

UNIVERSIDADE FEDERAL DE MINAS GERAIS  
Escola de Engenharia  
Colegiado do Curso de Graduação em Engenharia de Sistemas

Milton Pereira Bravo Neto

**APRIMORAMENTO DE PROCESSOS DE DEFINIÇÃO E VALIDAÇÃO DE  
REQUISITOS NO CICLO DE VIDA DE SISTEMAS LOW-CODE**

Belo Horizonte  
2025

Milton Pereira Bravo Neto

**APRIMORAMENTO DE PROCESSOS DE DEFINIÇÃO E VALIDAÇÃO DE  
REQUISITOS NO CICLO DE VIDA DE SISTEMAS LOW-CODE**

Trabalho de Conclusão de Curso apresentado ao  
Curso de Engenharia de Sistemas da Universi-  
dade Federal Minas Gerais, como requisito par-  
cial para o grau de bacharel (a) em Engenharia  
de Sistemas.

Orientador: Prof. Dr. André Costa Batista

Belo Horizonte  
2025

# Agradecimentos

Acima de tudo, agradeço a Deus. Sem Ele, não haveria vontade, propósito ou alegria nas coisas e nas atitudes. Este trabalho surgiu a partir desses três sentimentos e, por isso, sem Deus, nada aqui existiria.

Aos meus pais, à minha irmã e aos demais familiares, meu muito obrigado pela paciência e compreensão ao longo de toda esta jornada, que se encerra com este trabalho. Houve muitos momentos em que deixei de participar ou em que parecia distante, mas saibam que vocês sempre estiveram presentes nos meus pensamentos e no meu coração.

Gostaria de agradecer também a todos os professores que cruzaram meu caminho, ensinando, orientando e compartilhando suas vivências, que tanto me inspiraram e agregaram valor ao longo desses anos. Em especial, agradeço àqueles que ministraram as disciplinas específicas do curso de Engenharia de Sistemas, os quais, mesmo diante de tantos desafios, criaram um ecossistema de aprendizado que evidencia o valor e as aplicações do curso, trazendo oportunidades de atuação e ricas trocas de experiências.

Agradeço ao professor André Costa Batista, meu orientador neste trabalho, primeiramente por aceitar o desafio de orientar um projeto que, no início, até para mim era difícil de definir. Sou grato por sua contribuição e pela condução leve e transparente ao longo dessa jornada, bem como pelas conversas e trocas de experiências quase semanais, que sempre me motivaram a continuar melhorando.

“Nenhum homem está satisfeito com a própria sorte se fixar sua atenção na de outro”  
(Lúcio Aneu Sêneca)

# Resumo

Escreva aqui o resumo do seu trabalho.

**Palavras-chave:** palavra-chave 1; palavra-chave 2; palavra-chave 3.

# Abstract

Translate here the abstract of your work.

**Keywords:** keyword 1; keyword 2; keyword 3.

# Lista de Figuras

2.1	Ciclo de vida com estrutura em “V”. . . . .	18
2.2	Fases do estágio de Conceito, adaptado de (Kossiakoff et al., 2020) . . . . .	20
2.3	Fases do estágio de Desenvolvimento, adaptado de (Kossiakoff et al., 2020) . . . . .	22
2.4	Estágio de Pós Desenvolvimento, adaptado de (Kossiakoff et al., 2020) . . . . .	25
4.1	Fluxograma do processo que descreve o serviço prestado. . . . .	33
4.2	Relação com as fases do estágio de Conceito . . . . .	37
4.3	Relação com as fases do estágio de Desenvolvimento . . . . .	38
4.4	Relação com os estágios Pós-Desenvolvimento . . . . .	39
4.5	Destaque dos pontos falhos no ciclo de vida . . . . .	41
4.6	Padrão de arquitetura funcional e lógica relacionadas aos dados. . . . .	43
4.7	Padrão de arquitetura funcional e lógica relacionadas a documentos. . . . .	44
4.8	Padrão de arquitetura funcional e lógica relacionadas a processos e automações. . . . .	45
4.9	Padrão de arquitetura física dos sistemas desenvolvidos. . . . .	46
4.10	Arquitetura funcional do aplicativo proposto . . . . .	49
4.11	Arquitetura física do aplicativo proposto . . . . .	50
5.1	Arquitetura física da Solução X . . . . .	56
5.2	Arquitetura física da Solução Y . . . . .	57
5.3	Comparação agrupada de média e desvio padrão por categoria, antes e depois das mudanças . . . . .	58
5.4	Tela de visualização das soluções. . . . .	58
5.5	Tela de visualização das soluções com a expansão dos campos para adicionar nova solução. . . . .	59
5.6	Tela de visualização dos elementos. . . . .	59
5.7	Tela de visualização dos elementos com a expansão dos campos para adicionar novo elemento e editar dados da solução atual. . . . .	59
5.8	Tela de visualização das dependências. . . . .	60
5.9	Tela de visualização das dependências com a expansão dos campos para adicionar nova dependência e editar o elemento atual . . . . .	60
5.10	Tela de visualização evidenciando a dependência cruzada. . . . .	60
5.11	Dependências relacionadas ao Cenário 1 . . . . .	61

5.12	Dependências relacionadas ao Cenário 2 . . . . .	61
5.13	Dependências relacionadas ao Cenário 3 . . . . .	62
5.14	Comparativo da média e desvio padrão de esforço entre os três cenários, antes e depois da análise pelo aplicativo . . . . .	63



# Lista de Tabelas

2.1	Características das abordagens de um ciclo de vida . . . . .	20
2.2	Materialização do sistema, adaptado de (Kossiakoff et al., 2020) . . . . .	25
5.1	Distribuição de esforço por solução e categoria antes das mudanças sugeridas .	54
5.2	Média e desvio padrão dos pontos de esforço por tarefa, por categoria, antes de aplicar as mudanças sugeridas . . . . .	54
5.3	Distribuição de esforço por solução e categoria após as mudanças sugeridas . .	55
5.4	Média e variância dos pontos de esforço por categoria após aplicação das mudanças sugeridas . . . . .	55
5.5	Quantidade de tarefas geradas pelo agente de IA, a quantidade realmente utilizada e a Diferença percentual . . . . .	57
5.6	Comparativo do esforço alocado em três cenários, antes e depois do uso do aplicativo . . . . .	62

# Lista de Siglas e Símbolos

## Siglas

RPA	Robot Process Automation
RAG	Retrieval-Augmented Generation
ES	Engenharia de Sistemas
SoI	System of Interest
SoS	System of Systems
T&A	Testes e Avaliações
IA	Inteligência Artificial

# Sumário

<b>1</b>	<b>Introdução</b>	<b>13</b>
1.1	Objetivos Geral e Específicos . . . . .	15
1.2	Contribuições e Originalidade . . . . .	15
1.3	Organização do Trabalho . . . . .	15
<b>2</b>	<b>Revisão Bibliográfica</b>	<b>17</b>
2.1	Ciclo de Vida . . . . .	17
2.1.1	Fases dos estágios de Conceito e Desenvolvimento . . . . .	20
2.1.2	Estágios de Pós Desenvolvimento . . . . .	25
2.2	Arquitetura do Sistema . . . . .	26
2.3	Trabalhos relacionados . . . . .	27
2.3.1	Low Code Development Cycle Investigation . . . . .	27
2.3.2	Exploring Low-Code Development: A Comprehensive Literature Review	28
<b>3</b>	<b>Aspectos Socioeconômicos e Humanidades</b>	<b>30</b>
<b>4</b>	<b>Metodologia</b>	<b>32</b>
4.1	Ciclo de Vida . . . . .	32
4.1.1	Adequação de representação . . . . .	37
4.1.2	Avaliação do ciclo de vida . . . . .	39
4.2	Proposta de modificações . . . . .	42
4.2.1	Arquitetura do Sistema . . . . .	42
4.2.2	Rastreabilidade . . . . .	48
4.3	Procedimentos para Coleta de Resultados . . . . .	49
4.3.1	Comparação com Dados Históricos . . . . .	49
4.3.2	Avaliação de cenários de manutenção . . . . .	50
4.4	Instrumentos e Materiais . . . . .	51
<b>5</b>	<b>Resultados</b>	<b>52</b>
5.1	Resultados para a criação de novas histórias de usuário . . . . .	52
5.2	Resultados para Rastreabilidade . . . . .	57

5.3	Consolidação dos Resultados . . . . .	62
<b>6</b>	<b>Conclusão</b>	<b>64</b>
	<b>Referências Bibliográficas</b>	<b>66</b>

# Capítulo 1

## Introdução

Ao longo dos anos, vivenciando a experiência no mercado de trabalho em diferentes empresas, é possível notar a dificuldade em mapear, seguir e otimizar processos nas organizações. Um fator agravante é quando o processo se propõe não somente a definir operações administrativas, mas também a estabelecer um modelo de serviço prestado, como consultorias, treinamentos e criação de aplicativos. Além dos processos para a concepção do produto, a prestação do serviço possui processos próprios que afetam o relacionamento com as partes interessadas, bem como o resultado final a ser entregue. O serviço em questão pode ser resumido no desenvolvimento de ferramentas ou soluções para estruturação, automação e/ou digitalização de processos dentro de organizações. Outro ponto que vale ser ressaltado é a necessidade de um vasto conhecimento em arquitetura de soluções, de ferramentas e técnicas de desenvolvimento para viabilização do serviço e sua manutenção.

Nota-se que o mercado cria a necessidade do serviço, que surge de forma orgânica e, na maioria dos casos, sem uma definição adequada. Um ciclo de vida do sistema de interesse, seja ele um produto ou um serviço, é essencial para padronizar as etapas. A documentação dessas etapas pode ser estabelecida de acordo com os padrões da Engenharia de Sistemas (ES).

Quando um ciclo de vida não é bem definido, deficiências operacionais nas integrações e nos relacionamentos entre as partes interessadas são comuns de serem observadas. Onde mais se destaca essa deficiência é na área de gerenciamento de requisitos e rastreabilidade do sistema. Quando um ciclo de vida não possui nenhuma documentação de requisitos ou rastreabilidade, isso pode afetar tanto as estimativas de esforço e tempo para os desenvolvimentos e possíveis mudanças quanto deixar uma grande incerteza de impacto em futuras manutenções. Ao resolver um problema ou instabilidade, podem ser gerados outros que só serão notados pelos usuários finais do sistema.

Alterações e inclusão de requisitos nas fases finais de desenvolvimento ou depois do sistema desenvolvido são muito mais custosas do que se estes tivessem sido refinados e definidos no início. E, sem a capacidade de analisar a rastreabilidade do sistema, como dito anteriormente, as estimativas têm pouco fundamento e refletem pouco a realidade. Logo, prazos são mal calculados, métricas são extraídas de maneira errônea e o gerenciamento da equipe fica

prejudicado.

Manter o ciclo de vida do sistema de interesse atualizado e alinhado à realidade ajuda a reduzir custos tanto na sua concepção quanto na sua operação. Isso ocorre de forma direta, otimizando os recursos utilizados por meio de processos e etapas bem definidas, e de forma indireta, diminuindo o retrabalho e o tempo gasto em atividades desnecessárias.

Tem sido uma tendência, em diversas empresas, a criação de times, setores ou áreas focadas na digitalização e automação de processos, rotinas ou atividades repetitivas na empresa. As áreas de negócios que são colocadas como clientes para esse serviço abrangem equipes como marketing, finanças, pagamentos, segurança da informação, tesouraria, infraestrutura de tecnologia, qualidade, recursos humanos e diversos outros times que desempenham atividades de escritório. Essas equipes podem ser alvo da iniciativa de digitalização e são consideradas os principais interessados, atuando como clientes do serviço prestado.

Este trabalho se propõe a estudar e analisar um sistema dentro de uma empresa multinacional, com diversos ramos de atuação e ativos geradores de receita. A partir do mapeamento do ciclo de vida atual, propostas de revisão e melhoria são investigadas e elaboradas para abordar as deficiências operacionais nas integrações e nos relacionamentos com as partes interessadas. Em seguida, será proposta uma ferramenta para garantir a rastreabilidade do sistema e o gerenciamento dos requisitos de forma mais integrada e dinâmica.

O ciclo de vida a ser analisado descreve um serviço interno dessa empresa, prestado pelo setor de digitalização. Esse serviço consiste no desenvolvimento de uma solução tecnológica que resolva algum problema ou automatize algum processo trazido pelo cliente. É feita uma análise do contexto e realizada uma proposta de resolução para o problema, e em seguida se inicia o desenvolvimento, caso isso seja decidido. Em alguns casos, é necessário ainda o suporte à solução desenvolvida, dependendo da complexidade e do volume de utilização.

O time de desenvolvimento trabalha numa estrutura de fábrica de aplicativos ou fábrica de software, com poucos desenvolvedores experientes e generalistas, para que estejam bem engajados com todas as possibilidades a serem exploradas. Esse serviço é repetido a cada nova solução desenvolvida para os times clientes, novos ou não. Diversas tecnologias são utilizadas durante o desenvolvimento, sendo definidas de acordo com a necessidade de cada situação. Entretanto, as duas principais tecnologias são as ferramentas *low code*<sup>1</sup> da empresa Microsoft, conhecidas como ferramentas da *Power Platform*, e códigos em Python para execução de RPAs (do inglês *Robot Process Automation*).

---

<sup>1</sup>Ferramentas low code são plataformas de desenvolvimento que permitem a criação de aplicativos e sistemas com pouca ou nenhuma necessidade de programação manual e infraestrutura dedicada. Elas utilizam interfaces visuais, como arrastar e soltar componentes, e configurações pré-definidas para simplificar o processo de desenvolvimento.

## 1.1 Objetivos Geral e Específicos

O objetivo geral deste trabalho é melhorar a prestação de serviços de software de curta duração por meio da reforma de processos dentro do seu ciclo de vida.

De forma específica, pretende-se, primeiramente, realizar o mapeamento da situação atual do serviço prestado e do produto entregue. Em seguida, busca-se identificar oportunidades de melhoria no cenário existente, com foco na eficiência e qualidade das entregas. Também é objetivo aprimorar a definição e validação do conceito aplicado ao processo em uso, tornando-o mais claro e confiável. Por fim, este trabalho visa ainda a elaboração de uma ferramenta que auxilie no gerenciamento de rastreabilidade e dependências ao longo do desenvolvimento dos serviços.

## 1.2 Contribuições e Originalidade

As contribuições deste trabalho situam-se na interseção entre a digitalização e otimização de processos e o uso de tecnologias *low code*, áreas que ainda carecem de padrões consolidados em termos de desenvolvimento, arquitetura e boas práticas. A proposta se destaca por aplicar técnicas de Engenharia de Sistemas (ES) em um cenário atípico para a disciplina, caracterizado por projetos de curta duração e execução simultânea.

Ao contrário do fluxo comum no desenvolvimento de software, que adota metodologias ágeis com múltiplas iterações e entregas incrementais, o processo analisado neste estudo aproxima-se de um modelo sequencial, similar ao tradicional modelo em cascata. Contudo, trata-se de uma versão extremamente condensada, com prazos máximos entre três e quatro meses e múltiplos projetos sendo conduzidos em paralelo, o que impõe desafios adicionais de rastreabilidade e validação de requisitos (ou conceitos).

Nesse contexto, os processos clássicos da Engenharia de Sistemas, como definição e validação de conceito, análise de requisitos, gerenciamento de interfaces e integração, são adaptados para atender às restrições de tempo e recursos, utilizando ferramentas de apoio como as da *Power Platform*.

A adaptação dos métodos de ES para ciclos curtos de desenvolvimento e sua aplicação prática com suporte de plataformas *low code* representa uma contribuição original deste trabalho, tanto do ponto de vista metodológico quanto tecnológico. Além disso, este estudo demonstra a viabilidade da Engenharia de Sistemas em ambientes com forte demanda por entregas rápidas, oferecendo uma abordagem estruturada que mantém a rastreabilidade, consistência e qualidade dos produtos entregues.

## 1.3 Organização do Trabalho

O restante deste trabalho está organizado da seguinte forma:

No Capítulo 2, é apresentada a revisão bibliográfica necessária para o entendimento do tema, abordando os principais conceitos e definições relacionados à Engenharia de Sistemas, com ênfase no ciclo de vida de um sistema e nas diferentes arquiteturas que podem ser utilizadas para documentá-lo.

No Capítulo 3, são discutidos os aspectos e impactos socioeconômicos relacionados ao trabalho desenvolvido, considerando como esses fatores influenciam o ambiente organizacional, os colaboradores e a sociedade ao redor.

O Capítulo 4 apresenta a metodologia empregada na realização do estudo, incluindo o levantamento do processo atual de prestação do serviço, a adequação às boas práticas da Engenharia de Sistemas, a identificação de problemas ao longo do ciclo de vida, as propostas de melhoria e a concepção da aplicação desenvolvida para gerenciar a rastreabilidade.

**Mais pra frente aqui você vai mencionar os capítulos de resultado e conclusão.**



## Capítulo 2

# Revisão Bibliográfica

### 2.1 Ciclo de Vida

A definição e criação de um ciclo de vida é uma das formas da Engenharia de Sistemas (ES) atuar no seu propósito de viabilizar o sucesso de um sistema ao mesmo tempo que otimiza a competição existente entre os objetivos das partes interessadas. Ao desmembrar o esforço total e definir os estágios, seus papéis, as novas características do sistema, os critérios de conclusão, os riscos envolvidos e, finalmente, tomar uma decisão, está-se criando o ciclo de vida. Esse processo organiza e estrutura as etapas necessárias para o desenvolvimento e a evolução do sistema de forma clara e eficiente.

Entre cada estágio definido, existem os chamados *decision gates* (portões de decisão, em tradução livre). Nesses pontos, é realizada uma análise do progresso e, como o nome sugere, uma decisão é tomada em relação ao desenvolvimento do sistema.

O ciclo de vida de um sistema é definido a partir de suas características e particularidades, de modo que seus estágios sejam inseridos para atender todas as suas necessidades. Os estágios podem aparecer mais de uma vez, serem executados sequencialmente ou paralelamente e serem inseridos a qualquer momento do ciclo de vida.

Em alguns casos, o Sistema de Interesse (SoI, do inglês *System of Interest*) faz parte de um Sistema de Sistemas (SoS, do inglês *System of Systems*). Nesse contexto, cada um possui seu próprio ciclo de vida. Geralmente, em um SoS, cada elemento do sistema tem seu ciclo de vida independente, e o ciclo de vida do SoS influencia diretamente o do SoI. Por isso, ao analisar o ciclo de vida do SoI, é essencial considerar a evolução e as interações com o SoS.

O ciclo de vida genérico trazido no INCOSE (2023) nos mostra os seis estágios básicos existentes numa estruturação em “V” que busca mostrar de forma visual a aparição desses estágios ao longo do tempo, realçando também o possível paralelismo entre eles. Os estágios são de: conceito, desenvolvimento, produção, utilização, suporte e descontinuação. Na figura 2.1 podemos ver uma representação do que foi mostrado no livro. A seguir cada estágio será abordado:

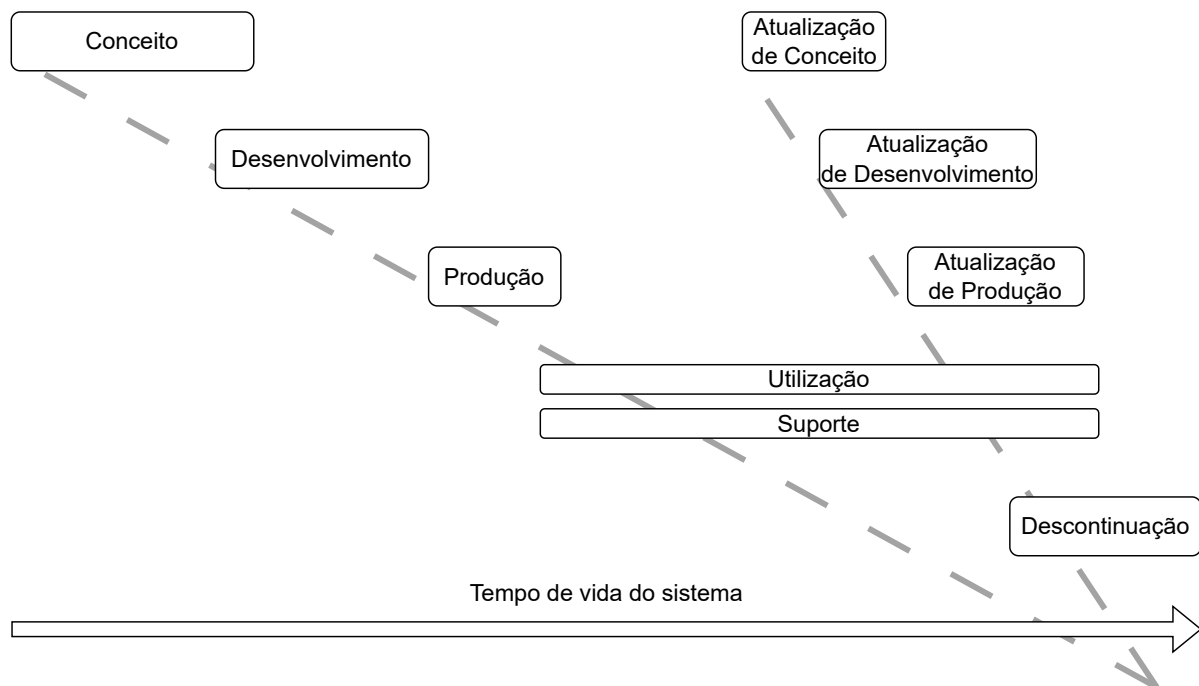


Figura 2.1: Ciclo de vida com estrutura em “V”.

