. Por consequência também será migrado o módulo de autorização e autenticação, que será escolhida uma solução de mercado para tal tarefa.

Os objetivos específicos propostos são:

1. Definir e testar um padrão de autenticação e autorização unificado para ser utilizado por todos os serviços;
2. Validar a estrutura e estratégias de comunicação entre os serviços e possíveis problemas que se possa ter com eles, como por exemplo, definição de estratégias para comunicação, transações distribuídas, isolamento e consistência de dados entre serviços;
3. Analisar as complexidades da arquitetura distribuída e seus componentes e verificar se ela é uma alternativa viável para a arquitetura monolítica atual;

## Cronograma do Trabalho

A seguir é apresentado o cronograma proposto para as etapas deste trabalho.

|  |  |  |  |
| --- | --- | --- | --- |
| **Datas** | | **Atividade / Tarefa** | **Produto / Resultado** |
| **De** | **Até** |
| 08/02/2022 | 09/02/2022 | 1. Definição do Cronograma do Trabalho | Relatório / Cronograma do Trabalho |
| 10/02/2022 | 14/02/2022 | 2. Elaboração da Introdução do Relatório | Relatório / Introdução |
| 15/02/2022 | 16/02/2022 | 3. Definição das Restrições Arquiteturais | Relatório / Restrições Arquiteturais |
| 17/02/2022 | 18/02/2022 | 4. Definição dos Requisitos Funcionais | Relatório / Requisitos Funcionais |
| 19/02/2022 | 21/02/2022 | 5. Definição dos Requisitos Não Funcionais | Relatório / Requisitos Não Funcionais |
| 22/02/2022 | 23/02/2022 | 6. Mecanismos Arquiteturais | Relatório / Mecanismos Arquiteturais |
| 23/02/2022 | 23/02/2022 | 7. Diagrama de Contexto | Relatório / Diagrama de Contexto |
| 24/02/2022 | 25/03/2022 | 8. Revisão do Conteúdo da Etapa 1 | Resultado: Melhoria do texto/Conteudo |
| 26/03/2022 | 26/03/2022 | 9. Produção do Vídeo da Etapa 1 | Resultado: Vídeo da Etapa 1 |
| 28/03/2022 | 28/03/2022 | 10. Entrega do Relatório Parcial Etapa 1 | Resultado: Relatório Parcial da Etapa 1 |
| 29/03/2022 | 29/03/2022 | 11. Diagrama de Container | Relatório / Diagrama de Container |
| 30/03/2022 | 30/03/2022 | 12. Diagrama de Componentes | Relatorio / Diagrama de Componentes |
| 31/03/2022 | 06/04/2022 | 14. Wireframes da POC. | Wireframes da POC. |
| 07/04/2022 | 10/05/2022 | 15. Código Fonte da POC | Serviço POC: POC funcional |
| 11/05/2022 | 12/05/2022 | 16. Vídeo de Apresentação da POC | Resultado: Vídeo de Apresentação da POC |
| 13/05/2022 | 13/05/2022 | 17. Entrega do Conteúdo da Etapa 2 | Resultado: Entrega do Conteúdo da Etapa 2 |
| 15/05/2022 | 21/05/2022 | 18. Análise de Abordagens Arquiteturais | Relatório / Analíse de Abordagens Arquiteturais |
| 23/05/2022 | 28/05/2022 | 19. Cenários | Relatório / Cenários |
| 30/05/2022 | 04/06/2022 | 20. Evidências da Avaliação | Relatório / Evidencias da Avaliação |
| 06/06/2022 | 13/06/2022 | 21. Resultados Obtidos | Relatório / Resultados Obtidos |
| 14/06/2022 | 20/06/2022 | 22. Avaliação Crítica dos Resultados | Relatório / Avaliação Crítica dos Resultados |
| 21/06/2022 | 27/06/2022 | 23. Conclusão | Relatório / Conclusão |
| 28/07/2022 | 29/07/2022 | 24. Produção do Vídeo da Etapa 03 | Resultado: Vídeo da Etapa 03 |
| 30/07/2022 | 30/07/2022 | 25. Entrega da Etapa 03/Entrega Final | Resultado: Entrega do Relatório Final/Etapa 3 |

Fonte: Elaborada pelo autor. 2022.

