UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

CENTRO TECNOLÓGICO

COLEGIADO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Matheus De Oliveira Lima

Título do Trabalho

Vitória, ES

2025

Matheus De Oliveira Lima

Título do Trabalho

Anteprojeto apresentado ao Colegiado do Curso de Ciência da Computação do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito para aprovação na Disciplina Projeto de Graduação I.

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Colegiado do Curso de Ciência da Computação

Orientador: Prof. Dr. Vítor Estêvão Silva Souza

Vitória, ES

2025

1 Introdução

O Sistema Marvin surgiu como uma iniciativa para ajudar a gestão acadêmica e administrativa da Universidade Federal do Espírito Santo (UFES), complementando os sistemas institucionais existentes. Desenvolvido originalmente em 2019 como um projeto de extensão, o Marvin oferece suporte a demandas específicas de departamentos cursos e programas de pós-graduação - como acompanhamento de egressos.

Este trabalho tem como foco melhorar a interface do Sistema Marvin que foi criado para ajudar na gestão administrativa da faculdade, diferenciando o padrão visual de acordo com os diferentes papéis de usuários (como professores, coordenadores, alunos, entre outros).

1.1 Motivação e Justificativa

A escolha deste tema foi motivada pela identificação de uma oportunidade de aplicar conhecimentos sobre desenvolvimento web em um contexto real e relevante para a universidade. O desenvolvimento de interfaces personalizadas representa um desafio técnico interessante que permite explorar conceitos de engenharia de software, design de interfaces e experiência do usuário. Além disso, a possibilidade de contribuir para o aprimoramento de uma ferramenta com potencial de uso efetivo dentro da universidade adiciona um valor ao projeto, permitindo que os conhecimentos adquiridos durante a graduação sejam aplicados gerando um impacto positivo na comunidade acadêmica.

A justificativa para esse trabalho está na necessidade de melhorar a experiência dos usuários do Sistema Marvin. Atualmente, ele funciona com páginas baseadas em operações CRUD (Create, Read, Update, Delete - criar, ler, atualizar e excluir), sem levar em conta o papel específico que cada usuário exerce dentro da instituição. Isso acaba tornando o uso do sistema mais confuso e pouco eficiente, especialmente para quem não tem familiaridade com o funcionamento interno das atividades administrativas e acadêmicas. Além disso, embora o sistema já implemente diferentes níveis de permissão para cada tipo de usuário, as interfaces não refletem adequadamente essas distinções, dificultando a navegação e o acesso às funcionalidades relevantes para cada perfil.

Desenvolver interfaces personalizadas, que se adaptem ao perfil de cada usuário, pode aumentar a usabilidade do sistema e agilizar tarefas rotineiras. O trabalho de (CHANE, 2023) modelou o sistema pensando tanto no acesso do administrador através das operações CRUD quanto nos acessos específicos de cada perfil de usuário. Porém, foram implementadas apenas as funcionalidades voltadas para o administrador. Por isso, é

relevante continuar desenvolvendo soluções que não apenas funcionem tecnicamente, mas que também sejam intuitivas e adequadas ao contexto real de uso de cada perfil dentro da universidade.

1.2 Objetivos

O **objetivo geral** é desenvolver interfaces personalizadas no módulo administrativo do sistema Marvin, adaptando as páginas de acordo com o papel (Role) de cada usuário, com o objetivo de tornar a navegação mais intuitiva e funcional, bem como dar acesso a esses perfis às funcionalidades planejadas no trabalho do (CHANE, 2023), mas não implementadas.

1.3 Método de Desenvolvimento do Trabalho

O desenvolvimento deste trabalho será feito em etapas, seguindo uma abordagem prática baseada no processo de engenharia de software, com foco em desenvolvimento web. A metodologia utilizada envolve análise dos requisitos, análise dos papéis dos usuários, planejamento das interfaces, implementação e validação.

Inicialmente, será feita um estudo da documentação de requisitos do (CHANE, 2023), com foco no subsistema administrativo, para entender como os diferentes tipos de usuários (como administradores, gestores de departamento, coordenadores de curso e programas de pós-graduação) interagem com as funcionalidades existentes. A partir disso, serão identificadas as principais funções associadas a cada papel e os pontos em que a interface atual não atende bem às necessidades específicas desses perfis.

Com base nessa análise, serão projetadas novas interfaces personalizadas, utilizando ferramentas modernas de desenvolvimento frontend. A implementação será feita por meio de tecnologias web já utilizadas no projeto Marvin, respeitando a estrutura já existente e buscando garantir compatibilidade com o sistema atual.