- **Estágio de conceito:** Este estágio representa a fase exploratória, onde são identificadas as origens de uma necessidade, uma nova missão, uma nova capacidade de negócio ou a alteração de algum desses elementos. Nele, são analisados diversos fatores do sistema, como mercado, aspectos ambientais, condições econômicas, recursos disponíveis e escopo de atuação. O objetivo é definir os limites do problema a ser resolvido, as missões do sistema, onde ele será aplicado e realizar uma análise do negócio, da missão e dos valores que serão entregues. Para garantir uma definição clara do problema, são realizados levantamentos dos requisitos do sistema, das partes interessadas envolvidas e de suas necessidades, além da exploração do espaço de soluções possíveis. Com base nisso, é possível estimar um custo inicial do esforço necessário e criar uma agenda preliminar, que servirá como base para o ciclo de vida do sistema. Alguns dos resultados típicos desse estágio incluem documentos preliminares da arquitetura do sistema, análise de viabilidade, requisitos, design, agenda e estimativa de esforço. Esse estágio é crucial, pois é nele que o sistema é definido. Embora mudanças possam ocorrer posteriormente, sua implementação tende a ser mais complexa e custosa devido a fatores como tempo e recursos adicionais necessários.
- **Estágio de desenvolvimento:** nesse estágio é definido um SoI que atende e vai de encontro com as necessidades e requisitos das partes interessadas, e que pode ser produzido, utilizado, suportado e descontinuado caso necessário. O objetivo principal dessa fase é definir um projeto base de engenharia que pode ser executado, sem buscar a perfeição, mas atendendo às partes interessadas e respeitando os possíveis “trade-offs” previamente definidos nesse mesmo estágio. Nesse projeto base devem estar os requisitos, arquitetura,

modelagens, documentação e planejamento para próximas fases que também podem ser vistos como saídas dessa fazer.

- Estágio de produção: nesse estágio o projeto base definido no estágio de desenvolvimento, é aprovado e qualificado para ser colocado para utilização. Esse estágio simboliza o período de instalação, implementação ou transição para um ambiente de fato produtivo.
- Estágio de utilização: o início desse estágio se dá com a liberação do sistema ou parte dele para uso, incluindo os sistemas de apoio que são necessários para certas funcionalidades. Esse estágio comumente é o mais longo do ciclo de vida e é comum que mudanças e melhorias no SoI ocorram ao longo da utilização, lembrando sempre de fazer o gerenciamento dos riscos e documentação para garantir a integridade e manutenção do SoI.
- Estágio de suporte: segue paralelo ao estágio de utilização assim que alguma funcionalidade se torna disponível, no entanto o preparo e planejamento desse estágio pode ser iniciado antes como a aquisição de sobressalentes. Nesse estágio que são percebidas as melhorias e mudanças que podem vir a ser implementadas durante a utilização.
- Estágio de descontinuação: ocorre quando o sistema é retirado de operação, marcando o fim dos estágios de utilização e produção, ou, no máximo, havendo uma pequena sobreposição entre eles. Além de definir como será realizado o descarte ou armazenamento físico ou virtual dos componentes do sistema, essa etapa também envolve a análise da possível extensão da vida útil de algumas partes do sistema e o arquivamento de documentos importantes relacionados a ele. É uma fase crucial para garantir o encerramento adequado do ciclo de vida do sistema.

Ainda no INCOSE (2023), são trazidos conceitos importante sobre os *decision gates* que coexistem entre os estágio dos ciclo de vida, tanto no início quanto no fim de cada estágio. Dentre os objetivos dos *decision gates* estão o acompanhamento da evolução da maturidade do sistema, a conferência dos critérios de saída ou entrada de um estágio, a análise de risco mediante à situação atual do sistema, e por fim uma tomada de decisão sobre o que será feito. Podendo haver um regresso no ciclo de vida, um avanço, uma pausa ou até mesmo o cancelamento do projeto.

É importante saber equilibrar a formalidade e frequência desses eventos, visto que eles envolvem diferentes partes interessadas, gestores e especialistas, e além disso as decisões devem ser guiadas por dados tomados nos estágios do ciclo de vida e nos artefatos que são gerados para esse momento. Isso evita considerações desnecessárias e inadequadas que podem prejudicar futuramente.

As três abordagens principais trazidas pelo livro para os ciclos de vida são, sequencial, incremental e evolucionário. As principais características dessas três abordagens podem ser resumidas na tabela 2.1 apresentada.

Durante a execução dos estágios do ciclo de vida várias tarefas são executadas, e para isso alguns processos precisam ser realizados para garantir a consistência das atividades. Um

Tabela 2.1: Características das abordagens de um ciclo de vida

Abordagem	Requisitos definidos no início	Iterações planejadas	Múltiplas instalações
Sequencial	Todos os requisitos	Apenas uma	Não
Incremental	Todos os requisitos	Múltiplas	Potencialmente
Evolucionário	Parte dos requisitos	Múltiplas	Tipicamente

conjunto de processos é definido no livro e a execução de cada um deles varia de acordo com os estágios existentes no ciclo de vida do sistema.

### 2.1.1 Fases dos estágios de Conceito e Desenvolvimento

Kossiakoff et al. (2020) fazem uma quebra dos estágios de conceito e desenvolvimento citados anteriormente, subdividindo-os em três fases cada. Dessa maneira a análise do ciclo de vida de um sistemas, bem como a sua estruturação fica mais clara e objetiva. Para o estágio de conceito temos as fases mostradas na figura 2.2, Kossiakoff et al. (2020) as descreve da seguinte maneira:

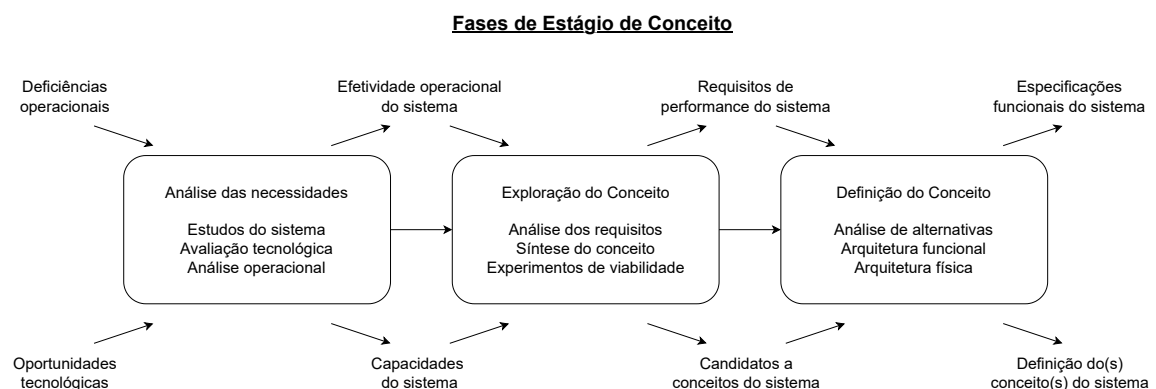


Figura 2.2: Fases do estágio de Conceito, adaptado de (Kossiakoff et al., 2020)

#### Fase de análise de necessidade

A fase de análise de necessidades tem como objetivo identificar se há, de fato, uma necessidade legítima para o desenvolvimento de um novo sistema, bem como avaliar se existe uma abordagem viável para atendê-la. Essa etapa exige uma análise crítica das limitações dos meios existentes em suprir as demandas atuais ou futuras previstas. Também é considerada a viabilidade técnica, ou seja, se a tecnologia disponível é capaz de suportar a capacidade adicional desejada.

Frequentemente, o início do ciclo de vida de um sistema não se dá de forma repentina, mas sim como resultado de uma análise contínua das necessidades operacionais ou do desenvolvimento de produtos inovadores. O principal resultado dessa fase é a descrição das capacidades

e da efetividade operacional esperadas do novo sistema. Embora ainda não configure um conjunto formal de requisitos, essa descrição serve como base para sua futura definição.

Para apoiar essa fase, são utilizados diversos métodos e ferramentas, especialmente as áreas da análise operacional e da pesquisa operacional. Além disso, avaliações tecnológicas e experimentações complementam essas abordagens matemáticas, contribuindo para uma definição mais precisa das necessidades do sistema.

### **Fase de exploração de conceito**

Na fase de exploração de conceitos, busca-se responder às seguintes questões: “Qual desempenho o novo sistema precisa atingir para satisfazer a necessidade identificada?” e “Existe pelo menos uma abordagem viável para alcançar esse desempenho a um custo aceitável?” A resposta positiva a essas perguntas é essencial para estabelecer objetivos realistas e viáveis antes de investir fortemente no desenvolvimento do sistema.

O principal resultado desta fase é a formulação do primeiro conjunto de requisitos de desempenho do sistema, considerados “oficiais” por serem passíveis de medição e avaliação por parte de contratantes ou agências. Além disso, são gerados os conceitos candidatos de sistema — ou seja, múltiplas alternativas de solução. A exploração de diferentes abordagens é fundamental para compreender as possibilidades disponíveis para atender à necessidade identificada.

Diversas ferramentas e técnicas são utilizadas nessa etapa, incluindo métodos de processo (como análise de requisitos) e julgamento especializado (como sessões de *brainstorming*). Inicialmente, o número de conceitos pode ser elevado, mas o processo visa reduzi-los rapidamente a um conjunto manejável. A viabilidade das alternativas finais precisa ser comprovada, pois elas serão a base para as decisões da próxima fase do ciclo de vida.

### **Fase de definição de conceito**

A fase de definição de conceito tem como foco selecionar o conceito de sistema mais promissor, ou seja, aquele que apresenta o melhor equilíbrio entre capacidade, vida útil operacional e custo. Para isso, diferentes alternativas devem ser comparadas em relação ao desempenho, utilidade operacional, riscos de desenvolvimento e custos. Com base nessa análise, é decidido se vale a pena investir recursos significativos no desenvolvimento do novo sistema.

O principal resultado dessa fase é uma descrição funcional clara do que o sistema deve fazer e qual desempenho deve alcançar, além da definição do conceito selecionado de sistema. Em sistemas simples, essa descrição pode ser feita de forma direta; em sistemas mais complexos, é necessária uma arquitetura de sistema mais detalhada, que represente o sistema sob diferentes perspectivas — principalmente funcional e física.

As ferramentas utilizadas nesse estágio incluem análise de alternativas e práticas de

arquitetura de sistemas. Em contextos comerciais, as fases anteriores são frequentemente integradas em um estudo de viabilidade, que serve como base para decidir se o conceito deve ser desenvolvido.

Mesmo que já tenham sido dedicados esforços significativos na compreensão do ambiente operacional e das tecnologias relacionadas, até este ponto o investimento direto no desenvolvimento do sistema costuma ser limitado. É nas fases seguintes que a maior parte dos recursos será aplicada, com foco no desenvolvimento técnico e implementação.

Olhando agora para o estágio de desenvolvimento vemos na figura 2.3 suas fases e entradas e saídas. Novamente suas definições segundo Kossiakoff et al. (2020) encontram-se abaixo:

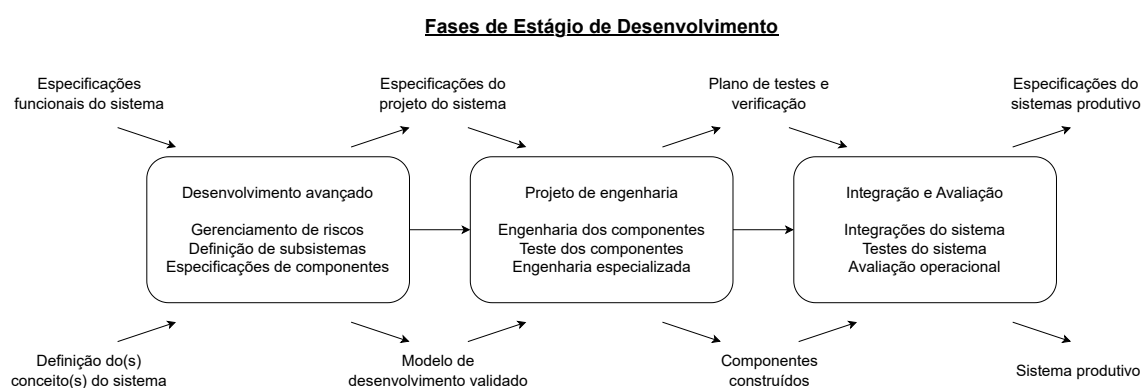


Figura 2.3: Fases do estágio de Desenvolvimento, adaptado de (Kossiakoff et al., 2020)

### Fase de desenvolvimento avançado

A fase de desenvolvimento avançado marca a transição entre a definição conceitual e o início do desenvolvimento técnico de um sistema. Seu sucesso depende fortemente da solidez das decisões tomadas nas fases conceituais anteriores. No entanto, como essas fases iniciais geralmente envolvem análises com recursos limitados, ainda restam incertezas importantes que precisam ser identificadas e resolvidas o quanto antes. Esta fase tem como principal objetivo minimizar esses riscos e transformar os requisitos funcionais do sistema em especificações técnicas mais detalhadas.

Duas finalidades centrais definem esta fase: a identificação e mitigação de riscos de desenvolvimento, e a produção das especificações de design do sistema. Isso é particularmente crítico em projetos que envolvem tecnologias inovadoras ou desempenho que extrapola os limites previamente testados. A fase concentra-se no desenvolvimento e validação das partes do sistema que ainda não estão consolidadas, assegurando que seus requisitos possam de fato ser atendidos.

Nesta etapa, a ES desempenha papel essencial ao definir o que precisa ser validado, como isso será feito e como interpretar os resultados obtidos. Frequentemente, modelos

experimentais e simulações são empregados para validar conceitos de projeto de componentes e subsistemas, reduzindo os custos de desenvolvimento.

O produto final desta fase é um modelo de desenvolvimento validado, junto a um conjunto refinado de especificações de projeto. Esse modelo deve demonstrar que o sistema pode ser projetado e fabricado de forma viável e com riscos aceitáveis. Por isso, todos os riscos precisam estar classificados como controláveis antes que o projeto avance para a próxima fase do ciclo de vida.

### **Fase de projeto de engenharia**

A fase de projeto de engenharia representa a transição entre a definição conceitual e a concretização técnica do sistema. É nesta etapa que o projeto detalhado dos componentes é desenvolvido, resultando na criação de um protótipo funcional ou virtual do sistema. Devido à sua complexidade e escopo, essa fase é geralmente marcada por revisões formais de projeto, que permitem ao cliente acompanhar o progresso, controlar custos, revisar o cronograma e fornecer feedback essencial aos desenvolvedores.

Embora aspectos como confiabilidade, manutenibilidade e fabricabilidade – conhecidos como “Engenharia especializada” – já tenham sido considerados anteriormente, nesta fase eles assumem um papel central. O engenheiro de sistemas tem a responsabilidade de garantir que cada componente implementa corretamente os requisitos funcionais e de compatibilidade, além de coordenar o processo de mudanças de engenharia para manter o controle das interfaces e da configuração do sistema.

As principais atividades dessa etapa incluem a conversão das especificações de componentes em projetos de engenharia completos, além da execução de testes preliminares. Esses testes podem ser realizados imediatamente após o projeto ou paralelamente a ele. Outro elemento crucial é o refinamento do plano de testes e avaliações (T&A), iniciado em fases anteriores, mas consolidado com base nas decisões e dados acumulados até aqui.

Os dois principais produtos desta fase são o plano de T&A e um protótipo do sistema. Este protótipo pode assumir diferentes formas — física, virtual ou híbrida — dependendo da natureza do sistema em questão. Por exemplo, no caso de sistemas complexos e de grande escala, como embarcações de carga, o protótipo pode combinar simulações digitais e modelos físicos em escala reduzida.

Ferramentas modernas de projeto assistido por computador (CAD) e simulações de sistemas são amplamente utilizadas para apoiar as decisões de engenharia, garantindo que o sistema projetado seja viável, testável e alinhado com os objetivos operacionais definidos nas fases iniciais do ciclo de vida.

## **Fase de integração e avaliação**

A fase de integração e avaliação marca o momento em que os componentes projetados e desenvolvidos são reunidos para formar um sistema funcional completo. Embora ainda seja parte do processo de desenvolvimento, essa etapa possui características distintas, especialmente no que diz respeito ao papel da ES, justificando seu tratamento como uma fase separada no ciclo de vida do sistema.

É nessa etapa que o sistema é montado e testado pela primeira vez em um ambiente realista ou simulado. Isso permite verificar se as interfaces entre os componentes estão compatíveis e se a integração geral atende aos requisitos funcionais estabelecidos. Apesar de testes prévios em subsistemas ou protótipos terem sido realizados, somente agora é possível validar a integridade do sistema como um todo.

Frequentemente, a execução dessa fase exige a construção de instalações específicas capazes de simular estímulos operacionais e restrições reais, de modo a permitir uma avaliação precisa do desempenho do sistema. A complexidade e os recursos necessários para essa infraestrutura não devem ser subestimados.

Os principais resultados da fase de integração e avaliação são: (i) as especificações finais de produção do sistema, conhecidas como “baseline de produção”, que orientam a fabricação do produto final, e (ii) o sistema de produção propriamente dito, que inclui tudo o que é necessário para manufaturar e montar o sistema em escala.

Ferramentas modernas de integração, T&A, bem como métodos e princípios atualizados, estão disponíveis para apoiar os engenheiros nesse processo. Antes que a produção em larga escala possa começar, é essencial que o sistema final seja verificado e validado em um ambiente operacional real ou em uma simulação que represente fielmente esse contexto.

Kossiakoff et al. (2020) trazem ainda um relacionamento entre as fases desses dois estágios com os níveis de detalhamento do sistema. De maneira visual é destacado em quais os níveis cada fase deve alcançar, consequentemente definindo o detalhamento de cada fase. Assim fica mais claro a materialização do sistema, onde as primeiras fases tocam na parte mais superficial e mais abstrata, e a medida que se avança de fase, vai se adentrando nos níveis de funcionalidades, quebrando e detalhando as funções e associando-as com os elementos do sistema no nível correspondente. A tabela 2.2 ilustra essas relações.

É fundamental perceber que, mesmo que o projeto detalhado só se conclua tardiamente no desenvolvimento, a visualização geral do sistema deve ocorrer ainda nas fases iniciais. Isso é necessário para estimar custos, viabilidade técnica e orientar a escolha do conceito do sistema. A visualização inicial não precisa ser precisa em todos os detalhes, mas deve ser realista o suficiente para orientar decisões viáveis.