## Especificação Arquitetural da solução

Esta seção apresenta a especificação básica da arquitetura da solução a ser desenvolvida, incluindo diagramas, restrições e requisitos definidos pelo autor, tal que permitem visualizar a macroarquitetura da solução.

## Restrições Arquiteturais

As restrições arquiteturais foram definidas levando em conta o background técnico da equipe que trabalha hoje na versão legada do software ERP.

|  |  |
| --- | --- |
| **ID** | **Descrição** |
| RA01 | Os módulos deverão ser desenvolvidos utilizando a linguagem Java, na versão 11 utilizando a stack de tecnologias Spring Boot. |
| RA02 | As APIs devem seguir o padrão RESTFUL |
| RA03 | Deverá ser utilizado o banco de dados PostgreSQL |
| RA04 | Deverá ser utilizar o protocolo OAuth 2.0 para autorização de acesso dos serviços. |

Fonte: Elaborada pelo autor. 2022.

## Requisitos Funcionais

Os requisitos funcionais abaixo foram baseados nas operações básicas do módulo de Gestão de Finanças, o suficiente para a validação inicial da arquitetura.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Descrição** | **Prioridade**  **B/M/A** | **Dificuldade**  **B/M/A** |
| RF01 | O usuário poderá manter o Plano de Contas de Fluxo de Caixa da empresa. | A | M |
| RF02 | O usuário poderá manter as contas bancárias da empresa (contas correntes, contas poupança e contas de caixa). A conta deverá ter uma descrição, tipo (corrente, poupança ou caixa) e a qual filial ela pertence. | A | M |
| RF03 | O usuário poderá manter lançamentos financeiros nas contas. O usuário poderá informar a descrição, data, hora, tipo de movimentação (débito ou crédito), valor, a Pessoa de origem/destino do movimento, Filial de Movimento e Conta de Fluxo de Caixa. | A | M |
| RF04 | O usuário poderá efetuar transferências entre contas bancárias. Nela ele irá escolher uma conta origem, uma conta destino, a data e valor a ser transferido. | B | A |
| RF05 | O usuário poderá anexar arquivos aos seus lançamentos, possibilitando guardar comprovantes, NF’s e afins no sistema. | B | M |
| RF06 | O usuário poderá manter os Pagar e Receber da empresa. O usuário deverá informar a Filial do Movimento, a Filial de Cobrança, a Conta de Fluxo de Caixa, o Tipo (Pagar ou Receber), o Fornecedor/Cliente, a Data de Emissão, Data de Vencimento e Valor. | A | A |
| RF07 | O sistema irá manter Lançamentos Financeiros para Baixas de Pagar e Receber efetuadas no sistema. Para cada Baixa, será gerado um Lançamento Financeiro. Se removida ou alterada a Baixa, o sistema deverá atualizar o Lançamento Financeiro gerado. | A | A |
| RF08 | O usuário poderá manter baixas (integrais ou totais) de Pagar e Receber. | M | M |
| RF09 | O sistema irá importar os lançamentos financeiro de conciliação bancária (extrato bancário) via integração bancária, para contas configuradas. | M | M |
| RF10 | O usuário poderá enviar para ordens de Pagamento a Fornecedores ou transferências, via integração bancária para contas configuradas. A Ordem de Pagamento/Transferência irá gerar um Pagar ou Receber. | M | A |
| RF11 | O sistema fará baixas automáticas de Ordens de Pagamento a Fornecedores efetuadas. | M | M |
| RF12 | O sistema irá importar os boletos a pagar via integração bancária DDA - Débito Direto Autorizado. O sistema irá gerar um Pagar ou Receber para boletos a pagar | M | M |
| RF13 | O usuário poderá gerar um relatório em XSLX Pendências financeiras. Nele o usuário poderá exibir informações de valores a pagar ou a receber. | M | B |
| RF14 | O usuário poderá gerar um relatório em XLSX de pagamentos e recebimentos. Nele poderá gerar o que já foi pago ou recebido de Fornecedores/Clientes. | M | B |
| RF15 | O usuário final poderá exportar o extrato consolidado dos lançamentos financeiros para Excel (XSLX), utilizando filtros de data, contas bancárias e/ou fluxo de caixa e período. | M | B |
| RF16 | O usuário final poderá visualizar gráficos de evolução financeira por conta bancária e por fluxo de caixa, podendo filtrar por data, conta bancária e/ou fluxo de caixa. | B | B |
| RF17 | O serviço de Pagar e Receber deverá permitir via evento que outros módulos do ERP criem, alterer ou excluam títulos de Pagar e Receber. Também irá enviar eventos de cancelamento, estorno ou rejeições dos Pagar e Receber enviados. | A | A |
| RF18 | O sistema deverá disponibilizar um log de auditoria de Lançamentos Financeiros. | M | M |
| RF19 | O sistema deverá disponibilizar um log de auditoria de Pagar e Receber. | M | M |
| RF20 | O sistema deverá disponibilizar um log de auditoria de Baixas de Pagar e Receber. | M | M |