Por fim, será realizada uma etapa de testes com usuários reais ou simulações de uso, para avaliar a usabilidade das novas interfaces. A documentação do processo completo, incluindo decisões técnicas e dificuldades encontradas, também fará parte do trabalho final.

1.4 Cronograma

O cronograma de execução apresenta a distribuição no tempo das atividades que serão realizadas ao longo do desenvolvimento do trabalho. As atividades foram definidas para garantir que cada uma delas contribua diretamente para a obtenção do objetivo geral.

- **Atividade 1: Estudo de requisitos**
  Estudar os requisitos do (CHANE, 2023), especialmente o subsistema administrativo, e identificar os diferentes tipos de usuários e suas respectivas funções.

- **Atividade 2: Avaliação da usabilidade atual**
  Estudar as interfaces existentes no sistema e levantar as principais dificuldades enfrentadas por cada perfil de usuário.

- **Atividade 3: Projeto das interfaces personalizadas**
  Definir a estrutura, layout e fluxo de navegação das novas interfaces específicas para cada papel.

- **Atividade 4: Implementação das interfaces - Parte 1**
  Desenvolver e integrar as metade das interfaces personalizadas.

- **Atividade 5: Implementação das interfaces - Parte 2**
  Finalizar o restante das interfaces dos demais papéis.

- **Atividade 6: Testes, validação e documentação**
  Realizar testes com usuários (ou simulações), ajustar problemas encontrados e documentar todo o processo de desenvolvimento.

Tabela 1 – Cronograma de execução das atividades

	Mês 1	**Mês 2**	**Mês 3**	**Mês 4**	**Mês 5**	**Mês 6**
Atividade 1	X					
Atividade 2		X				
Atividade 3			X			
Atividade 4			X	X		
Atividade 5				X	X	
Atividade 6						X

2 Fundamentação Teórica e Tecnologias Utilizadas

A fundamentação teórica deste trabalho aborda conceitos essenciais para o desenvolvimento de interfaces personalizadas em sistemas web, com foco na experiência do usuário (UX) e na adaptação de funcionalidades conforme o papel de cada usuário. Além disso, são apresentadas as principais tecnologias utilizadas no desenvolvimento do Sistema Marvin.

2.1 Experiência do Usuário (UX) e Usabilidade

A experiência do usuário (UX) refere-se à qualidade da interação do usuário com o sistema, englobando aspectos como facilidade de uso, eficiência e satisfação (NORMAN, 2013). Interfaces bem projetadas aumentam a produtividade e reduzem erros, especialmente em sistemas complexos como os utilizados em ambientes acadêmicos.

Segundo Nielsen (NIELSEN, 1994), a usabilidade pode ser dividida em cinco atributos: facilidade de aprendizado, eficiência de uso, facilidade de memorização, taxa de erros e satisfação subjetiva. No contexto do Sistema Marvin, a personalização das interfaces visa melhorar esses atributos para diferentes perfis de usuários.

2.2 Personalização de Interfaces e Controle de Acesso por Papéis (Role-Based Access Control)

A personalização de interfaces consiste em adaptar a apresentação e as funcionalidades do sistema conforme o perfil do usuário. O controle de acesso baseado em papéis (RBAC) é uma abordagem comum para gerenciar permissões em sistemas multiusuário (SANDHU et al., 1996). Cada papel (por exemplo, administrador, gestor, aluno) possui um conjunto específico de permissões e visualizações, tornando o sistema mais seguro e intuitivo.

2.3 Engenharia de Software

A Engenharia de Software é uma área da computação dedicada à especificação, desenvolvimento, manutenção e gerenciamento de sistemas de software de forma sistemática, disciplinada e quantificável (PRESSMAN; MAXIM, 2016). Seu objetivo é garantir a qualidade, confiabilidade e eficiência dos sistemas desenvolvidos, utilizando processos,

métodos e ferramentas apropriadas. Entre as principais atividades da engenharia de software estão a definição de requisitos, o projeto de software, a implementação, os testes e a manutenção.