Fase Nível	Estágio de Conceito			Estágio de Desenvolvimento		
	Análise das necessidades	Exploração do Conceito	Definição do Conceito	Desenvolvimento avançado	Projeto de engenharia	Integração e Avaliação
Sistema	Define as capacidades e efetividades do sistema	Identifica, explora e sintetiza conceitos	Define conceitos selecionados e especificações	Validação de conceito		Teste e avaliação
Subsistema		Define requisitos e garante viabilidade	Define arquitetura funcional e física	Validação de subsistemas		Integração e testes
Componente			Aloca funções a componentes	Define especificações	Projeto e testes	Integração e testes
Subcomponente				Aloca funções a subcomponentes	Projeto	
Peças					Fazer ou comprar	

Tabela 2.2: Materialização do sistema, adaptado de (Kossiakoff et al., 2020)

### 2.1.2 Estágios de Pós Desenvolvimento

Após a conclusão do desenvolvimento de um sistema, inicia-se a etapa pós-desenvolvimento, composta por duas fases principais, de acordo com (Kossiakoff et al., 2020): Produção e Utilização e suporte. Os autores unificaram Utilização e Suporte pois de fato seguem em paralelo a todo momento. Porém, será adicionado o último estágio do ciclo de vida apresentado pelo INCOSE (2023) a esse grupo, o estágio de Descontinuação. A figura 2.4 trás no mesmo estilo a representação desses estágios.

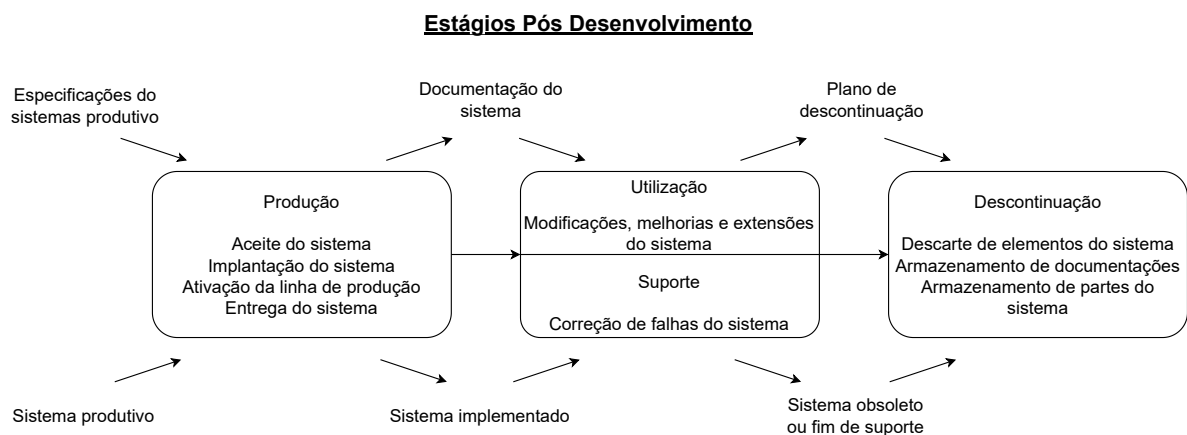


Figura 2.4: Estágio de Pós Desenvolvimento, adaptado de (Kossiakoff et al., 2020)

## 2.2 Arquitetura do Sistema

Como mencionado na seção 2.1 existem diferentes etapas durante o ciclo de vida de um sistema. Uma delas é o “Processo de Definição da Arquitetura do Sistema”, que tem como objetivo, conforme registrado no INCOSE (2023), gerar alternativas de arquiteturas do sistema, selecionar uma ou mais alternativa que atende aos requisitos das partes interessadas, e expressar isso em uma visualização e modelos consistentes.

Dessa forma, esse processo provê informação e dados necessários para identificar e caracterizar os conceitos e propriedades fundamentais do sistema e seus elementos. Esse processo é vivo ao longo do ciclo de vida, e deve ser sempre retomado quando há modificações de conceitos ou de perspectivas no decorrer do desenvolvimento do sistema.

Dentre as entradas para a execução desse processo mostradas no INCOSE (2023), destacam-se os “Requisitos do Sistema”, as “Restrições de solução”, a “Descrição do projeto do sistema”, “As soluções alternativas” e um “Projeto de arquitetura validado e verificado”.

Com essas entradas em mãos são realizadas as atividades para a definição de qual será a arquitetura definitiva do sistema. E após isso temos como saídas resultantes típicas os “Artefatos e registros da arquitetura do sistema definida”, a “Descrição da arquitetura do sistema”, o “Mapeamento de rastreabilidade” e a “Justificativa da arquitetura do sistema”.

A existência de um “Estilo de Arquitetura” é de extrema importância para que esse processo seja executado com êxito. Ele atua como um modelo, ou guia, para se construir a arquitetura do sistema. Os “Estilos de Arquitetura” podem ser definidos com base no ponto de vista da arquitetura, nos elementos do sistema e seus relacionamentos, nas conexões, interfaces, mecanismos de interação e possíveis restrições.

Além dos “Estilos de Arquitetura”, outro conceito importante é o de “Padrões de Arquitetura”. Eles são modelos simplificados mas completos no que diz respeito aos elementos do sistema e são reutilizáveis para diferentes tipos cenários. O uso de “Padrões de Arquitetura” agiliza a documentação, facilita a comunicação, promove o reúso, melhora a produtividade e eficiência e serve como um ponto de início para o desenvolvimento de novos sistemas.

Como o conceito de arquitetura pode ser muito abrangente, o SEBoK Editorial Board (2024) mostra três segmentações, a arquitetura funcional, lógica e a física.

A arquitetura funcional compreende as funcionalidades do sistema, ou seja, quais funções ou comportamentos aquele sistema executa ou possui em diferentes contextos para atingir os objetivos esperados pelos diferentes interessados. Essa arquitetura está fortemente conectada com a definição de conceito do sistema e tem papel chave em dar início à materialização do sistema ao traduzir esses conceitos em um projeto de sistema viável. Não há exatamente um modelo ou padrão recomendado de arquitetura funcional, em casos de sistemas com poucas funcionalidades ou menos complexos, um texto descritivo já seria suficiente, em outros casos, um diagrama hierárquico de funcionalidades pode ser mais interessante. Essa arquitetura é construída já no primeiro estágio do ciclo de vida do sistema.

O desenvolvimento da arquitetura lógica tem como propósito definir, sintetizar e documentar a lógica por trás do sistema a ser desenvolvido resultando em uma modelagem que poderá ser utilizada, no futuro, para verificar e validar os requisitos do sistema em todos os cenários operacionais. Essa arquitetura pode incluir mais de um artefato, como um modelo de arquitetura funcional decomposta em hierarquia de funções e subfunções, um modelo de arquitetura comportamental com diagramas de atividades, de estados, de fluxos de dados ou de blocos funcionais do sistema, e ainda um modelo de arquitetura temporal do sistema, com as funções do sistema classificadas de acordo com a frequência de execução, podendo incluir também os aspectos síncronos e assíncronos do sistema.

Já a na definição da arquitetura física há o propósito de elaborar modelos e visualizações concretas de soluções que acomodem a arquitetura lógica e que atendam ao requisitos do sistema conforme acordado. A arquitetura física é uma organização dos elementos do sistema que compõem a solução, seja ela um produto, um serviço ou até mesmo um empresa. Como existem diferentes tipos de sistema, o elemento do sistema pode assumir diferentes características ou interfaces, em produtos eles podem ser de fato componentes físicos mecânicos ou eletrônicos, podem ser softwares específicos e papeis de operação, já para serviços por exemplo eles podem ser bancos de dados, processos, papeis de operação, aplicações. Nesse momento é feita a associação dos elementos lógicos do sistema, derivados dos requisitos, aos elementos físicos do sistema gerando então o mapeamento de rastreabilidade. Mais de uma abordagem de arquitetura pode ser sugerida às partes interessadas, que devem analisar e escolher a que melhor atende. Essa modelagem pode ser feita utilizando diagramas de estrutura de blocos e layouts ou outros modelos que podem variar de acordo com o domínio que o sistema está envolvido.

## 2.3 Trabalhos relacionados

### 2.3.1 Low Code Development Cycle Investigation

O uso de *low code* nas organizações tem diferentes motivos, um deles é o empoderamento de seus funcionários como citado por Pańkowska (2024). Ao qualificar e incentivar os funcionários a utilizarem essas tecnologias para resolverem seus problemas, são criadas comunidades de *citizen developers*, que em tradução livre pode ser chamado de “desenvolvedores cidadãos”. Estes são usuários finais que, mesmo sem conhecimentos técnicos aprofundados em desenvolvimento de softwares, conseguem criar automações e aplicativos simples para si próprios, devido à baixa barreira de entrada para se adotar essas tecnologias. Dessa maneira, são economizados ou evitados investimentos em infraestrutura e recursos humanos para se manter e criar soluções que utilizam código profissional ou *pro code*, que é muito impactante no orçamento de pequenas e médias empresas.

Ainda em Pańkowska (2024) é introduzido o papel dos “facilitadores *low code*”, que

são justamente desenvolvedores profissionais que suportam, capacitam e ajudam a fortalecer a comunidade de *citizen developers*. A autora ainda faz uma segregação em dois cenários de desenvolvimento, um para pequenas e médias empresas onde o desenvolvimento é feito inteiramente pelo *citizen developer* e todos os elementos e requisitos do sistema são sua responsabilidade. O outro cenário é mais comum em grandes empresas onde há a integração com sistemas externos ou legados, e assim é necessário o envolvimento das partes interessadas na elucidação dos requisitos e na definição das lógicas de negócio. Esse segundo cenário é mais adequado ao contexto desse trabalho, porém as sugestões apresentadas pela autora são focadas num ciclo de vida gerenciado pelo próprio *citizen developer*, onde ele próprio já sabe os requisitos daquilo que quer desenvolver, e em poucos casos precisa de apoio para definí-los.

Como reforçado, em Pańkowska (2024), o desenvolvimento *low code* não é amplamente utilizado por engenheiros de software profissionais, e há ainda uma falta de padrões e materiais de apoio para a comunidade de desenvolvedores. Para os *citizen developers* esses padrões realmente são mais difíceis de serem definidos devido à heterogeneidade dos motivos que levam ao uso do *low code* e mesmo dos próprios desenvolvedores que tem bases de conhecimentos e visões muito diferentes entre si. Todavia, quando se fala de desenvolvimento profissional e a prestação de um serviço, isso se torna mais fácil. Mesmo que o ciclo de vida e processos técnicos definidos não tenham uma sobreposição exata em outros contextos, podem ser modificados ou complementados por outros profissionais antes de serem aplicados.

### 2.3.2 Exploring Low-Code Development: A Comprehensive Literature Review

O trabalho realizado em Rokis & Kirikova (2023), trás uma síntese do desenvolvimento utilizando ferramentas *low-code* abordando sua definição, suas características, seus benefícios e seus desafios. O autor trás seguinte definição para o desenvolvimento de software *low-code*:

*“(...) é uma abordagem de desenvolvimento que melhora o desenvolvimento rápido, flexível e iterativo tornando possível uma rápida tradução dos requisitos de negócios através de uma programação visual com uma interface gráfica, abstração visual, e minimizando a programação manual; envolvendo praticantes com variadas bases de conhecimento e níveis de experiência em desenvolvimento de software.”*

Uma plataforma de desenvolvimento *low-code* possui algumas funcionalidades chave que são necessárias para viabilizar um ambiente gerenciável e um desenvolvimento rápido e robusto. Uma dessas funcionalidades indicadas é o suporte à modelagem dos requisitos. A importância dessa funcionalidade segundo o autor é garantir a correta implementação dos requisitos bem como garantem a rastreabilidade e verificação. Vale citar que nem toda

plataforma possui todas as funcionalidades, elas na verdade são as mais comuns dentre várias plataformas que serviram de pesquisa.

O ciclo de vida de desenvolvimento *low-code* proposto em Rokis & Kirikova (2023) é bem semelhante a um ciclo de vida ágil, com as seguintes etapas: idealização e análise de requisitos, planejamento, design da aplicação, desenvolvimento, testes, implantação, e manutenção.

Dentre os benefícios citados pelo trabalho para a utilização do *low-code*, está a redução de custo, aceleração do ciclo de desenvolvimento, aumento da responsividades ao negócio e mercado, promoção da inovação digital e maior colaboração entre o time de desenvolvimento e negócios.

Em contraste, os desafios para essa abordagem apresentados na etapa de análise de requisitos são a especificação dos requisitos e a constante mudança dos requisitos. E um dos 13 princípios apresentados para o desenvolvimento *low-code* é justamente suportar essa mudança nos requisitos. Isso retém os clientes e vai de encontro com as necessidades do negócio, onde as organizações precisam dessa habilidade para responder a mudanças de mercado e processos. Os autores deixam claro ainda que os desenvolvedores *low-code* devem manter um mente aberta para essas mudanças, permitindo deixar fluir a dinâmica do mercado e do negócio. Pelo fato dessa abordagem de desenvolvimento ser rápida e ter um ciclo de vida mais curto, isso se torna possível mas continua desafiador.

## Capítulo 3

# Aspectos Socioeconômicos e Humanidades

A digitalização de processos empresariais, aliada à automação de tarefas e ao uso de ferramentas *low-code*, é uma estratégia indispensável para aumentar a competitividade, eficiência e segurança nas operações, conforme enfatizado por Sebrae (2023). Este estudo aponta que sua adoção pode elevar a produtividade, com ganhos de até 30%, ao reduzir em até 90% o tempo para tarefas repetitivas, permitindo que colaboradores se concentrem em atividades estratégicas. Além disso, contribui para aumentar a eficiência, uma vez que a automação otimiza fluxos de trabalho, com melhorias de até 80% na performance operacional, reduzindo prazos de entrega e aumentando a satisfação dos clientes. Outro benefício significativo é a redução de custos, com economias que podem chegar a 90% no processamento de dados, 30% na manutenção de equipamentos e 40% na gestão documental. A digitalização também melhora a experiência do cliente, pois processos digitais permitem respostas mais rápidas e personalizadas, podendo aumentar a receita em até 10% e impulsionar o crescimento das empresas em 2,2 vezes. Por fim, aumenta a segurança da informação, com controles mais rigorosos que podem ampliar a proteção de dados em até 50%, reduzindo riscos e fortalecendo a confiabilidade.

A Engenharia de Sistemas fornece a base conceitual e metodológica para gerenciar a possível complexidade envolvida na transformação digital. Segundo INCOSE (2023), essa disciplina busca desenvolver soluções integradas e coerentes que levem em conta todos os domínios de um sistema — técnico, humano, organizacional e ambiental.

Nesse contexto, o uso de ferramentas *low-code* e automação deve ser compreendido como parte de um ciclo de vida sistêmico, que abrange requisitos multidisciplinares, envolvendo áreas de negócio, tecnologia e stakeholders sociais; modelagem de processos, para representar fluxos e identificar pontos críticos de automação; validação e verificação, garantindo que as soluções automatizadas entreguem valor de forma confiável; e integração e interoperabilidade, facilitada por abordagens modulares e por plataformas *low-code* com suporte a APIs.

Com base na Engenharia de Sistemas, é possível aplicar princípios como o pensamento sistêmico, a modelagem orientada a funções e a arquitetura empresarial para guiar decisões tecnológicas alinhadas com os objetivos organizacionais e sociais.

A adoção dessas tecnologias impacta diretamente as dinâmicas de trabalho: promove

a reconfiguração de papéis, com profissionais de negócio assumindo funções mais ativas no ciclo de vida de sistemas, o que fortalece o alinhamento entre necessidades e soluções; facilita o empoderamento dos colaboradores, permitindo que usuários finais prototipem, testem e implantem soluções de forma autônoma, reduzindo burocracias; e exige requalificação, com foco no desenvolvimento de habilidades analíticas, resolução de problemas e pensamento sistêmico.

Segundo McKinsey Global Institute (2017), até 45% das atividades humanas podem ser automatizadas com tecnologias atuais. Isso não implica apenas substituição de postos de trabalho, mas principalmente transformação e valorização de funções com maior envolvimento cognitivo e decisório.

A Engenharia de Sistemas enfatiza a visão de ciclo de vida completo, o que inclui o descarte responsável, a adaptabilidade dos sistemas ao longo do tempo e o uso sustentável de tecnologias.

O uso conjunto de automação, plataformas *low-code* e princípios da Engenharia de Sistemas representa uma convergência poderosa para transformar organizações de forma eficaz, inclusiva e sustentável. Essa abordagem interdisciplinar oferece não apenas ganhos operacionais, mas também impactos positivos sobre a estrutura organizacional, o desenvolvimento humano e a sustentabilidade ambiental.

## Capítulo 4

# Metodologia

Como fundamento do desenvolvimento desse trabalho, está a utilização das técnicas de ES para tornar o dia-a-dia de trabalho e gestão de atividades mais claros e concisos. Além disso, busca-se a especificação das deficiências no processo ou ciclo de vida que, apesar de já serem percebidas, não estão bem colocadas ou esclarecidas.

A primeira etapa de trabalho consiste na definição do ciclo de vida como é hoje. Através do estudo do fluxo atual, são definidas as associações das etapas elencadas com as fases de cada estágio do ciclo de vida da ES, bem como seus entregáveis e os pontos de decisão entre elas. Recapitulando, o serviço prestado é basicamente o desenvolvimento de diferentes sistemas para atender requisitos específicos das áreas de negócios trazidos ao time.

Para complementar esse ciclo de vida, é desenvolvida a arquitetura dos elementos do sistema e estabelecida a relação entre eles e com as funcionalidades. Assim, são definidas todas as opções de possíveis sistemas do serviço prestado, através das combinações de elementos do sistema e das funcionalidades.

Após essa primeira parte do trabalho, com os artefatos e documentações já produzidas, é feita a análise e listagem dos problemas e deficiências encontradas no ciclo de vida. Tendo em mãos os problemas identificados, são feitas propostas para saná-los ou amenizá-los, sendo detalhado como e o que deve ser feito, quais materiais ou instrumentos são utilizados, e o que se espera com a proposta sugerida.

Por fim, propõe-se uma coleta de resultados para a análise e avaliação do trabalho realizado.

### 4.1 Ciclo de Vida

Inicialmente definiu-se em formato de fluxograma o processo do serviço prestado, para dessa forma ser mais fácil o entendimento da ordem das etapas. Depois traremos essa representação para o formato esperado na ES. A Figura 4.1 mostra esse fluxograma.

Cada um dos blocos do fluxograma representa uma etapa do processo, e cada etapa é



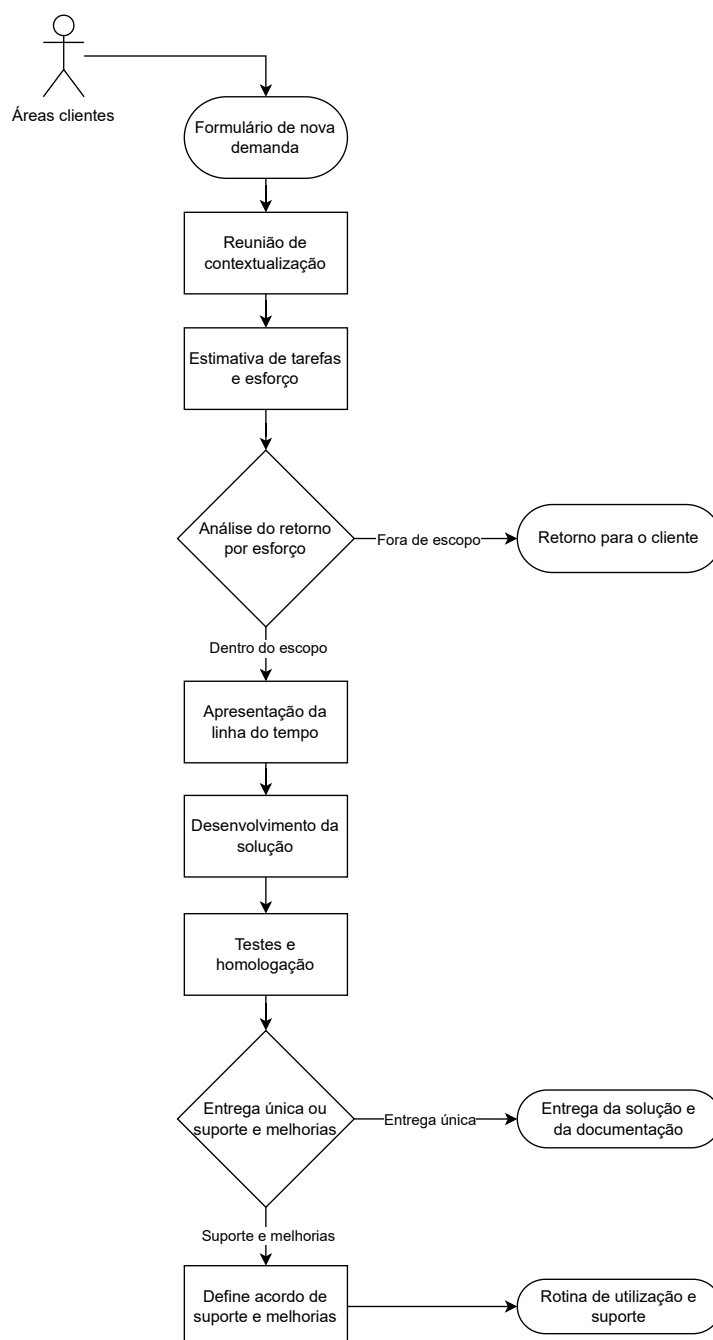


Figura 4.1: Fluxograma do processo que descreve o serviço prestado.

descrita abaixo:

- **Formulário de novo desenvolvimento**

- Descrição: Esse formulário é o ponto de entrada de uma nova demanda para o time. Quando uma área cliente manifesta a necessidade e procura o auxílio do time, ela é direcionada a responder esse formulário online com uma série de perguntas. E há ainda, casos em que auditorias internas encontram irregularidades em processos existentes, e os encaminham para entrar em contato com o time para possíveis soluções de remediação, novamente eles são direcionados para responder

o formulário.