Fonte: Elaborada pelo autor. 2022.

\*B=Baixa, M=Média, A=Alta.

## Requisitos Não-funcionais

Os requisitos não-funcionais abaixo foram baseados nos requisitos não-funcionais existentes para o software ERP legado, com a adição de alguns requisitos relativos a softwares distribuídos em geral.

|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Prioridade**  **B/M/A** |
| RNF01 | O sistema deve ser apresentar disponibilidade de funcionamento mensal de 99%. | A |
| RNF02 | O tempo máximo aceitável de duração de qualquer requisição síncrona a API será de 5 segundos. | M |
| RNF03 | A interface frontend deverá ser compatível com os navegadores Google Chrome, Mozilla Firefox, Apple Safari, em versões com até 2 anos. | M |
| RNF04 | O sistema deverá atender as normas legais descritas na LGPD (Lei Geral de Proteção de Dados) | A |
| RNF05 | A API REST deverá ser pública e documentada para integrações externas. | M |
| RNF06 | Todos os logs internos deverão ser acompanhados com a data e hora do evento e identificador de origem do evento (mensagem, usuário, ip, etc) | M |
| RNF07 | O sistema deverá manter um registro dos eventos. | A |

Fonte. Elaborada pelo autor. 2022.

\*B=Baixa, M=Média, A=Alta.

## Mecanismos Arquiteturais

Abaixo uma visão geral mecânicos que compõem a arquitetura para os novos módulos/funcionalidades do ERP.

|  |  |  |
| --- | --- | --- |
| **Análise** | ***Design*** | **Implementação** |
| Persistência | ORM | JPA / Hibernate / Spring Data JPA |
| Persistência | Banco de Dados Relacional | Amazon RDS for PostgreSQL |
| Persistencia | Armazenamento de arquivos | Amazon S3 |
| Front end | Framework de interface de usuário | Vue |
| Front end | UI Library | PrimeVUE |
| Backend | Regras de Negócio | Java |
| Backend | Framework de Aplicação | Spring Boot |
| Backend | Autenticação e Autorização - Cliente | Spring Security |
| Autenticação e Autorização | Servidor de Autenticação e Autorização | Amazon Cognito |
| Integração | Eventos/Mensageria | AWS SQS/SNS |
| Integração | Service Discovery | Netflix Eureka |
| Integração | API de Integração | API REST usando JSON implementada com Spring MVC REST |
| Integração | API Gateway | Amazon API Gateway |
| Monitoramento | Framework de Log | Logback |
| Monitoramento | Visualização de logs | AWS Cloudwatch Logs |
| Monitoramento | Alarmes e Métricas | AWS Cloudwatch |
| Teste de Software | Framework de Testes Unitários - Backend | JUnit |
| Entrega de Software | Ferramenta de Integração contínua - Backend | AWS CodeBuild |
| Entrega de Software | Ferramenta de entrega contínua - Backed | AWS Codepipeline / AWS Elastic Beanstalk |
| Entrega de Software | Ferramenta de Build - Backend | Maven |
| Entrega de Software | Ferramenta de Integração continua e entrega continua – Frontend | AWS Amplify |
| Entrega de Software | Ferramenta de Build - Frontend | Vue CLI |
| Versionamento | Versionamento de Código da Aplicação | Git/Github |