2.4 Levantamento de Requisitos

O levantamento de requisitos é uma das etapas mais críticas do desenvolvimento de software, pois envolve a identificação e documentação das necessidades e expectativas dos usuários e demais stakeholders (SOMMERVILLE, 2011). Os requisitos podem ser classificados em funcionais, que descrevem as funcionalidades do sistema, e não funcionais, que tratam de restrições e qualidades, como desempenho, segurança e usabilidade. Uma boa especificação de requisitos é fundamental para o sucesso do projeto, evitando retrabalho e garantindo que o produto final atenda aos objetivos propostos. No contexto do Sistema Marvin, a especificação de requisitos foi detalhada em documento próprio (CHANE, 2023), servindo de base para o desenvolvimento das funcionalidades e interfaces.

2.5 Projeto de Software

O projeto de software consiste na definição da arquitetura, dos componentes e das interfaces do sistema, a partir dos requisitos levantados. Essa etapa busca transformar as necessidades identificadas em uma solução técnica viável, considerando aspectos como modularidade, reutilização, escalabilidade e manutenibilidade (PRESSMAN; MAXIM, 2016). No contexto de sistemas web, o projeto também envolve decisões sobre a experiência do usuário (UX), a navegação, a disposição dos elementos na interface e a integração entre frontend e backend. O projeto do Sistema Marvin foi elaborado considerando as melhores práticas de engenharia de software, com ênfase na modularidade e escalabilidade. O documento de projeto descreve a arquitetura, os componentes principais e as integrações do sistema (SOUZA, 2024).

2.6 Engenharia Web

A Engenharia Web é um ramo específico da engenharia de software voltado para o desenvolvimento de aplicações baseadas na web. Ela abrange técnicas, metodologias e ferramentas para lidar com os desafios próprios desse ambiente, como a heterogeneidade de plataformas, a necessidade de interfaces responsivas, a segurança e a escalabilidade (GELLERSEN; GAEDKE, 2002). O desenvolvimento web moderno valoriza a experiência do usuário, a acessibilidade e a personalização das interfaces, aspectos essenciais para sistemas utilizados por diferentes perfis de usuários, como no caso do Sistema Marvin.

2.7 Tecnologias Utilizadas

O desenvolvimento do Sistema Marvin faz uso de tecnologias consolidadas e amplamente adotadas no contexto de aplicações web corporativas. Entre as principais, destacam-se:

- **Jakarta EE**: Plataforma para desenvolvimento de aplicações corporativas em Java, utilizada como base do backend do sistema.
- **EJB (Enterprise JavaBeans)**: Componente da plataforma Jakarta EE para implementação da lógica de negócio.
- **JPA (Jakarta Persistence API)**: Responsável pelo mapeamento objeto-relacional e persistência dos dados.
- **CDI (Contexts and Dependency Injection)**: Gerenciamento de dependências e ciclo de vida dos componentes.
- **JSF (Jakarta Server Faces) e Facelets**: Frameworks para construção das interfaces web do sistema, permitindo a criação de páginas dinâmicas e reutilizáveis.
- **Jakarta Security**: Gerenciamento de autenticação e autorização dos usuários.
- **PostgreSQL**: Sistema gerenciador de banco de dados relacional utilizado para armazenamento das informações.
- **WildFly**: Servidor de aplicações utilizado para execução e gerenciamento do sistema.
- **IntelliJ IDEA**: Ambiente de desenvolvimento integrado (IDE) utilizado pela equipe para implementação e manutenção do código-fonte.
- **Git**: Ferramenta de controle de versão utilizada para colaboração e gerenciamento do histórico de desenvolvimento.

Essas tecnologias foram escolhidas por sua robustez, comunidade ativa e aderência às melhores práticas de desenvolvimento web, além de já serem utilizadas no contexto do Sistema Marvin, facilitando a integração das novas interfaces ao sistema existente.

2.8 Trabalhos Relacionados

Diversos estudos destacam a importância da personalização de interfaces em sistemas acadêmicos e administrativos (ALBERT; JEONG; BARABÁSI, 1999; SOUZA; MYLOPOULOS, 2013). A literatura aponta que a adaptação das interfaces conforme o contexto de uso contribui para a eficiência e satisfação dos usuários.

2.9 Considerações Finais

A fundamentação teórica apresentada embasa as decisões de projeto e implementação das interfaces personalizadas no Sistema Marvin, destacando a importância da usabilidade, da personalização e do uso de tecnologias adequadas para o desenvolvimento web.

2.10 Seções e Subseções

O documento é organizado em capítulos (`\chapter{}`), seções (`\section{}`), subseções (`\subsection{}`), sub-subseções (`\subsubsection{}`) e assim por diante. Atenção, porém, a não criar estruturas muito profundas (sub-sub-sub-...) pois o documento não fica bem estruturado.