- **Objetivos:** Obter informações de contato e departamento do requisitante bem como o centro de custo em caso de possíveis cobranças no futuro. Levantar informações iniciais sobre a solução desejada como classificação dos dados envolvidos, interdependência com outras áreas e quais dores seriam sanadas com o desenvolvimento. Coletar dados e percepções que ajudem a mensurar o valor para a área e para a empresa como um todo, ao ser desenvolvida uma solução para essa demanda. Solicitar documentação disponível do processo atual caso exista.
- **Saídas:** Registro das respostas ao formulário, anexos e documentações enviadas.

- **Reunião de contextualização**

- **Descrição:** Após recebida e revisada a resposta do formulário, o gerente de projetos organiza essa reunião com os desenvolvedores envolvidos e o requisitante, orientando-o a convidar todas as outras partes interessadas. Nessa reunião, o requisitante faz uma nova explicação dos problemas e possíveis de soluções, e também são sanadas possíveis dúvidas sobre as respostas do formulário e sobre a documentação enviadas anteriormente.
- **Objetivos:** Entender o papel de todas as partes interessadas na operação. Ver o processo atual existente sendo executado integralmente pelos responsáveis. Obter exemplos de artefatos do processo como emails enviados, arquivos gerados ou compartilhados, indicadores calculados e outros artefatos que sejam importantes, caso existam. Discutir casos de uso e cenários que não existem atualmente mas são desejáveis na solução. Discutir políticas de retenção de dados.
- **Saídas:** Gravação da reunião e operação atual. Arquivos com os exemplos de artefatos do processo atual.

- **Estimativa de tarefas e esforço**

- **Descrição:** Nessa etapa os desenvolvedores de maior senioridade realizam uma sintetização das necessidades e requisitos para propor um ou mais cenários de solução. Assim, em um curto espaço de tempo, são definidas histórias de usuário em um alto nível, sem muito detalhamento, para que seja estimado o esforço e tempo total necessário para o desenvolvimento dos cenários de solução.
- **Objetivos:** Definir cenários de solução para o caso trazido pelo cliente. Estimar o esforço e tempo necessário para executar as tarefas definidas para cenário.
- **Saídas:** Documento com cenários propostos, tempo e esforço necessários para cada um.

- **Análise do retorno por esforço**

- **Descrição:** Ao receber o documento da etapa anterior o gerente de projetos se reúne com o gerente e o diretor da área, onde realizam uma análise de enquadramento de escopo de todas as submissões que estão aguardando o início do desenvolvimento, e depois uma priorização das que serão iniciadas. Essa etapa é realizada em uma

menor frequência, geralmente bimestralmente, salvo exceções em que envolvem uma falha em auditoria e que precisam de mais celeridade.

- Objetivos: Definir se a demanda solicitada está no escopo do time (dentro das capacidades técnicas do time, dentro do tempo máximo de desenvolvimento por projeto, não conter dados estritamente confidenciais, gerar um valor que valha o investimento). Priorizar a ordem de execução das tarefas.
- Saídas: Lista com os projetos selecionados e priorizados para execução.

- **Retorno para o cliente**

- Descrição: Quando um projeto é definido como fora de escopo, é marcada uma reunião com o requisitante e apresentado os motivos da decisão. A depender do motivos são passadas orientações para a área cliente, como procurar outro time de desenvolvimento interno na empresa, utilizar uma ferramenta existente no mercado, contratar time externo para executar o projeto, ou mesmo redefinir os requisitos e necessidades e submeter novamente para análise. Essa etapa é um dos possíveis fins do fluxo de serviço.
- Objetivos: Apresentar uma devolutiva ao requisitante da demanda. Orientar sobre possíveis alternativas.
- Saídas: Ata da reunião com principais tópicos, decisões, e participantes.

- **Apresentação da linha do tempo do projeto para o cliente**

- Descrição: Quando um projeto é definido como dentro do escopo e priorizado, é marcada uma reunião com o requisitante para dar um retorno sobre essa priorização. O gerente de projetos apresenta como ficou a linha do tempo estimada para os diferentes cenários e quando será iniciado o desenvolvimento.
- Objetivos: Apresentar o cronograma e início do desenvolvimento. Definição de qual cenário será desenvolvido. Coletar possíveis ressalvas quanto aos prazos informados. Definir agendas para acompanhamento do progresso do desenvolvimento.
- Saídas: Ata da reunião com principais tópicos, decisões, e participantes. Agenda com as reuniões de acompanhamento.

- **Desenvolvimento da solução**

- Descrição: Esta é a etapa dedicada ao trabalho de desenvolvimento de programação da solução proposta e escolhida. A partir das histórias de usuário criadas anteriormente, semanalmente são criadas tarefas detalhando as atividades a serem realizadas na próxima semana para cada história de usuário.
- Objetivos: Detalhar as histórias de usuário em tarefas. Executar as tarefas de cada história de usuário.
- Saídas: Solução desenvolvida com todas as suas funcionalidades.

- **Teste e homologação**

- Descrição: Ao ser finalizado o desenvolvimento da solução é realizada uma demonstração ao requisitante e às partes interessadas. A solução é implantada no

ambiente de qualidade para que os usuários façam testes e validem se as funcionalidades e requisitos foram cumpridos. Em paralelo o time de desenvolvimento trabalha para corrigir problemas encontrados.

- Objetivos: Usuários validarem as funcionalidades da solução desenvolvida. Corrigir possíveis problemas encontrados. Definir uma data para fim dos testes em qualidade.
- Saídas: Relatório de problemas encontrados na ferramenta. Relatório com possíveis melhorias futuras. Relatório com as correções realizadas.

- **Entrega única ou suporte e melhorias**

- Descrição: Nessa etapa se inicia o encerramento do projeto. É realizada uma reunião com o requisitante e as partes interessadas, onde discute-se sobre como foi o andamento do projeto e são explicadas as opções de continuidade para a área cliente demandante. Onde existem as opções de contratarem um plano de suporte e melhorias com o time de desenvolvimento ou seguirem por conta própria com a solução desenvolvida.
- Objetivos: Formalizar a entrega da solução desenvolvida. Apresentar com mais detalhes as opções de continuidade. Estipular um prazo para retorno sobre qual será a opção escolhida, caso não seja decidido na reunião.
- Saídas: E-mail com a formalização da entrega. Prazo para decisão dos próximos passos.

- **Envio do pacote da solução e entrega da documentação**

- Descrição: Caso o requisitante e sua área decidam por não contratar o pacote de suporte e melhorias, são enviadas por e-mail as instruções de como dar continuidade à operação da solução, bem como o pacote contendo a própria solução desenvolvida. Espera-se um atestado de recebimento por parte do requisitante.
- Objetivos: Enviar documentação funcional da solução desenvolvida. Enviar pacote com a solução desenvolvida.
- Saídas: E-mail com os documentos de entrega do projeto. Atestado de recebimento dos arquivos, que simboliza o encerramento do projeto.

- **Acordo de suporte e melhorias**

- Descrição: Ao optar por seguir com o pacote de suporte e melhorias, são definidas as horas mensais dedicadas e valor a ser pago para esse pacote. Além disso, é demonstrado como são abertas requisições de suporte e melhorias.
- Objetivos: Definir horas e custos mensais do pacote de suporte e melhorias. Obter autorização para débito no centro de custo da área cliente.
- Saídas: E-mail com acordo de suporte e melhorias.

- **Rotina de manutenção e suporte**

- Descrição: A aplicação é implantada no ambiente produtivo e liberada para operação aos usuários.

- Objetivos: Implantar a solução em produção. Encerrar o projeto de desenvolvimento. Iniciar suporte e manutenção da solução.
- Saídas: E-mail com instruções de acesso e utilização da ferramenta. E-mail com as instruções para criação de requisições de suporte.

Para as áreas que não optaram por seguir com um acordo de suporte e melhorias, caso seja necessário uma nova implementação ou modificação relacionada à solução desenvolvida, deverão responder novamente o formulário e passar por todo o processo de análise de retorno por investimento e priorização, como uma nova demanda. Já as que tem o acordo de suporte e melhorias, podem utilizar as horas mensais e fazer o balanceamento das mesmas para realizar essas implementações sem passar por todo o processo. Entretanto, mesmo com o acordo, caso seja uma modificação muito grande ou uma nova funcionalidade que demande muito tempo de desenvolvimento, é necessário responder o formulário e iniciar como uma nova demanda ao time, porém ao fim não precisam de um novo acordo de suporte e melhorias, pois já o possuem.

Trazendo o ciclo de vida em “V” mencionado anteriormente, as áreas que possuem o contrato de suporte percorrem todas os estágios, dos dois lados do “V”. Já as outras áreas, param na primeira metade.

#### 4.1.1 Adequação de representação

Tendo todas as etapas do processo sido explicadas, são feitas sua relações com os estágios e fases do ciclo de vida proposto na engenharia de sistemas. De maneira descritiva, sem ainda avaliar a otimização ou melhoria do ciclo de vida, podemos representar o ciclo de vida com as figuras 4.2, 4.3 e 4.4.



Figura 4.2: Relação com as fases do estágio de Conceito

Olhando para o estágio de Conceito na figura 4.2, na fase de **Análise das necessidades** temos as etapas de **Formulário de nova demanda** e **Reunião de contextualização**. De fato, nessas etapas é realizado o primeiro contato com as áreas clientes e visto a operação que se deseja digitalizar ou otimizar. De forma macro já é entendida as principais capacidades do sistema bem como uma pré-análise de quais tecnologias utilizar.

Agora na fase de **Exploração do Conceito** foram alocadas as etapas de **Estimativa de tarefas e esforço** e **Análise de retorno por esforço**. A etapa de **Estimativa de tarefas e esforço** já começa a documentar alguns requisitos e separar subsistemas para mensuração de esforço de desenvolvimento, nela são propostas mais de uma abordagem de solução quando aplicável e também é analisada a viabilidade técnica. A etapa de **Análise de retorno por esforço** também foi posicionada nessa fase pois nela ocorre a análise de viabilidade estratégica de desenvolvimento do sistema.

Foi destacado ainda o ponto de decisão entre a fase de **Exploração do Conceito** e de **Definição do Conceito**, que é justamente após a análise de viabilidade estratégica. Nesse ponto, é decidido se será empenhado mais esforço a fim de continuar o trabalho para o desenvolvimento da solução ou o cancelamento da iniciativa, interrompendo o ciclo de vida de avançar, acontece nesse caso a etapa de **Retorno ao Cliente**.

Na fase de **Definição do Conceito** temos a etapa de **Apresentação da linha do tempo**, pois nessas apresentação são sugeridas as alternativas de desenvolvimento que satisfaçam os requisitos estabelecidos, de maneira integral ou não, e suas justificativas para serem utilizadas.

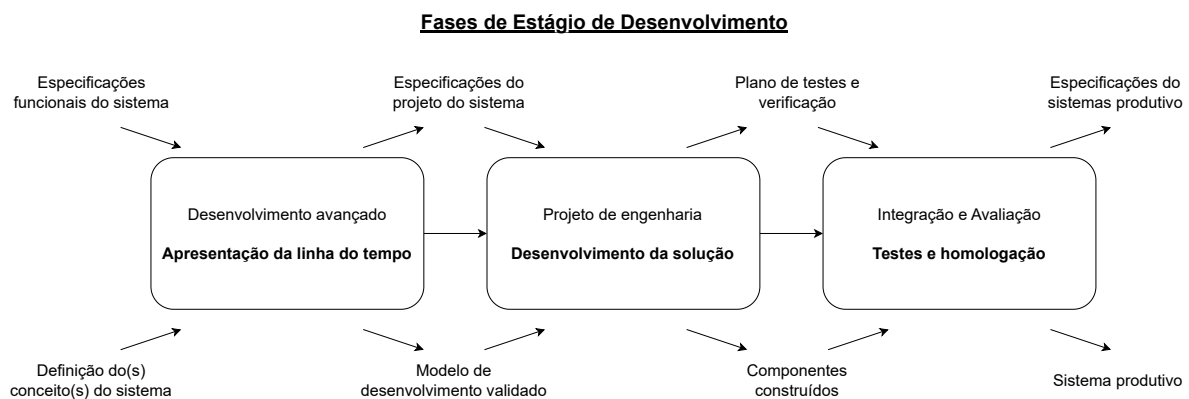


Figura 4.3: Relação com as fases do estágio de Desenvolvimento

Voltando a atenção para o estágio de Desenvolvimento, na primeira fase de **Desenvolvimento Avançado** temos novamente a etapa de **Apresentação da linha do tempo**, pois é realizada a apresentação das histórias de usuário, buscando a validação das partes interessadas. São apresentadas as principais funcionalidades junto com a programação de desenvolvimento e também discutido o cronograma, captando os riscos do mesmo, no ponto de vista dos outros interessados.

Na fase de **Projeto de Engenharia** se encontra a etapa de **Desenvolvimento da solução**, é feita a programação dos componentes do sistema através das histórias de usuário definidas semanalmente. Nesse caso os desenvolvedores e engenheiros empenham os esforços nos componentes onde são especialistas, e depois de desenvolvidos os testam individualmente.

Por fim, na fase de **Integração e avaliação** está alocada a etapa de **Testes e Homologação** visto nessa etapa é realizada a integrações entre os componentes e os testes juntamente com a área demandante e um grupo pequeno do time operacional para simular o

dia a dia de utilização da solução. São avaliadas a qualidade e completude das funcionalidades pré-estabelecidas anteriormente, e uma avaliação geral do escopo do sistema.

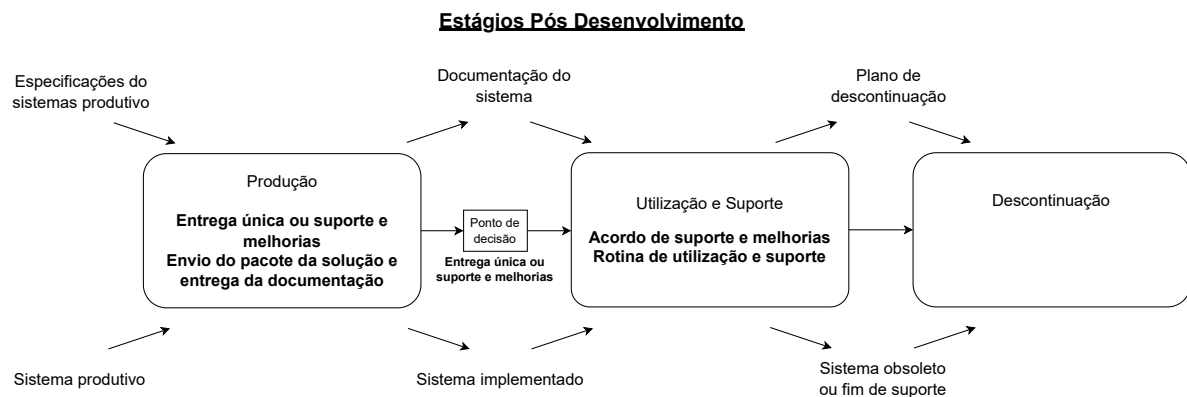


Figura 4.4: Relação com os estágios Pós-Desenvolvimento

Nos estágios de pós-desenvolvimento, em **Produção** temos as etapas de **Entrega única ou suporte e melhorias** e **Envio do pacote da solução e entrega da documentação**, na primeira ocorre o aceite da entrega da solução/sistema e após isso é realizado a implantação do sistema em ambiente produtivo. Nesse ponto a documentação do sistema também deve estar feita e disponibilizada. Uma nuance do estágio de produção ocorre de acordo com a decisão tomada pela area cliente na etapa **Entrega única ou suporte e melhorias**, pois caso não optem pelo plano de suporte e melhoria, a implantação do sistema ocorre em um ambiente gerenciado por eles, e não pelo time de desenvolvimento, e por isso ocorre a etapa **Envio do pacote da solução e entrega da documentação**.

É destacado outro ponto de decisão, entre o estágio de **Produção** e o de **Utilização e Suporte**, pois como dito, o ciclo de vida continua apenas se for feito a contratação do serviço de suporte. Ressaltando que o sistema vai ser utilizado, mas deixa de ser responsabilidade do time de desenvolvimento.

O estágio de **Utilização e Suporte** compreende as etapas de **Acordo de suporte e melhorias** e **Rotina de utilização e suporte**, onde os usuários, já utilizando o sistema, relatam falhas, problemas e melhorias que são corrigidas ou implementadas ainda nesse estágio, ou seja, a segunda metade do ciclo de vida em “V”.

Ainda não existe uma etapa criada para **Descontinuação**, sendo que o time é relativamente novo com cerca de dois anos de criação não houve caso de obsolescência mapeado até então.

#### 4.1.2 Avaliação do ciclo de vida

Ao longo da repetição desse ciclo de vida para cada solução desenvolvida, já havia sido notado um deficiência na parte de definição e validação de conceito. Por vezes, já no fim da fase de **Projeto de Engenharia** ou durante a fase de **Integração e avaliação** eram identificadas

funcionalidades essenciais que não havia sido mapeadas, ou ainda um desenvolvimento que tecnicamente funcionava mas que desempenhava uma funcionalidade desalinhada com a real intenção das partes interessadas.

Olhando para a representação do ciclo de vida de acordo com as definições da ES, fica mais claro identificar a raiz dessa deficiência bem como o que pode ser feito para mudar. A transição entre os estágios de Conceito e Desenvolvimento, marcado pelas fases de Definição de Conceito e Desenvolvimento Avançado, apresenta uma falta de artefatos ou documentos que permitiriam essa transição de forma suave mas concisa.

Um ponto relevante é que essas duas fases estão sendo representadas por uma única etapa no processo atual: a **Apresentação da linha do tempo**. Esta etapa contempla apenas parcialmente as características necessárias das fases, abordando aspectos como a definição e validação do conceito do sistema em nível macro. Contudo, ela não desce para níveis mais detalhados da hierarquia, deixando de lado elementos essenciais como a validação de subsistemas e a alocação de funções aos componentes.

Olhando em conjunto para a tabela 2.2 sobre a materialização do sistema e as fases de Definição de Conceito e Desenvolvimento avançado, é observado e destacado em vermelho e laranja os seguintes pontos falhos, mostrados na Figura 4.5.