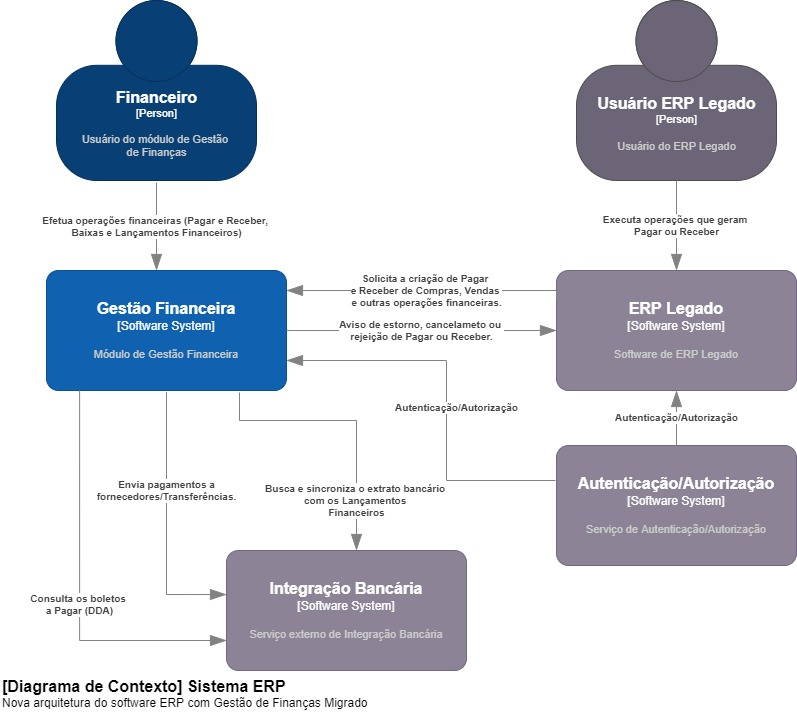
Fonte: Elaborada pelo autor. 2022.

## Modelagem Arquitetural

Esta seção apresenta a modelagem arquitetural da solução proposta, de forma a permitir seu completo entendimento visando à implementação da prova de conceito (PoC) do módulo de Gestão Financeira na seção 5.

Para esta modelagem arquitetural optou-se por utilizar o modelo C4 para documentação de arquitetura de software. Mais informações a respeito podem ser encontradas aqui: e aqui: <https://www.infoq.com/br/articles/C4-architecture-model/>. Dos quatro nível que compõem o modelo C4 três serão apresentados aqui e somente o Código será apresentado na próxima seção (5).

## 4.1 Diagrama de Contexto



**Figura 1 - Visão Geral da Solução de ERP com o módulo migrado.**

A figura 1 mostra a especificação o diagrama geral da solução proposta, com todos seus principais módulos, serviços e pessoas envolvidas. Nele é apresentado a interação entre os papéis e sistemas que compõem a solução completa deseja para o ERP e a interação entre eles.

# Referências

FOWLER, M. StranglerFigApplication. **Martin Fowler**, 2004. Disponivel em: <https://martinfowler.com/bliki/StranglerFigApplication.html>. Acesso em: 10 fev. 2022.

FOWLER, M. **Refactoring:** Improving the Design of Existing Code. Second Edition. ed. [S.l.]: Addison-Wesley Professional, 2018.

NEWMAN, S. **Building Microservices**. 2nd Edition. ed. [S.l.]: O'Reilly Media, Inc., 2021.

WIKIPEDIA. Conway's law. **Wikpedia**, 2022. Disponivel em: <https://en.wikipedia.org/wiki/Conway%27s\_law>. Acesso em: 10 fev. 2022.

YANAGA, E. **Migrating to Microservice Databases**. [S.l.]: O'Reilly Media, Inc., 2017.