2.10.1 Referências a seções

Cada parte do documento (capítulo, seção, etc.) deve possuir um rótulo logo abaixo de sua definição. Por exemplo, este capítulo é definido com `\chapter{Introdução}` seguido por `\label{sec-fundteo}`. Assim, podemos fazer referências cruzadas usando o comando `\ref{rótulo}`: "O Capítulo 2 começa com a Seção 2.10, que é ainda subdividida nas subseções 2.10.1 e 2.10.2.

Para melhor organização das partes do documento, sugere-se primeiro utilizar o prefixo `sec-` (para diferenciar de referências à figuras, tabelas, etc. quando usarmos o comando `\ref{}`) e também representar a hierarquia das seções nos rótulos. Por exemplo, o Capítulo 2 tem rótulo `sec-fundteo`, sua Seção 2.10 tem rótulo `sec-fundteo-secoes` e a Subseção 2.10.1 tem rótulo `sec-fundteo-secoes-refs`.

2.10.2 Sobre referências cruzadas

Nas próximas seções, veremos que é possível fazer referência cruzada não só a seções mas também a listagens de código, figuras, tabelas, etc. Em todos estes casos, quando nos referimos à Seção X, Listagem Y ou Figura Z, consideramos que estes são os nomes próprios destes elementos e, portanto, usa-se a primeira letra maiúscula. Isso pode ser visto na Subseção 2.10.1, acima. A exceção é quando nos referimos a vários elementos ao mesmo tempo, por exemplo: "as subseções 2.10.1 e 2.10.2".

Por fim, ao usar o comando `\ref{}`, sugere-se separá-lo da palavra que vem antes dele com um ~ ao invés de espaço. Por exemplo: `o Capítulo~\ref{sec-fundteo}`. Isso faz com que o LaTeX não quebre linha entre a palavra `capítulo` e o número do capítulo.

2.11 Citações Bibliográficas

Este documento utiliza a ferramenta de gerenciamento de referências bibliográficas do LaTeX, chamada *BibTeX*. O arquivo `bibliografia.bib`, referenciado no arquivo LaTeX principal deste documento, contém algumas referências bibliográficas de exemplo. Assim como capítulos, seções, etc., tais referências também possuem rótulos, especificados como primeiro parâmetro de cada entrada (ex.: `@incollection{albert1999internet, ...}`.

Sugere-se um padrão para rótulos de referências bibliográficas para que fique claro também no código LaTeX qual referência está sendo citada. Por exemplo, ao citar a referência `dijkstra1959note`, sabemos que é um artigo escrito por *Souza* e outros, publicado no *SESAS* em *2013* (geralmente a pessoa que citou sabe que publicação é SESAS e quem é Souza).

Para citar uma referência bibliográfica contida no arquivo *BibTeX*, basta usar seu rótulo como parâmetro de um de dois comandos possíveis de citação:

- O comando `\cite{}` efetua uma citação tradicional, colocando o nome do(s) autor(es) e o ano entre parênteses. Por exemplo, `\cite{albert1999internet}` é transformado em (ALBERT; JEONG; BARABÁSI, 1999);
- O comando `\citeonline{}` efetua uma citação integrada ao texto, colocando o nome do(s) autor(es) direto no texto e somente o ano entre parênteses. Por exemplo, "de acordo com `\citeonline{albert1999internet}`" é transformado em: de acordo com Albert, Jeong e Barabási (1999);

Também é possível citar vários trabalhos de uma só vez, separando os rótulos das referências bibliográficas com uma vírgula dentro do comando apropriado. Por exemplo, `\cite{dijkstra1959note,lattes1947observations}` (DIJKSTRA, 1959; LATTES; OCCHIALINI; POWELL, 1947).

Os trabalhos citados são automaticamente incluídos na seção de referências bibliográficas, ao final do documento. Tudo é formatado automaticamente segundo padrões da ABNT.

2.12 Listagens de Código

O pacote `listings`, incluído neste template, permite a inclusão de listagens de código. Análogo ao já feito anteriormente, listagens possuem rótulos para que possam ser referenciadas e sugerimos uma regra de nomenclatura para tais rótulos: usar como prefixo o rótulo do capítulo, substituindo `sec-` por `lst-`.

A Listagem 2.1, por exemplo, possui o rótulo `lst-intro-exemplo` e representa o código que foi usado no próprio documento para exibir as listagens desta seção. Como podemos ver, a sugestão é que os arquivos de código sejam colocados dentro da pasta `codigos/` e tenham nome idêntico ao rótulo, colocando a extensão adequada ao tipo de código.