Em vermelho, temos as tarefas ou atividades que são completamente ignoradas nessas fases:

- Arquitetura física e funcional: Não é realizada a documentação da arquitetura do sistema na fase de definição de conceito. Embora os desenvolvedores já possuam uma boa noção de como seria o resultado final, a ausência desses artefatos impede a validação dos subsistemas e a realização de um gerenciamento de riscos efetivo. Sem as arquiteturas definidas, mesmo em alto nível, as outras partes interessadas não conseguem contribuir adequadamente nas ponderações.
- Alocação de funções a componentes: Como não existe nenhuma das duas arquiteturas, essa tarefa não é realizada. Essa tarefa seria justamente responsável por definir as relações entre as arquiteturas, gerando assim uma forma de garantir a rastreabilidade do sistema.
- Validação de subsistemas: Essa tarefa também não é realizada devido à falta da documentação das arquiteturas do sistema. Essa deficiência constitui a raiz de algumas falhas de entendimento dos requisitos, que acarretam a inclusão ou modificação de funcionalidades no estágio de **Integração e avaliação**, afetando diretamente os prazos estipulados.

Já em laranja, temos algumas atividades que são parcialmente realizadas no ciclo de vida atual:

- Análise de alternativas: Essa atividade é parcialmente realizada, pois acontece a discussão de alternativas com a área cliente durante a etapa de **Apresentação da linha do tempo**. Entretanto, não há a documentação das mesmas e, em alguns casos, uma melhor definição de conceito já descartaria opções inviáveis de serem implementadas.



Fase \ Nível	Estágio de Conceito			Estágio de Desenvolvimento		
	Análise das necessidades	Exploração do Conceito	Definição do Conceito	Desenvolvimento avançado	Projeto de engenharia	Integração e Avaliação
Sistema	Define as capacidades e efetividades do sistema	Identifica, explora e sintetiza conceitos	Define conceitos selecionados e especificações	Validação de conceito		Teste e avaliação
Subsistema		Define requisitos e garante viabilidade	Define arquitetura funcional e física	Validação de subsistemas		Integração e testes
Componente			Aloca funções a componentes	Define especificações	Projeto e testes	Integração e testes
Subcomponente				Aloca funções a subcomponentes	Projeto	
Peças					Fazer ou comprar	

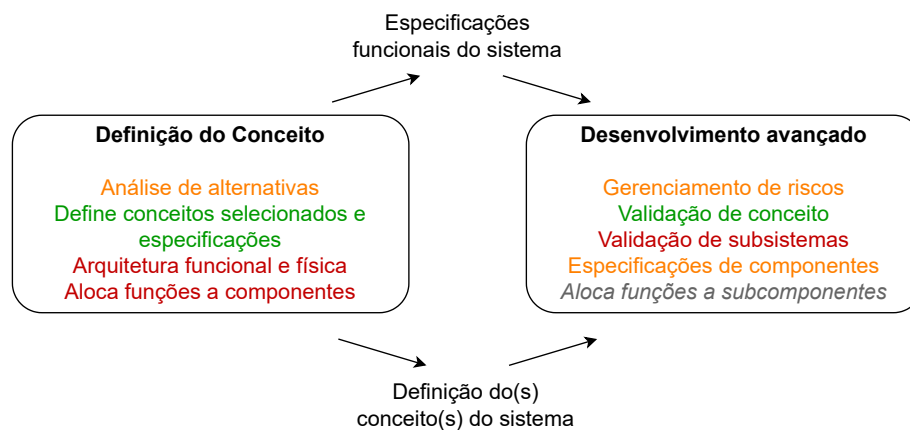


Figura 4.5: Destaque dos pontos falhos no ciclo de vida

- **Gerenciamento de riscos:** Do ponto de vista gerencial ou estratégico, é realizado um gerenciamento de risco sucinto no que diz respeito aos prazos e marcos de entrega, novamente na etapa de **Apresentação da linha do tempo**. Por outro lado, na parte funcional e técnica, esse gerenciamento deixa a desejar. Como não há uma definição consistente e suficientemente detalhada, é incerto dizer quais são os possíveis pontos críticos ou que exigem maior atenção durante o desenvolvimento. Logo, não são planejadas ações para reduzir ou contornar os riscos envolvidos.
- **Especificações de componentes:** Essa tarefa está classificada como parcialmente realizada, pois era realizada simultaneamente com o desenvolvimento da solução na fase de **Projeto de Engenharia**. Semanalmente, as histórias de usuário eram detalhadas com as tarefas e especificações para serem realizadas na semana seguinte. Isso não permite uma validação prévia e, conseqüentemente, interfere também no gerenciamento de riscos. A realização dessa tarefa no momento correto interfere diretamente no bem-estar do projeto e da construção do sistema.

Por fim, em verde temos tarefas que são executadas de maneira satisfatória durante o ciclo de vida do sistema:

- Define conceitos selecionados e especificações: Ao observar a tabela de materialização do sistema, percebe-se que essa tarefa diz respeito ao primeiro nível, ou seja, são especificações e conceitos em nível de sistema, sem descer para níveis de maior detalhamento. Na etapa **Apresentação da linha do tempo** são apresentados possíveis cenários e alternativas criados anteriormente, como na etapa de **Estimativa de tarefas e esforço**, e é definido quais serão de fato considerados, bem como repassadas as suas especificações.
- Validação de conceito: Na mesma hierarquia, essa tarefa diz respeito à validação do conceito no nível de sistema, e isso acontece também na etapa **Apresentação da linha do tempo**.

O fato de o sistema ser construído utilizando majoritariamente ferramentas de desenvolvimento *low-code* faz com que exista um nível de abstração já intrínseco ao sistema. Dessa forma, durante as fases de **Desenvolvimento Avançado** e **Projeto de Engenharia**, não é necessário descer aos dois últimos níveis da hierarquia do sistema, ou seja, não são realizadas tarefas relacionadas aos subcomponentes ou às peças. Isso foi destacado na figura 4.5, com as duas últimas linhas da tabela esmaecidas e com a tarefa **Aloca funções a subcomponentes** em cinza na fase de **Desenvolvimento Avançado**.

## 4.2 Proposta de modificações

A partir do que foi apresentado nas seções anteriores, fica sugestivo que ações devem ser tomadas para que o ciclo de vida seja mais eficiente, focando nas fases de **Definição do Conceito** e **Desenvolvimento Avançado**. Na fase de definição de conceito faltam tarefas essenciais, que são dependências para a realização das tarefas da próxima fase. Elas são a criação das arquiteturas física e funcional, e a alocação de funções a componentes.

### 4.2.1 Arquitetura do Sistema

É necessário estipular um estilo de arquitetura com padrões que deverão ser seguidos e atualizados, de maneira que a equipe consiga discutir o mesmo com as áreas clientes e entre si, avaliando a qualidade estrutural e a escolha dos componentes do sistema. Tendo em vista que o tempo de vida do desenvolvimento do sistema é curto, padrões muito detalhados e que é necessário um grande esforço para criação não são interessantes para compor esse estilo de arquitetura.

É proposto então a criação de dois artefatos de arquitetura, o primeiro seguindo um padrão que combina a arquitetura funcional com a lógica, baseado nas informações e requisitos obtidos para o sistema. Já o outro seguindo o padrão definido para a arquitetura física do sistema, de forma a documentar os componentes e subsistemas que serão responsáveis por

cobrir as funcionalidades e lógicas definidas anteriormente.

### Padrão para Arquitetura Funcional e Lógica

A documentação das arquiteturas funcionais e lógicas são importantes para a construção de uma boa arquitetura física e para a que a definição e validação de conceito seja assertiva. Como a arquitetura funcional pode compor a arquitetura lógica, optou-se por unificar essas duas em únicos documentos, de maneira que seja otimizado o tempo empenhado para esse desenvolvimento.

Esse padrão separa as funcionalidades em três grupos: Funcionalidades de Dados, Processos e Automações ou Documentos. Em cada grupo foram elencadas as funcionalidades básicas de um sistema genérico, representadas pelos retângulos preenchidos de azul. De cada uma dessas funcionalidades são derivados tópicos que elucidam o que deve ser documentado para expressar a lógica por trás da função de que derivam. As Figuras 4.6, 4.7 e 4.8 mostram esses grupos.

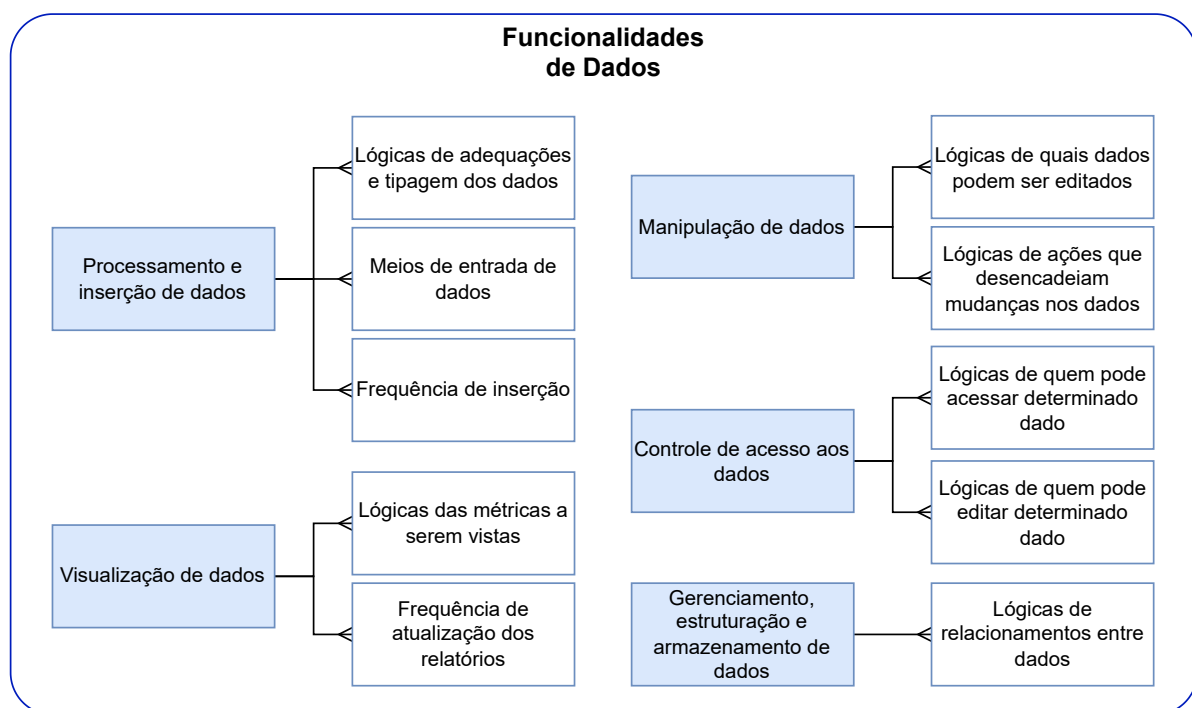


Figura 4.6: Padrão de arquitetura funcional e lógica relacionadas aos dados.

Nesse padrão, não é definido com qual ferramenta ou tecnologia cada funcionalidade será implementada, pois isso será abordado posteriormente na arquitetura física. Cada sistema desenvolvido pode apresentar funcionalidades específicas não contempladas neste padrão devido às particularidades de cada projeto. Quando essas funcionalidades adicionais surgirem, elas devem ser incorporadas à documentação seguindo o mesmo padrão aqui estabelecido.

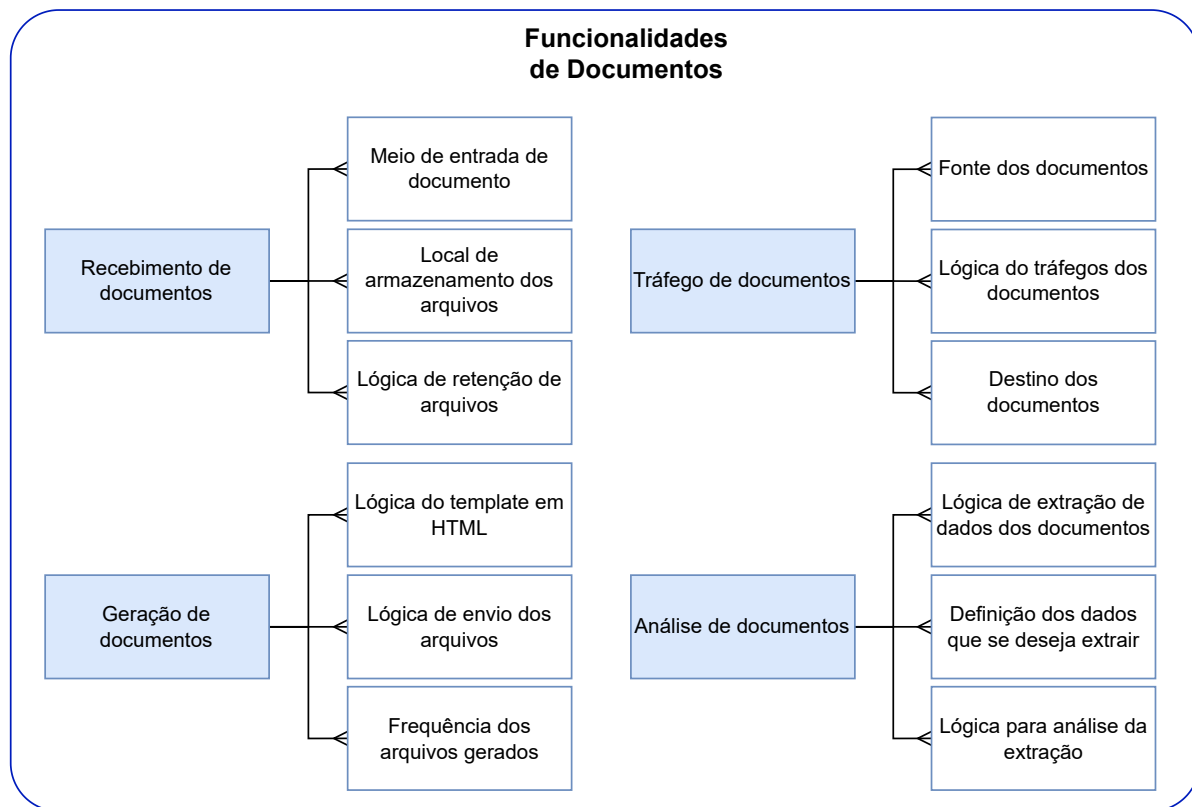


Figura 4.7: Padrão de arquitetura funcional e lógica relacionadas a documentos.

### Padrão para Arquitetura Física

O padrão de arquitetura física proposto mantém um nível de detalhamento apropriado para o contexto de desenvolvimento. A utilização de ferramentas *low code* permite essa simplificação sem comprometer a eficácia da documentação. Embora não seja prática comum incluir modelos ou fornecedores específicos na documentação de arquitetura física, o contexto já estabelecido da equipe de desenvolvimento justifica a inclusão desses elementos nesta proposta.

Os componentes do sistema são construídos utilizando majoritariamente as ferramentas disponíveis na *Power Platform* sendo elas:

- **Power Apps:** É uma ferramenta para o desenvolvimento ágil de aplicações personalizadas, utilizando pouco ou nenhum código. Ele permite que usuários construam interfaces gráficas interativas, conectadas a múltiplas fontes de dados, como o *Dataverse*, SQL Server, Excel e SharePoint. Essa ferramenta é ideal para digitalizar processos manuais e implementar soluções específicas de negócios com rapidez e escalabilidade.
- **Power Automate:** (anteriormente conhecido como Microsoft Flow) é uma ferramenta de automação de processos que permite criar fluxos de trabalho entre diferentes serviços e aplicações. É amplamente utilizado para automatizar tarefas rotineiras, como envio de notificações, aprovações de documentos, sincronização de dados entre sistemas e integração com APIs.
- **Power BI:** É uma plataforma de *Business Intelligence* que permite a criação de relatórios

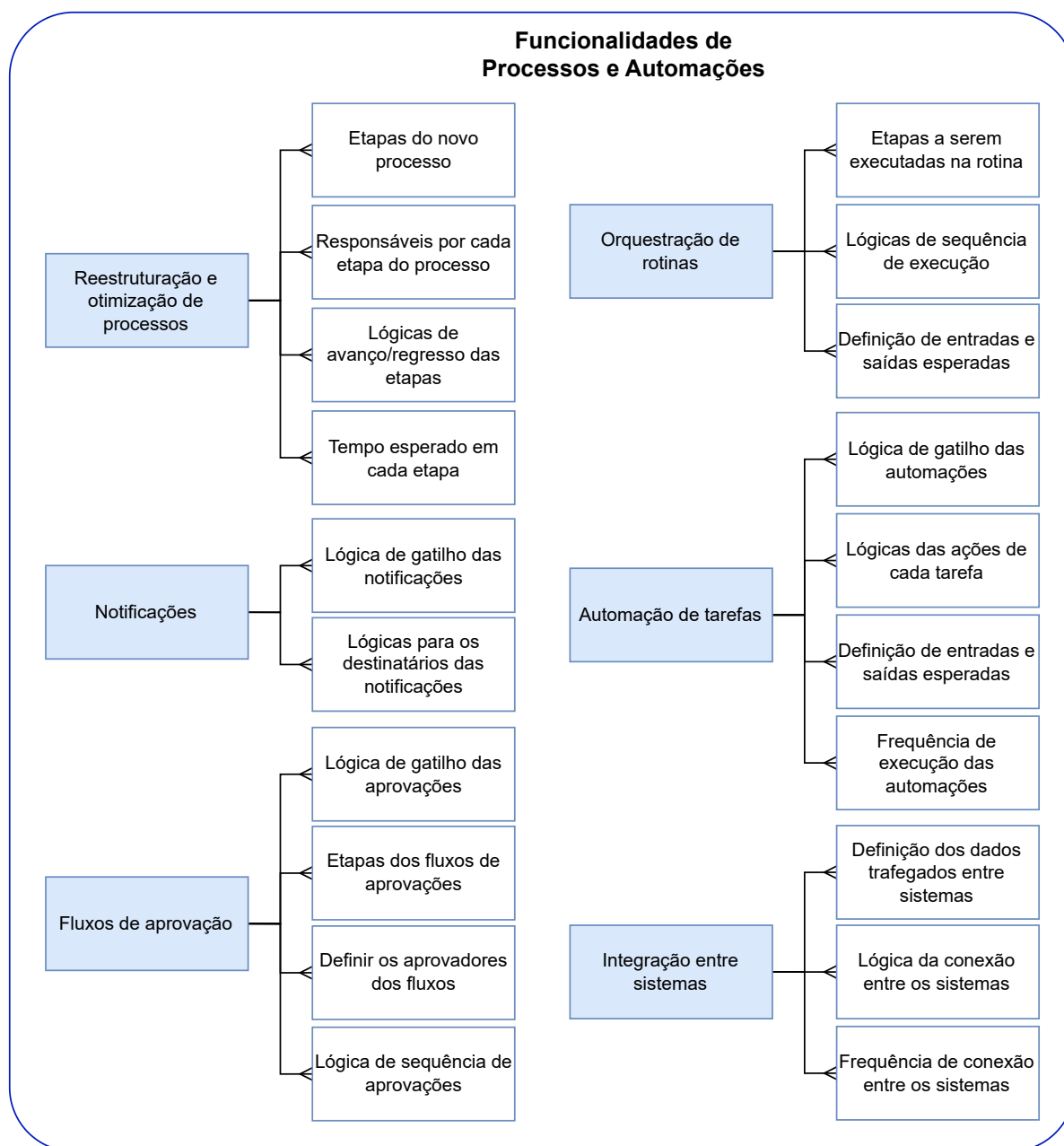


Figura 4.8: Padrão de arquitetura funcional e lógica relacionadas a processos e automações.

interativos e dashboards dinâmicos. Ele facilita a análise de dados provenientes de múltiplas fontes, promovendo a visualização e a tomada de decisões baseadas em evidências. Entre seus principais recursos estão: conexão com diferentes bancos de dados, uso de linguagem DAX para cálculos, compartilhamento de painéis e integração com outras ferramentas Microsoft.

- **Copilot Studio:** Anteriormente conhecido como *Power Virtual Agents*, é a ferramenta da Power Platform destinada à criação de assistentes virtuais e *chatbots* com baixa ou nenhuma necessidade de codificação. Com sua nova abordagem centrada em IA generativa, permite desenvolver copilotos inteligentes capazes de interagir com usuários,

acessar dados corporativos e executar ações automatizadas. É amplamente utilizado em cenários de atendimento ao cliente, suporte interno e automação de processos por meio de linguagem natural.

- **Microsoft Dataverse:** Ou simplesmente *Dataverse* é a plataforma de dados subjacente à Power Platform, projetada para armazenar, gerenciar e proteger dados empresariais estruturados. Ele fornece um modelo de dados comum, com suporte nativo a tipos de dados complexos, relacionamentos, regras de negócio, validações e integração com segurança do Microsoft Entra ID (antigo Azure AD). O Dataverse é fundamental para garantir consistência e integridade de dados em soluções desenvolvidas com o *Power Apps*, *Power Automate*, *Power BI* e *Copilot Studio*.

Na figura 4.9 pode ser observado o padrão de arquitetura física sugerido para os sistemas desenvolvidos. Ela contém todos os possíveis elementos do sistema que podem ser utilizados para a arquitetura final de cada projeto executado bem como o agrupamento de subsistemas.

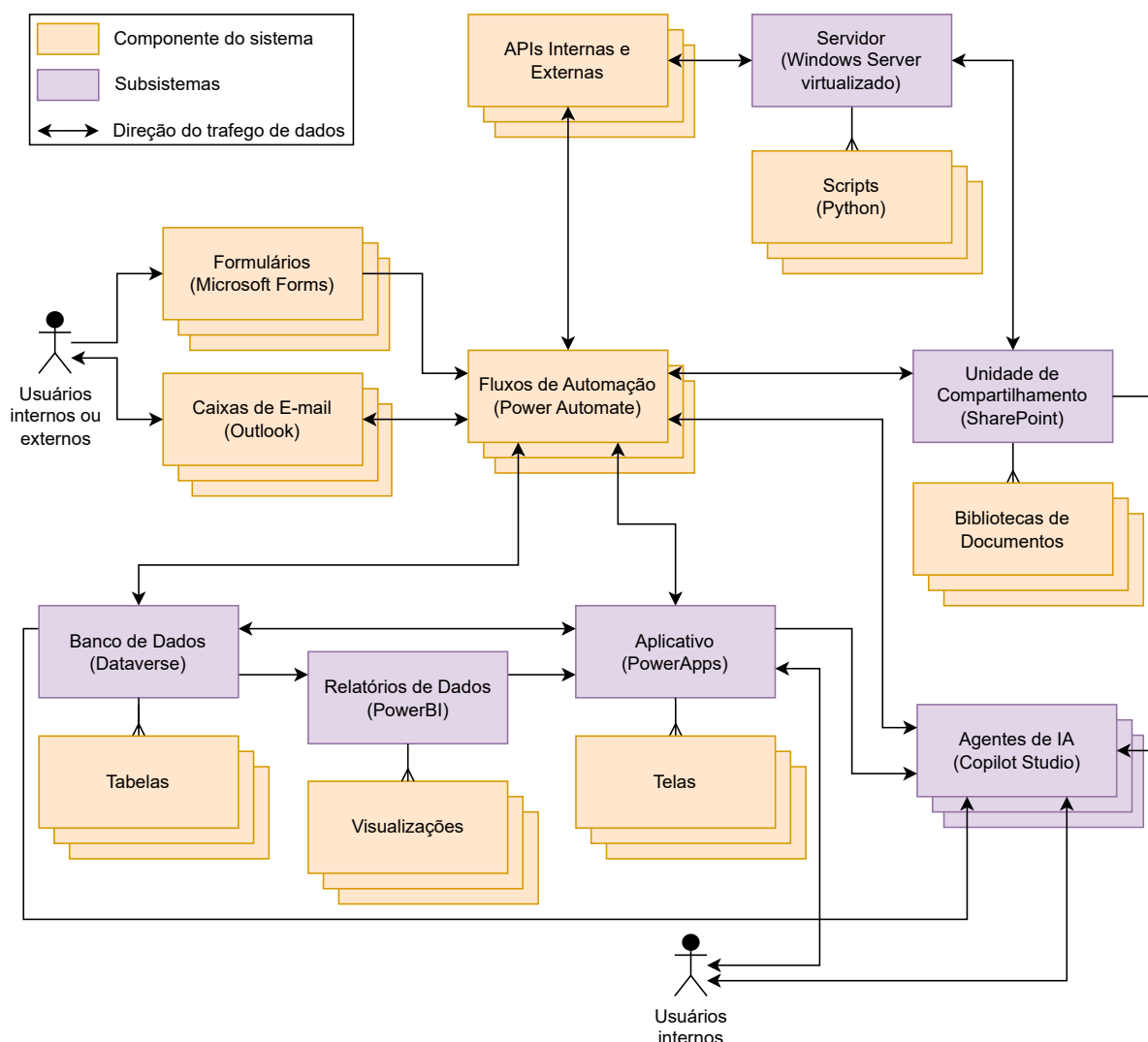


Figura 4.9: Padrão de arquitetura física dos sistemas desenvolvidos.

Os Formulários são componentes do sistema utilizados para coleta de dados e iniciar

algum processo a partir de sua resposta. Sua utilização acontece principalmente quando existe o envolvimento de algum fornecedor ou cliente externo, pois são acessíveis sem necessidade de uma conta interna da empresa. Cada sistema ou subsistema pode utilizar um ou mais formulários.

O E-mail ou mais especificamente uma caixa de e-mail compartilhada é um componente que pode ser utilizado tanto para comunicações e notificações do sistema quanto para recebimento de dados e arquivos por terceiros, visto que a possibilidade de anexo nos Formulários são exclusivos quando o uso é estritamente interno. Normalmente cada sistema possui uma caixa compartilhada única para seus processos, mas há casos em que são necessária múltiplas caixas.

Os Fluxos de Automação são o esqueleto da integração entre os subsistemas e sistemas externos. Cada Fluxo é um componente do sistema que realiza um conjunto de ações a partir de um gatilho. Esses fluxos são classificados em três tipos, e isso se dá pelo tipo de gatilho que os acionam. Fluxos Agendados executam sua tarefas de maneira recorrente (a cada hora, diariamente, semanalmente, mensalmente ...) ou única a partir de uma data de início. Fluxos Instantâneos são acionados pela ação de um usuário em um aplicativo ou chat bot, ou são acionados por outros fluxos. Fluxos Automatizados utilizam conectores já disponíveis para serem acionados a partir de um evento em algum outro sistema, por exemplo, ser acionado quando um formulário for respondido, quando um e-mail chegar na caixa de entrada, quando uma linha for criada em uma tabela no banco de dados, quando um arquivo for modificado, quando um acordo for assinado, etc. Existem centenas de conectores disponíveis para diversos sistemas, aplicações e ferramentas. Além disso, esses Fluxos de Automação são utilizados para consumir APIs que permitem métodos HTTP, criando assim integrações com sistemas e ferramentas externas que não possuem conectores disponíveis mas que possuem API, bem como sistemas internos que foram criadas por outros times e que também tem uma API disponível que seja acessível pela internet.

O Banco de Dados é um subsistema que armazena os dados do sistema, é relacional e necessita de uma modelagem e estruturação planejada, definindo os tipos dos dados de cada coluna nas tabelas, bem como os relacionamentos entre as tabelas criadas. Além disso, possui funcionalidades de gerenciamento de acesso aos dados.

O Relatório de Dados apresentam gráficos, medidas e tendências, agrupados pelo nome de Visualizações, com o intuito de apresentar dados para decisões estratégicas de gestores. Os Relatórios são considerados subsistemas pois tanto a manipulação dos dados quanto os cálculos para apresentar determinadas medidas podem levar a uma complexidade elevada. Este podem ser inseridos dentro de aplicativos e sites para uma facilidades de acesso aos usuários.

As APIs Internas e Externas foram elencadas como componentes do sistema pois são essenciais para viabilizar o consumo e tráfego de dados entre sistemas, atuando ainda como ponte entre sistemas locais e sistemas em nuvem.

O Servidor é um subsistema que possui diversos Scripts que são executados localmente, principalmente códigos de RPA. O mesmo servidor suporta diversas áreas, onde as execuções

são orquestradas para não interferirem umas nas outras.

A Unidade de Compartilhamento é um subsistema que é um serviço utilizado como um servidor de arquivos online. Possuem como componentes Bibliotecas de Documentos para armazenamento de arquivos relacionados a cada solução. Assim como no Banco de Dados, esse subsistema também possui funcionalidades de gerenciamento de acesso aos dados, no caso os documentos.

O Aplicativo é um subsistema que abrange uma diversa gama de funcionalidades e componentes. Entretanto, a quebra desses componentes ficou limitada apenas à documentação das telas do aplicativo, pois como já é utilizado uma ferramenta de *low code* não traria muito valor descer ainda mais o nível de detalhamento. Para o Aplicativo também são definidos tipos de perfis de acesso aos dados, e também a determinadas telas ou funcionalidades.

Os Agentes de IA são subsistemas que consultam Fontes de Dados como sites internos e externos, documentos e até Banco de Dados, e permitam que o usuário converse com um chat-bot do tipo RAG (Retrieval-Augmented Generation) que lhe responderá com base nas informações presentes nessas fontes selecionadas.

#### **4.2.2 Rastreabilidade**

Com as arquiteturas do sistema já definidas, cada elemento do sistema já possui suas funções e relacionamentos estabelecidos para acomodar as funcionalidades definidas para o sistema. Contudo, essa documentação pode não ser suficiente para definir de forma clara a interdependência dos elementos, visto que ela vai ser validada junto às áreas clientes e consequentemente tem um nível de detalhamento mais superficial.

Assim, é proposto a criação de um aplicativo para registrar essa dependência entre componentes e trazer a rastreabilidade do sistema a um nível palpável para o time de desenvolvimento. Tendo como modelo o estilo e padrões de arquitetura sugeridos, nas Figuras 4.10 e 4.11 vemos a proposta e conceito para esse aplicativo.

Nesse momento não há funcionalidades de Processos e Automações ou Documentos, o aplicativo é apenas uma forma de inserir e visualizar os dados de dependências entre os elementos.

O aplicativo é simples e consiste em três telas onde são cadastradas e visualizadas as soluções desenvolvidas, os elementos de cada solução e a dependência entre os elementos de uma mesma solução. Os perfis separados são importantes para deixar a administração e cadastros de dependências com os desenvolvedores do time, mantendo no entanto a visibilidade para os gerentes de projeto e equipe.



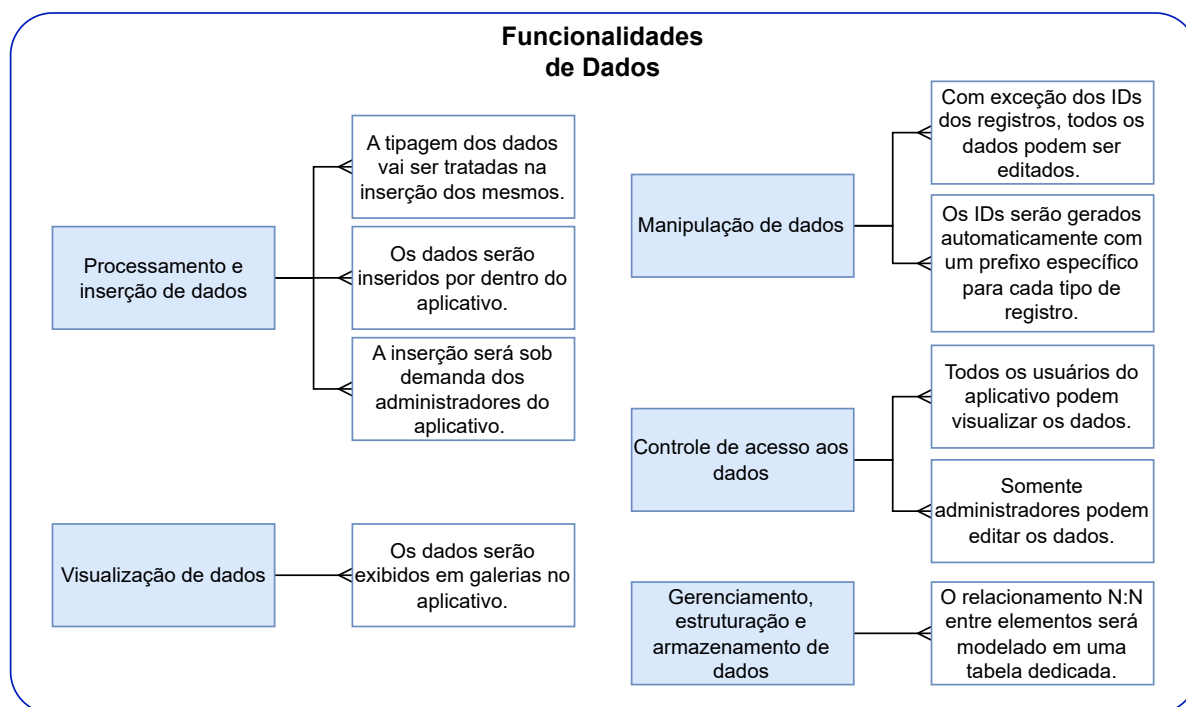


Figura 4.10: Arquitetura funcional do aplicativo proposto

## 4.3 Procedimentos para Coleta de Resultados

Com a intenção de trazer resultados quantitativos ao trabalho, são feitas comparações da percepção de esforço para a execução de tarefas em diferentes cenários e comparações com dados históricos de soluções desenvolvidas anteriormente pelo time. Além disso, uma avaliação do processo e dos ganhos nas relações com as áreas clientes, é realizada durante a rotina com as novas propostas sugeridas sendo aplicadas.

### 4.3.1 Comparação com Dados Históricos

Foi feito um levantamento das tarefas e esforços definidos anteriormente nas histórias de usuário para o desenvolvimentos dos elementos do sistema, separados em categorias, para as soluções já entregues e em produção, antes da aplicação das propostas apresentadas. Com isso, são definidas medidas estatísticas de Média de Pontos/Tarefas e Desvio Padrão de Pontos/Tarefas, onde cada tarefa recebeu pontos de esforço seguindo a sequência de Fibonacci.

A relação das histórias de usuário, é composta pelo identificador na coluna “ID”, seus dados de números de tarefas, os pontos de esforço totais extraídas da base histórica da equipe e foi adicionada ainda, na última coluna, a relação de “(Pontos)/(Número de Tarefas)” para ser utilizada nos cálculos das estatísticas a serem comparadas posteriormente com os novos dados a serem coletados.

As estatísticas foram calculadas por categoria do tipo de desenvolvimento. Foi escolhida essa métrica de (Pontos)/(Número de Tarefas) para a análise, pois se trata de uma comparação

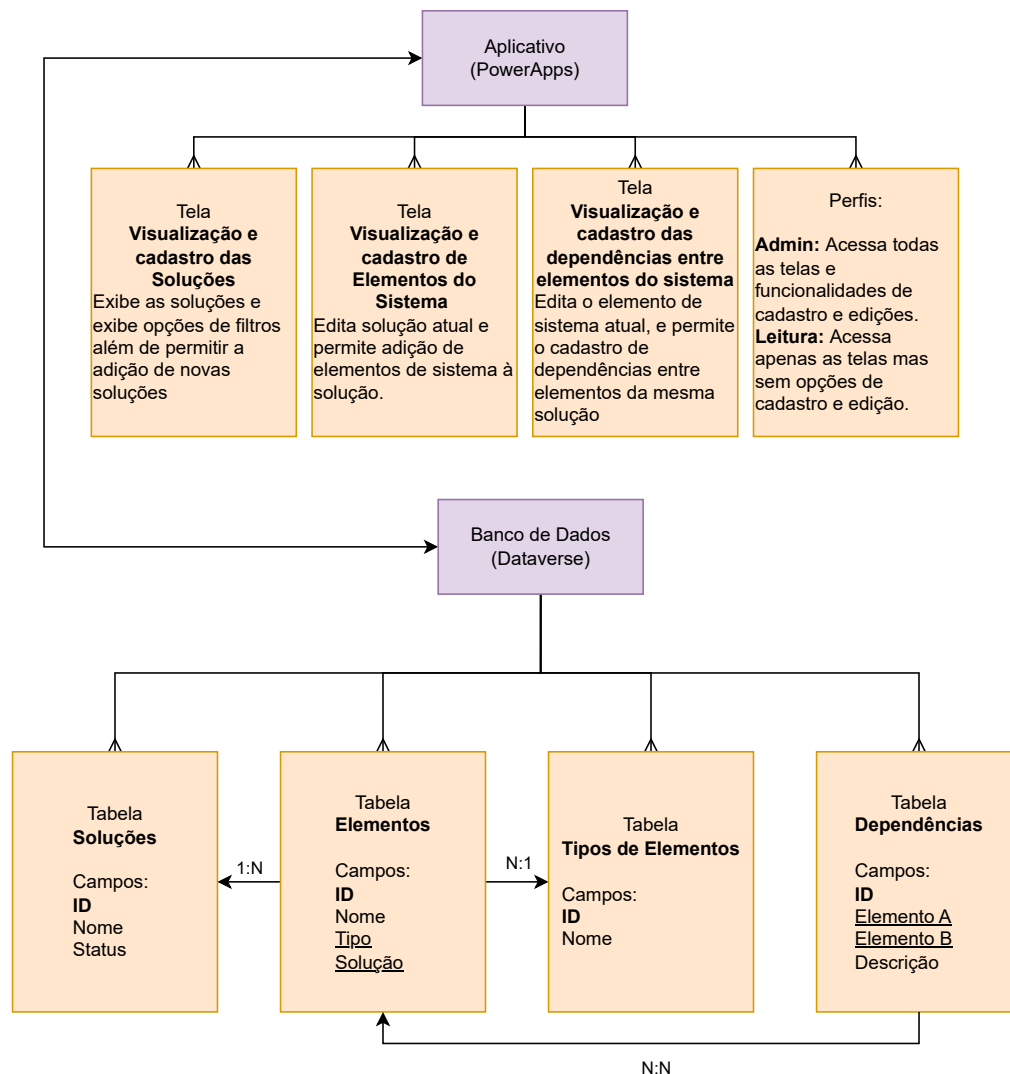


Figura 4.11: Arquitetura física do aplicativo proposto

que pode ser analisada mesmo entre categorias, e não é afetada pela complexidade do desenvolvimento, que seria o caso ao se analisar o número de tarefas ou o esforço total separadamente. Nesse caso é analisada a distribuição do esforço entre as tarefas, e como está esse comportamento em cada categoria.

Após o desenvolvimento de novas soluções aplicando as propostas de arquitetura para a definição e validação de conceito, novos dados coletados vão compor os resultados a serem analisados e comparados com esse levantamento histórico.

### 4.3.2 Avaliação de cenários de manutenção

Com o intuito de avaliar o impacto da utilização do aplicativo proposto para garantir a rastreabilidade e mapear as dependências entre os elementos do sistema, três cenários de falha ou melhoria são analisados no que diz respeito ao esforço necessário para executá-lo.

Para isso, soluções desenvolvidas no passado são cadastradas no aplicativo construído,

bem como seus elementos e as dependências entre eles. E os outros desenvolvedores da equipe fazem essa estimativa duas vezes, a primeira apenas com a contextualização do cenário proposto e a outra com a utilização do aplicativo para consultar as dependências envolvidas em cada cenário.

## 4.4 Instrumentos e Materiais

A documentação do ciclo de vida atual do sistema, os fluxogramas e diagramas apresentados nesse trabalho, bem como os padrões de arquitetura propostos foram confeccionados na ferramenta online *Draw.io*, que além de gratuita para o desenho de diagramas de diferentes tipos, não restringe o formato de salvamento do arquivo final, mantendo assim a alta qualidade dos diagramas com imagens vetorizadas.

A aplicação será desenvolvida com as ferramentas e recursos disponíveis na *Power Platform*. Para a interface de usuário será utilizado o *Power Apps* e para a criação do banco de dados será utilizado o *Dataaverse*.

Para a estimação de esforço das tarefas é utilizado uma extensão do aplicativo *Azure DevOps* que habilita a funcionalidade de *Planning Poker*, onde um desenvolvedor escolhe o esforço estimado sem ver a escolha dos outros, e depois todas as escolhas são reveladas simultaneamente, evitando assim a interferência na tomada de decisão alheia.

## Capítulo 5

### Resultados

O estudo e avaliação do processo de prestação do serviço nos moldes da Engenharia de Sistemas trouxe a clareza necessária para a proposição de sugestões de modificações no processo e a inclusão de ferramentas que auxiliam no gerenciamento dos sistemas desenvolvidos.

Seguindo os procedimentos para coleta de resultados, e aplicando as modificações no processo, foram gerados dados para avaliar o impacto do trabalho realizado, e comparar com medições previamente estabelecidas.

#### 5.1 Resultados para a criação de novas histórias de usuário

Na Tabela 5.1 vemos a relação das histórias de usuário, coletadas do histórico, precentes à aplicação das mudanças sugeridas.

ID	Solução	Categoria	Tarefas	Pontos	Pontos por Tarefa
169913	A	Automação	1	5	5.00
169967	B	Automação	2	7	3.50
169981	B	Automação	2	7	3.50
170102	B	Automação	1	5	5.00
169525	C	Automação	1	5	5.00
169516	C	Automação	1	5	5.00
169517	C	Automação	1	3	3.00
169515	C	Automação	1	3	3.00
169523	C	Automação	2	7	3.50
169521	C	Automação	1	3	3.00
169518	C	Automação	1	5	5.00
170090	E	Automação	2	8	4.00

*continua na próxima página*

ID	Solução	Categoria	Tarefas	Pontos	Pontos por Tarefa
170085	E	Automação	1	5	5.00
169821	E	Automação	2	6	3.00
169665	F	Automação	2	5	2.50
169671	F	Automação	2	10	5.00
169881	G	Automação	4	17	4.25
170032	H	Automação	1	5	5.00
170045	H	Automação	1	5	5.00
169281	I	Automação	1	8	8.00
169294	I	Automação	1	8	8.00
171835	I	Automação	4	21	5.25
169834	B	Banco de dados	1	8	8.00
169640	D	Banco de dados	3	6	2.00
170055	H	Banco de dados	1	3	3.00
169482	I	Banco de dados	5	15	3.00
169508	C	Relatório de dados	3	9	3.00
174691	E	Relatório de dados	3	8	2.67
169545	I	Relatório de dados	4	15	3.75
169699	A	Script	5	11	2.20
169731	A	Script	7	14	2.00
169941	A	Script	3	9	3.00
170128	H	Script	2	8	4.00
170073	A	Tela	3	5	1.67
169645	A	Tela	5	13	2.60
169539	A	Tela	2	5	2.50
170005	A	Tela	2	4	2.00
170120	B	Tela	1	5	5.00
169789	B	Tela	1	5	5.00
169778	B	Tela	2	8	4.00
169953	B	Tela	5	13	2.60
169556	D	Tela	5	12	2.40
169553	D	Tela	7	22	3.14
169639	D	Tela	5	10	2.00
170052	H	Tela	2	10	5.00
169293	I	Tela	2	10	5.00
169388	I	Tela	1	5	5.00
169612	I	Tela	1	5	5.00

*continua na próxima página*

ID	Solução	Categoria	Tarefas	Pontos	Pontos por Tarefa
169372	I	Tela	1	5	5.00
169354	I	Tela	1	8	8.00
169405	I	Tela	1	8	8.00
169546	I	Tela	7	32	4.57

Tabela 5.1: Distribuição de esforço por solução e categoria antes das mudanças sugeridas

Na Tabela 5.2, estão as estatísticas calculadas por categoria do tipo de desenvolvimento.

Categoria	Média de Pontos/Tarefas	Desvio Padrão de Pontos/Tarefas
Automação	4.52	1.44
Banco de dados	4.00	2.71
Relatório de dados	3.14	0.55
Script	2.80	0.91
Tela	4.13	1.85

Tabela 5.2: Média e desvio padrão dos pontos de esforço por tarefa, por categoria, antes de aplicar as mudanças sugeridas

Ao serem confeccionadas as arquiteturas funcional e física para as novas soluções foram obtidos artefatos de grande importância para as fases de **Desenvolvimento Avançado e Projeto de Engenharia**. No caso da arquitetura física, foi de grande valia para a criação das histórias de usuário e tarefas a serem executadas. Nas Figuras 5.1 e 5.2 vemos as arquiteturas físicas para dois sistemas desenvolvidos seguindo a nova metodologia, as caixas de e-mail foram censuradas para preservar o sigilo.

Baseado nessas arquiteturas, foram criadas as tarefas de cada história de usuário e definido os pontos de esforço. Na Tabela 5.3 foi incluído também o aplicativo construído para a rastreabilidade como a Solução Z.

ID	Solução	Categoria	Tarefas	Pontos	Pontos por Tarefa
178690	X	Automação	10	10	1.00
178750	X	Automação	10	10	1.00
178766	X	Automação	10	10	1.00
178830	X	Automação	3	5	1.67
179009	Y	Automação	2	3	1.50
179010	Y	Automação	2	5	2.50
179011	Y	Automação	2	2	1.00

*continua na próxima página*

ID	Solução	Categoria	Tarefas	Pontos	Pontos por Tarefa
178707	X	Banco de dados	7	13	1.86
179005	Y	Banco de dados	12	16	1.33
179020	Z	Banco de dados	4	6	1.50
178825	X	Relatório de dados	5	15	3.00
179006	Y	Relatório de dados	8	18	2.25
179007	Y	Script	3	8	2.67
179008	Y	Script	3	8	2.67
178784	X	Tela	9	16	1.78
178795	X	Tela	9	19	2.11
178806	X	Tela	5	7	1.40
178814	X	Tela	9	17	1.89
179001	Y	Tela	4	12	3.00
179002	Y	Tela	5	12	2.40
179003	Y	Tela	6	13	2.17
179004	Y	Tela	4	5	1.25
179021	Z	Tela	3	5	1.67
179022	Z	Tela	2	4	2.00
179023	Z	Tela	3	5	1.67

Tabela 5.3: Distribuição de esforço por solução e categoria após as mudanças sugeridas

Categoria	Média de Pontos/Tarefas	Desvio Padrão de Pontos/Tarefas
Automação	1.38	0.57
Banco de dados	1.56	0.07
Relatório de dados	2.63	0.28
Script	2.67	0.00
Tela	2.00	0.51

Tabela 5.4: Média e variância dos pontos de esforço por categoria após aplicação das mudanças sugeridas

Com o intuito de facilitar a comparação, os resultados das Tabelas 5.2 e 5.4 foram representados graficamente na Figura 5.3.

É notável ao observar a Figura 5.3 que houve uma queda nas médias de pontos por tarefas em todas as categorias de desenvolvimento ao se comparar com as mesmas estatísticas dos dados históricos. Nota-se também uma queda no desvio padrão de cada categoria, no caso da categoria Script, observa-se um valor nulo, pois as duas atividades de desenvolvimento nessa categoria possuíam os mesmos números de tarefas e esforço, e como são, ainda por cima, da mesma solução, não trazem consigo uma representatividade suficiente para serem levados em

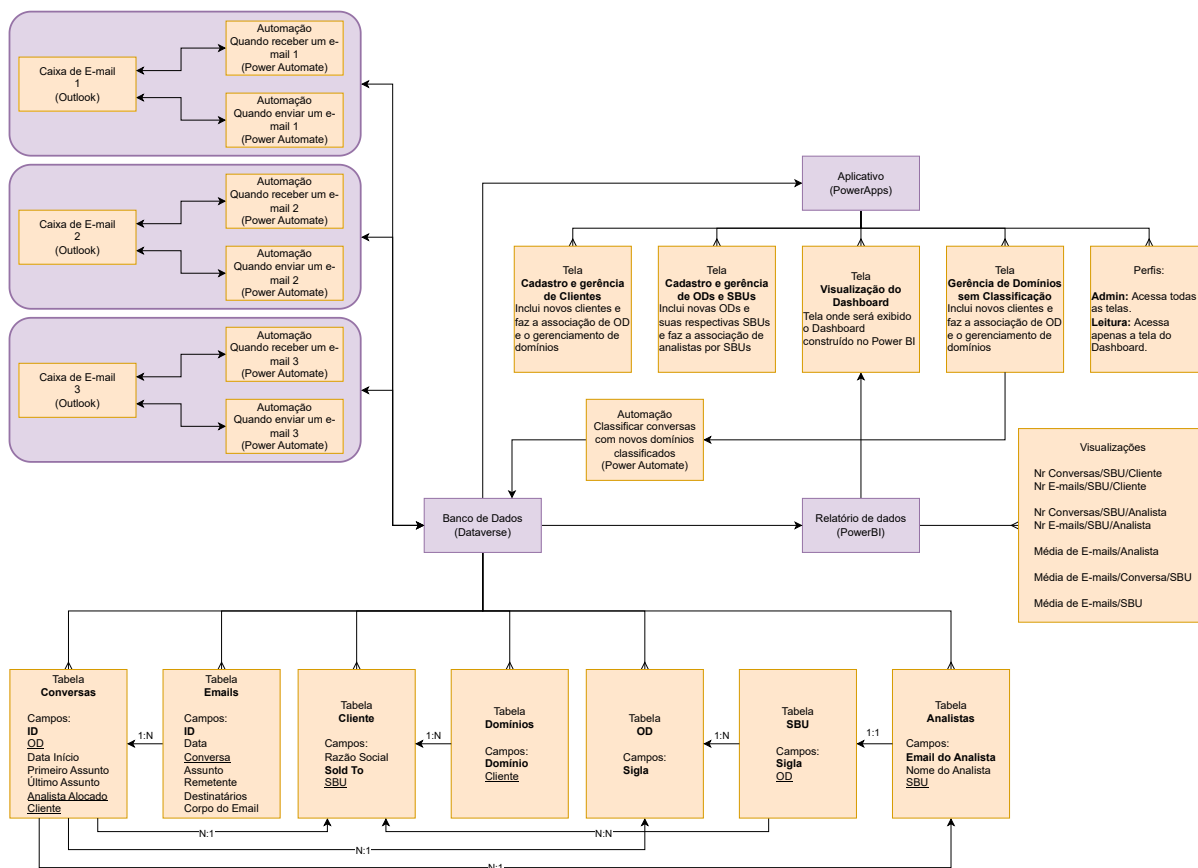


Figura 5.1: Arquitetura física da Solução X

consideração como um ganho.

Para as outras categorias, no entanto, esses valores representam uma distribuição mais uniforme de esforço por tarefa e um nível de detalhamento maior. Esse resultado anda paralelo a um melhor entendimento do serviço a ser executado.

Nas reuniões para validação da solução com as áreas clientes, ou simplesmente a validação de conceito, a arquitetura física foi de grande ajuda para o engajamento do pessoal. As partes interessadas foram capazes de opinar e sugerir alterações, visto que o nível de detalhamento apresentado não chegou a um nível inaccessível por pessoas fora da área técnica. Além disso, ela funciona como um documento comprobatório do escopo do projeto, evitando a inclusão de novas funcionalidades fora do escopo, que não estão acomodadas no sistema projetado e definido para o início do estágio de Desenvolvimento.

Além dos dados numéricos apresentados, vale ressaltar que com a documentação das arquiteturas funcionais e físicas, foi viabilizado a utilização do auxílio do agente de IA autônomo (*Microsoft Copilot*) para a criação das histórias de usuários. Ao serem enviados os documentos das arquiteturas com um prompt contextualizando sobre a solução e a estrutura e ferramentas de trabalho da equipe, o agente foi capaz de gerar as histórias de usuário e suas tarefas com bastante exatidão, e poucas alterações durante a revisão das mesmas, basicamente sendo necessária apenas a exclusão de tarefas desnecessárias, a unificação de tarefas pequenas



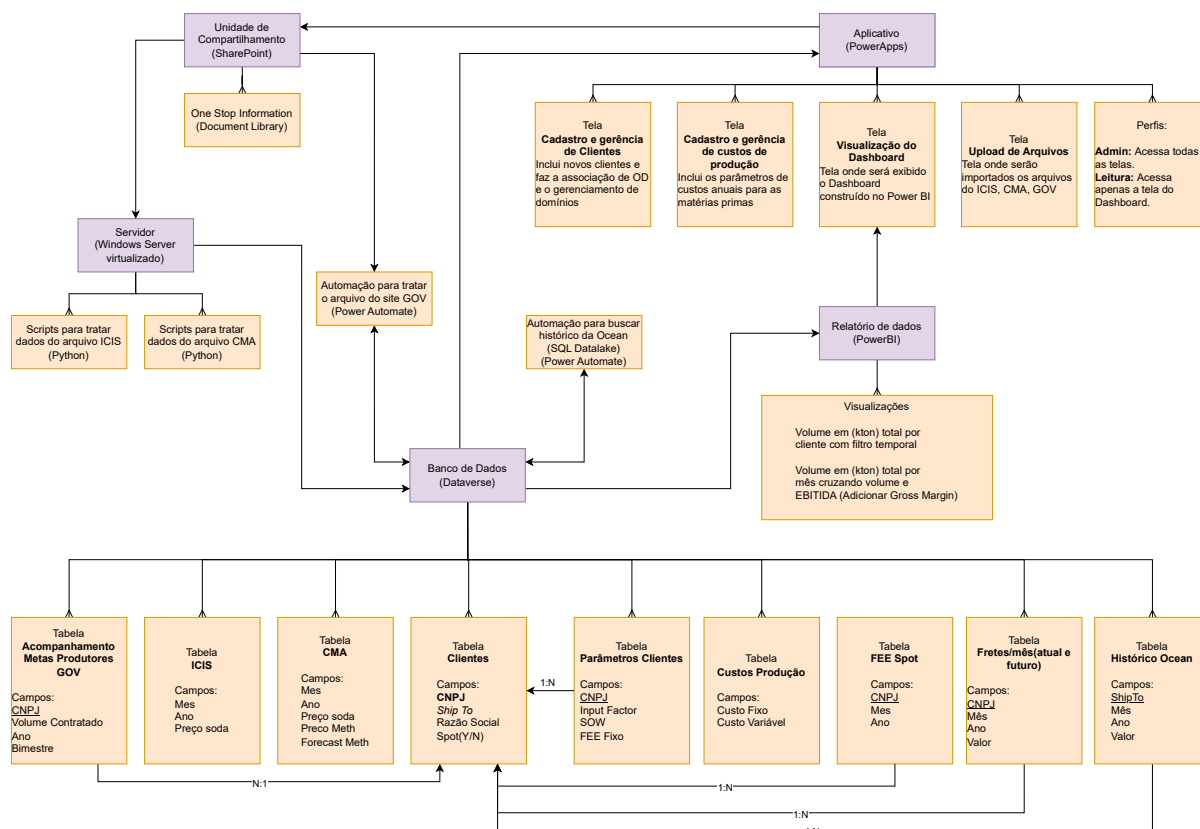


Figura 5.2: Arquitetura física da Solução Y

e muito relacionadas, ou a inclusão de poucas tarefas não consideradas.

Solução	Tarefas Sugeridas	Tarefas Utilizadas	Diferença percentual
X	92	77	-16.3%
Y	44	51	13.7%
Z	12	12	0%

Tabela 5.5: Quantidade de tarefas geradas pelo agente de IA, a quantidade realmente utilizada e a Diferença percentual

É visto um bom aproveitamento das tarefas sugeridas, e nota-se que para soluções menos complexas o agente autônomo foi mais assertivo nas tarefas a serem executadas. Esse tipo de ação e colaboração com IA trás uma economia de tempo e recurso muito significativa, onde antes o desenvolvedor deveria criar toda a estrutura, e agora apenas revisa e faz pequenas alterações, sobrando então mais tempo para atividades de desenvolvimento de fato.

## 5.2 Resultados para Rastreabilidade

A proposta de criar uma aplicação para garantir a rastreabilidade do sistema traz valor em fases mais avançadas do ciclo de vida, em especial as fases de utilização e suporte. Caso

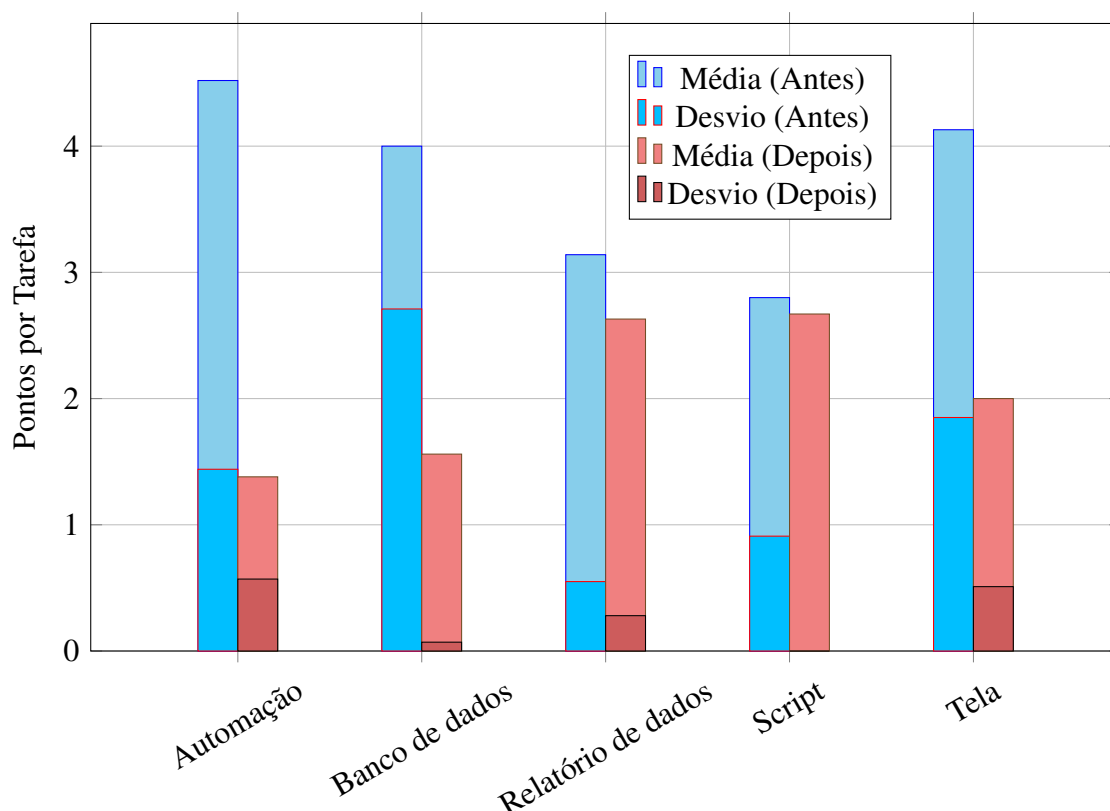


Figura 5.3: Comparação agrupada de média e desvio padrão por categoria, antes e depois das mudanças

surja alguma melhoria ou manutenção nessas etapas, o custo envolvido nessas mudanças é maior que se fossem feitas durante a fase de produção, no caso o custo é o equivalente ao esforço empenhado nessas mudanças. Então, saber exatamente o impacto dessas alterações no sistema se torna crucial para uma ação consciente e responsável.

O aplicativo construído pode ser observado nas Figuras 5.4 a 5.10.

Rastreabilidade 		
Soluções		
<div>Pesquise pelo nome da solução</div> <div>Adicionar</div>		
S1000	Solução X	Em Desenvolvimento
S1001	Solução Y	Em Produção (Interno)
S1002	Solução W	Em Inspeção de Qualidade
S1003	App Rastreabilidade	Em Produção (Interno)
S1004	Measurement Report	Em Produção (Interno)
S1005	Central de Faturas	Em Produção (Interno)

Figura 5.4: Tela de visualização das soluções.

Como sugerido, foram propostos três cenários para avaliação de esforço de atuação:

- **Cenário 1:** O fluxo de automação que envia o resultado da análise das notas fiscais para

**Rastreabilidade**

**Soluções**

Pesquise pelo nome da solução

Nome da solução Status

Salvar Cancelar

S1000	Solução X	Em Desenvolvimento
S1001	Solução Y	Em Produção (Interno)
S1002	Solução W	Em Inspeção de Qualidade
S1003	App Rastreabilidade	Em Produção (Interno)
S1004	Measurement Report	Em Produção (Interno)
S1005	Central de Faturas	Em Produção (Interno)

Figura 5.5: Tela de visualização das soluções com a expansão dos campos para adicionar nova solução.

**Rastreabilidade**

**Solução X** **S1000** Em Desenvolvimento

Elementos do sistema

Tipo de elemento

Pesquise pelo nome do elemento

Adicionar

ES1000	Elemento X1	Power Apps - Model Driven App
ES1001	Elemento X2	Script Python
ES1002	Elemento X3	Power Automate
ES1004	Elemento X4	Agente Copilot
ES1005	Elemento X5	Power Apps - Canvas App
ES1007	Elemento X6	Microsoft Forms
ES1008	Elemento X7	Tela

Figura 5.6: Tela de visualização dos elementos.

**Rastreabilidade**

**Solução X** **S1000** Em Desenvolvimento

Elementos do sistema

Power Automate Tela

Pesquise pelo nome do elemento

Nome do elemento do sistema Seleccione o tipo do elemento

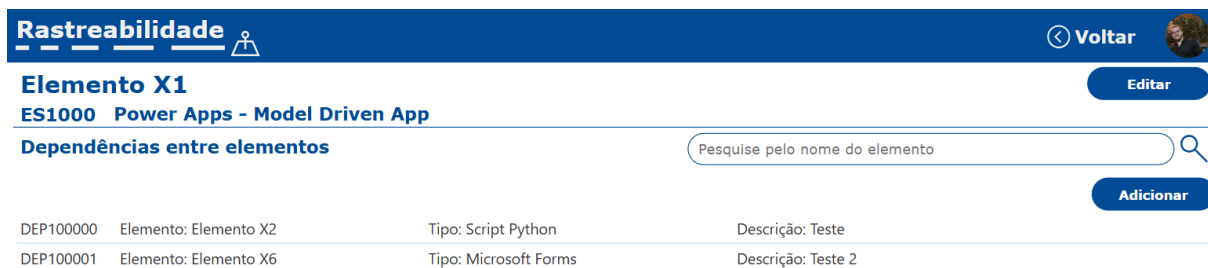
Salvar Cancelar

ES1002	Elemento X3	Power Automate
ES1008	Elemento X7	Tela

Figura 5.7: Tela de visualização dos elementos com a expansão dos campos para adicionar novo elemento e editar dados da solução atual.

o fornecedor e as notas aprovadas para pagamento, foi deletado por engano e precisa ser feito novamente.

- **Cenário 2:** A caixa compartilhada de e-mail utilizada para as notificações do sistema precisa ser substituída por outra.



**Rastreabilidade**

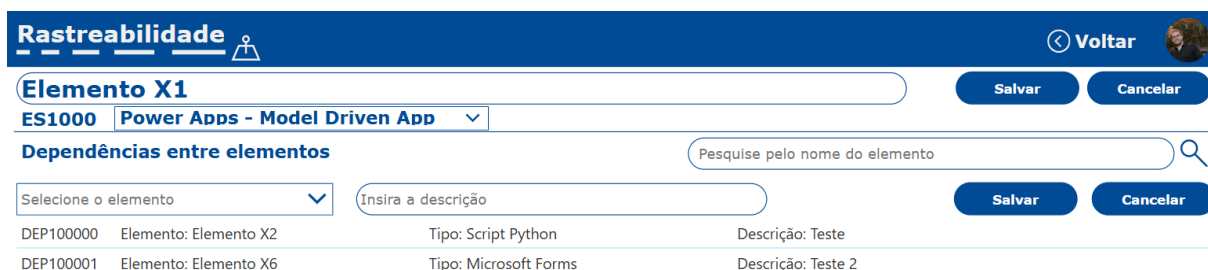
**Elemento X1** Editar

**ES1000 Power Apps - Model Driven App**

**Dependências entre elementos**  Adicionar

DEP100000	Elemento: Elemento X2	Tipo: Script Python	Descrição: Teste
DEP100001	Elemento: Elemento X6	Tipo: Microsoft Forms	Descrição: Teste 2

Figura 5.8: Tela de visualização das dependências.



**Rastreabilidade**

**Elemento X1** Salvar Cancelar

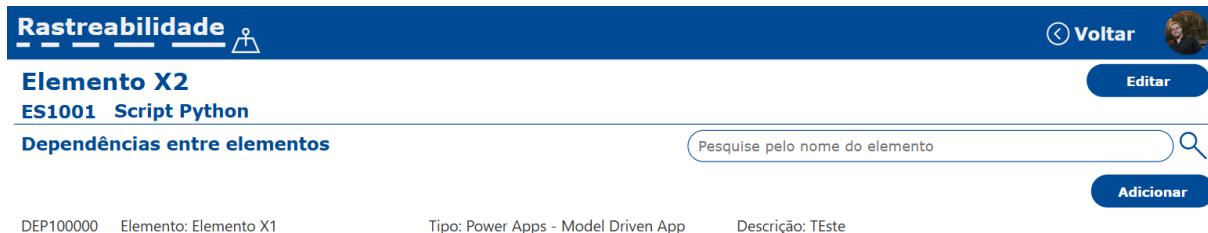
**ES1000 Power Apps - Model Driven App**

**Dependências entre elementos**  Salvar Cancelar

Selecione o elemento

DEP100000	Elemento: Elemento X2	Tipo: Script Python	Descrição: Teste
DEP100001	Elemento: Elemento X6	Tipo: Microsoft Forms	Descrição: Teste 2

Figura 5.9: Tela de visualização das dependências com a expansão dos campos para adicionar nova dependência e editar o elemento atual



**Rastreabilidade**

**Elemento X2** Editar

**ES1001 Script Python**

**Dependências entre elementos**  Adicionar

DEP100000	Elemento: Elemento X1	Tipo: Power Apps - Model Driven App	Descrição: TEste
-----------	-----------------------	-------------------------------------	------------------

Figura 5.10: Tela de visualização evidenciando a dependência cruzada.

- **Cenário 3:** A inclusão de uma coluna com dado importante em uma tabela que deve ser exibida junto com as outras informações provenientes dessa tabela, seja no e-mail ou no próprio aplicativo.

As imagens 5.11, 5.12 e 5.13 mostram as telas do aplicativo que foram mostradas para cada cenário antes da reavaliação pelo desenvolvedores.

Na Tabela 5.6 podem ser vistos os valores atribuídos a cada cenário antes e depois da apresentação do aplicativo.

Foi feito ainda um gráfico mostrado na Figura 5.14, que facilita essa comparação.

Apesar de indicado o desvio padrão para as medidas, não pode ser colocado uma grande importância nessa comparação visto que foram poucas amostragens de esforço devido ao pequeno número de desenvolvedores e cenários. No entanto, ao olharmos o gráfico da Figura

Rastreabilidade

Voltar

Send Reviewed Response

ES1021 Power Automate

Editar

Dependências entre elementos


Pesquise pelo nome do elemento

Adicionar


DEP100027	Elemento: Solicitações	Tipo: Dataverse - Tabela	Descrição: Leitura das informações do protocolo
DEP100011	Elemento: Central de Faturas - Managem...	Tipo: Power Apps - Canvas App	Descrição: Fluxo que notifica o fornecedor do resultado da análise
DEP100025	Elemento: Visualização e aprovação do p...	Tipo: Tela	Descrição: Acionamento do fluxo
DEP100026	Elemento: Anexos	Tipo: Dataverse - Tabela	Descrição: Leitura das informações das notas
DEP100028	Elemento: <div></div> ...	Tipo: Caixa de E-mail	Descrição: Caixa de e-mail para comunicação
DEP100029	Elemento: Notificações	Tipo: Dataverse - Tabela	Descrição: Leitura dos motivos e templates HTML para o e-mail en...
DEP100030	Elemento: Evidências	Tipo: Dataverse - Tabela	Descrição: Leitura das evidências de reprovação

Figura 5.11: Dependências relacionadas ao Cenário 1

Rastreabilidade



< Voltar




Editar

C[REDACTED]S@[REDACTED]

ES1030 Caixa de E-mail

Dependências entre elementos

Pesquise pelo nome do elemento



Adicionar

DEP100032	Elemento: Attachments E-mail DEV	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100033	Elemento: Attachments E-mail QUAL	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100035	Elemento: Requester Response DEV	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100020	Elemento: Export Data	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100028	Elemento: Send Reviewed Response	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100031	Elemento: Attachments E-mail PROD	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100034	Elemento: Requester Response PROD	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação
DEP100036	Elemento: Requester Response QUAL	Tipo: Power Automate	Descrição: Caixa de e-mail para comunicação

Figura 5.12: Dependências relacionadas ao Cenário 2

5.14 sugere-se que com mais informações a respeito das dependências houve um aumento desse desvio padrão em uma visão macro, o que é de certa forma esperado, pois com menos detalhes não muito questionamento e já quando se abre todas as relações envolvidas ao cenário, cada desenvolvedor pode identificar desafios que fujam em diferentes graus de suas especialidades.

Agora ao olhar para a média das avaliações, ou mesmo individualmente por desenvolvedor, após a apresentação do aplicativo, na reavaliação, os valores sobem em quase todos os

Rastreabilidade			
Solicitações			<a href="#">Voltar</a>
ES1041 Dataverse - Tabela			<a href="#">Editar</a>
Dependências entre elementos			<input type="text" value="Pesquise pelo nome do elemento"/>
			<a href="#">Adicionar</a>
DEP100027	Elemento: Send Reviewed Response	Tipo: Power Automate	Descrição: Leitura das informações do protocolo
DEP100045	Elemento: Clean Repeval Evidence	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos
DEP100046	Elemento: Requester Response PROD	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos
DEP100016	Elemento: Central de Faturas - Managem...	Tipo: Power Apps - Canvas App	Descrição: Leitura dos protocolos com as informações de envio
DEP100022	Elemento: Export Data	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos recebidos
DEP100037	Elemento: Central de Faturas de Serviços...	Tipo: Microsoft Forms	Descrição: Escrita das informações dos protocolos
DEP100039	Elemento: Central de Faturas de Serviços...	Tipo: Microsoft Forms	Descrição: Escrita das informações dos protocolos
DEP100041	Elemento: Requester Response QUAL	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos
DEP100038	Elemento: Central de Faturas de Serviços...	Tipo: Microsoft Forms	Descrição: Escrita das informações dos protocolos
DEP100040	Elemento: Detalhes e logs do protocolo ...	Tipo: Tela	Descrição: Leitura das informações dos protocolos
DEP100042	Elemento: Attachments E-mail PROD	Tipo: Power Automate	Descrição: Leitura e escrita das informações dos protocolos
DEP100043	Elemento: Attachments E-mail DEV	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos
DEP100044	Elemento: Attachments E-mail QUAL	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos
DEP100047	Elemento: Requester Response DEV	Tipo: Power Automate	Descrição: Leitura das informações dos protocolos
DEP100048	Elemento: Visualização e aprovação do p...	Tipo: Tela	Descrição: Leitura das informações dos protocolos
DEP100049	Elemento: Anexos	Tipo: Dataverse - Tabela	Descrição: Relacionamento entre protocolos e notas de serviço

Figura 5.13: Dependências relacionadas ao Cenário 3

Desenvolvedor	Cenário 1		Cenário 2		Cenário 3	
	Antes	Depois	Antes	Depois	Antes	Depois
1	5	5	1	2	3	5
2	8	8	1	3	3	8
3	5	8	1	1	3	5
Média	6	7	1	2	3	6
Desvio Padrão	1.73	1.73	0.00	1.00	0.00	1.73

Tabela 5.6: Comparativo do esforço alocado em três cenários, antes e depois do uso do aplicativo

casos individuais e em todos os casos de média.

Isso mostra um aumento na percepção de esforço a ser empenhado quando se tem clareza das dependências envolvidas na ação que se pretende realizar.

## 5.3 Consolidação dos Resultados

Consolidando então o que foi observado nesse capítulo, podem ser destacados quatro pontos:

- A criação de histórias de usuários para o desenvolvimento das soluções apresentou uma distribuição de esforço por tarefa mais homogênea e com menor concentração de esforço para uma única tarefa.
- O tempo e maneira de criação da histórias de usuário foi otimizado com o auxílio de IA,

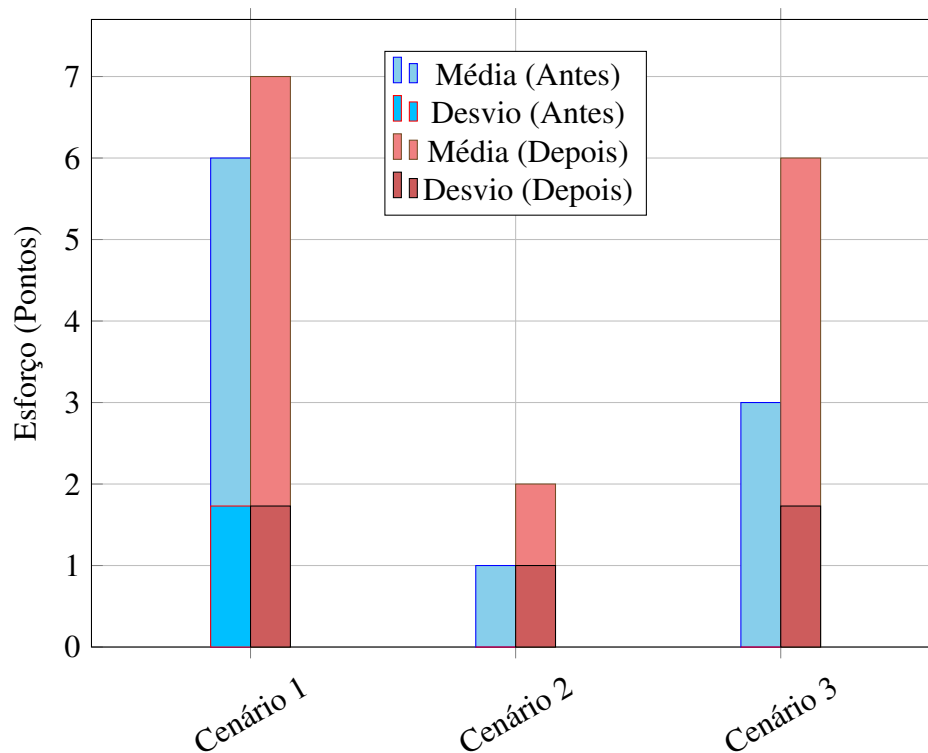


Figura 5.14: Comparativo da média e desvio padrão de esforço entre os três cenários, antes e depois da análise pelo aplicativo

agora que há a devida documentação do sistema.

- A relação com área cliente ficou mais transparente assim como as expectativas da entrega do produto, com uma validação mais eficiente antes de se iniciar de fato o desenvolvimento.
- A rastreabilidade de dependências aumentou a média de estimativa de esforço para os cenários propostos.

## Capítulo 6

### Conclusão

Os conceitos e boas práticas da Engenharia de Sistemas (ES) são versáteis e aplicáveis em diferentes tipos de ambientes e organizações. O trabalho realizado demonstra que, mesmo em cenários bastante específicos e fora dos padrões tradicionais, há ganhos significativos ao analisar o ciclo de vida sob a ótica da ES. Essa análise possibilita uma tomada de decisão mais embasada e assertiva nas mudanças sugeridas, que, por sua vez, mostraram-se eficazes na redução de problemas previamente identificados — especialmente na definição e validação de conceito e na rastreabilidade dos componentes do sistema.

A abordagem de revisar e buscar melhorias no ciclo de vida de um serviço de software por meio das práticas da ES reforça a diversidade de aplicações possíveis, extrapolando os limites da indústria tradicional e dos projetos de engenharia clássicos. O próprio INCOSE (2023) e a obra de Kossiakoff et al. (2020) apresentam exemplos de aplicação da ES em domínios como segurança cibernética e inteligência artificial.

A autonomia concedida para a implementação das modificações no ciclo de vida foi crucial para o êxito do trabalho. Sem isso, não teria sido possível a análise de dados reais da equipe. Ser o membro com maior senioridade no time técnico, aliado ao número reduzido de pessoas na equipe, facilitou a implantação das mudanças e a aceitação por parte dos demais membros — algo que poderia representar um desafio maior em equipes mais numerosas ou com profissionais mais experientes já acostumados com o processo anterior, mesmo que este apresentasse limitações. Vale destacar que o novo ciclo de vida foi aplicado em apenas dois sistemas reais voltados para a área de clientes, além de seu uso no próprio aplicativo desenvolvido neste trabalho. No entanto, nenhum dos projetos havia sido concluído até o momento, o que limita a maturidade dos dados disponíveis para uma avaliação mais ampla e representativa do serviço de criação de sistemas.

Os resultados obtidos reforçam a importância de um ciclo de vida bem estruturado e, especialmente, das fases de Definição de Conceito e Desenvolvimento Avançado — cruciais em projetos com escopo fechado. No caso de sistemas com prazos de desenvolvimento atipicamente curtos, como os analisados, essas fases tornam-se ainda mais críticas, já que qualquer erro na estimativa de esforço representa um impacto proporcionalmente elevado no



prazo final de entrega. A melhoria na distribuição do esforço por tarefa, observada neste trabalho, contribui para a redução desses erros ao aumentar a granularidade das ações a serem executadas. Isso também gera maior visibilidade para o gerente de projetos, permitindo um gerenciamento de riscos mais preciso. Além disso, a nova estrutura operacional possibilitou a introdução de ferramentas de inteligência artificial na rotina da equipe, alinhando-se às tendências de mercado voltadas à automação de tarefas repetitivas.

A ferramenta desenvolvida para rastreabilidade do sistema demonstrou ser um recurso eficaz para melhorar a acurácia na estimativa de esforço. O aumento dessa estimativa após sua adoção indica a redução de percepções excessivamente otimistas quanto ao esforço requerido por tarefas consideradas simples. Isso evita a alocação indevida de múltiplas tarefas simultâneas para um único recurso, contribuindo para uma gestão de carga de trabalho mais realista. Ademais, a rastreabilidade melhora o planejamento técnico para manutenções e inclusão de melhorias em sistemas existentes, apoiando diretamente o gerenciamento de riscos técnicos.

Como sugestões para trabalhos futuros, destaca-se a necessidade de analisar o impacto das mudanças no prazo efetivo de conclusão dos projetos. Outra oportunidade de evolução está na ampliação da hierarquia de elementos dentro do aplicativo de rastreabilidade, incluindo funcionalidades e processos de mais alto nível, oferecendo uma visão ainda mais integrada do sistema. A coleta de uma base de dados maior e mais variada também seria valiosa, especialmente para aprofundar a análise do esforço relacionado à rastreabilidade em múltiplos cenários.

Por fim, concluímos que estudos sobre aplicações não convencionais da Engenharia de Sistemas, como o realizado neste trabalho, são fundamentais para a identificação de novos ganhos e oportunidades de adaptação metodológica. Esta pesquisa contribui para o corpo de conhecimento da área ao demonstrar, de forma concreta, como os métodos e processos da ES podem ser aplicados com precisão na identificação de falhas no ciclo de vida e nas fases em que ocorrem, servindo de base para modificações eficazes de processos. Trata-se de uma demonstração prática do valor da Engenharia de Sistemas como ferramenta de transformação organizacional, inclusive em domínios que não são tradicionalmente associados à engenharia clássica.

## Referências Bibliográficas

- INCOSE (2023). INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities. International Council on Systems Engineering (INCOSE), San Diego, CA, 5th edition.
- Kossiakoff, A., Biemer, S. M., Seymour, S. J. & Flanagan, D. A. (2020). Systems Engineering: Principles and Practice. Wiley Series in Systems Engineering and Management. Wiley, Hoboken, NJ, 3rd edition.
- McKinsey Global Institute (2017). A future that works: Automation, employment, and productivity. <https://www.mckinsey.com/featured-insights/digital-disruption/harnessing-automation-for-a-future-that-works>.
- Pańkowska, M. (2024). Low code development cycle investigation. In Yang, X.-S., Sherratt, S., Dey, N. & Joshi, A., editors, Proceedings of Ninth International Congress on Information and Communication Technology, pages 265–275, Singapore. Springer Nature Singapore.
- Rokis, K. & Kirikova, M. (2023). Exploring low-code development: A comprehensive literature review. page 68–86.
- SEBoK Editorial Board (2024). The Guide to the Systems Engineering Body of Knowledge (SEBoK). The Trustees of the Stevens Institute of Technology, 2.11 edition. Acesso em: 14 jan. 2025. [www.sebokwiki.org](http://www.sebokwiki.org).
- Sebrae (2023). Como a digitalização de processos impacta os resultados da empresa. Acesso em: 14 jan. 2025.