Listagem 2.1 – Exemplo de código LaTeX para inclusão de listagens de código.

```
1 \lstinputlisting[label=lst-intro-exemplo, caption=Exemplo de código \latex para
      inclusão de listagens de código., float=htpb]{codigos/lst-intro-exemplo.tex}
2
3 \lstinputlisting[label=lst-intro-outroexemplo, caption=Exemplo de código \java
      especificando linguagem utilizada., language=Java]{codigos/lst-intro-
      outroexemplo.java}
```

A Listagem 2.2 mostra um exemplo de listagem com especificação da linguagem utilizada no código. O pacote `listings` reconhece algumas linguagens[1] e faz "coloração" de código (na verdade, usa **negrito** e não cores) de acordo com a linguagem. O parâmetro `float=htpb` incluído em ambos os exemplos impede que a listagem seja quebrada em diferentes páginas.

Listagem 2.2 – Exemplo de código Java™ especificando linguagem utilizada.

```
1 public class HelloWorld {
2   public static void main(String[] args) {
3     System.out.println("Hello, World!");
4   }
5 }
```

2.13 Figuras

Figuras podem ser inseridas no documento usando o *ambiente* `figure` (ou seja, `\begin{figure}` e `\end{figure}`) e o comando `\includegraphics{}`. Existem alguns outros elementos e propriedades úteis de serem configuradas, resultando no código exibido na Listagem 2.3.

O comando `\centering` centraliza a figura na página. A opção `width` do comando `\includegraphics{}` determina o tamanho da figura e usa-se `\textwidth` (opcionalmente multiplicado por um número) para se referir à largura da página.

O parâmetro do comando `\includegraphics{}` indica onde a imagem pode ser encontrada. Foi criado o diretório `figuras/` para conter as figuras do documento, dando uma melhor organização aos arquivos. Ao abrir esta pasta, repare que as figuras possuem

[1] Veja a lista de linguagens suportadas em <http://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings#Supported_languages>.

Listagem 2.3 – Código LaTeX utilizado para inclusão das figuras na Seção 2.13.

```
\begin{figure}
  \centering
  \includegraphics[width=.25\textwidth]{figuras/fig-fundteo-exemplo.png}
  \caption{Exemplo de figura.}
  \label{fig-fundteo-exemplo}
\end{figure}

\begin{sidewaysfigure}
  \centering
  \includegraphics[width=\textwidth]{figuras/fig-fundteo-exemplosideways}
  \caption{Exemplo de figura em modo paisagem: um modelo de objetivos~\cite{souza
      -mylopoulos:spe13}.}
  \label{fig-fundteo-exemplosideways}
\end{sidewaysfigure}
```

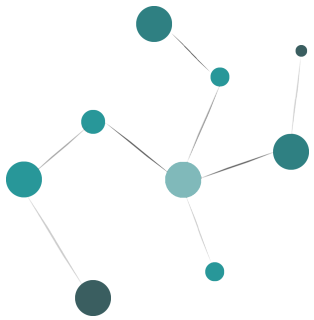

Figura 1 – Exemplo de figura.

duas versões—uma em `.eps` e outra em `.pdf`—e que o comando `\includegraphics{}` não especifica a extensão. Isso se dá porque o LaTeX possui um compilador para formato PostScript (`latex`) que espera as imagens em `.eps` e um compilador para PDF (`pdflatex`) que espera as imagens em `.pdf`.

Por fim, o comando `\caption{}` especifica a descrição da figura e `\label{}`, como de costume, estabelece um rótulo para permitir referência cruzada de figuras. Note ainda que é utilizada a mesma estratégia de nomenclatura de rótulos usada nas listagens, porém utilizando o prefixo `fig-`.

As figuras 1 e 2 mostram o resultado do código da Listagem 2.3. A Figura 2, em particular, utiliza o pacote `rotating` para mostrar figuras largas em modo paisagem. Basta usar o ambiente `sidewaysfigure` ao invés de `figure`.

2.14 Tabelas

Tabelas são um ponto fraco do LaTeX. Elas são complicadas de fazer e, dependendo da complexidade da tabela (muitas células mescladas, por exemplo), vale a pena construí-las em outro programa (por exemplo, em seu editor de texto favorito), converter para PDF e inclui-las no documento como figuras. Mostramos, no entanto, alguns exemplos de tabela a seguir. O código utilizado para criar as tabelas encontra-se nas listagens 2.4 e 2.5.

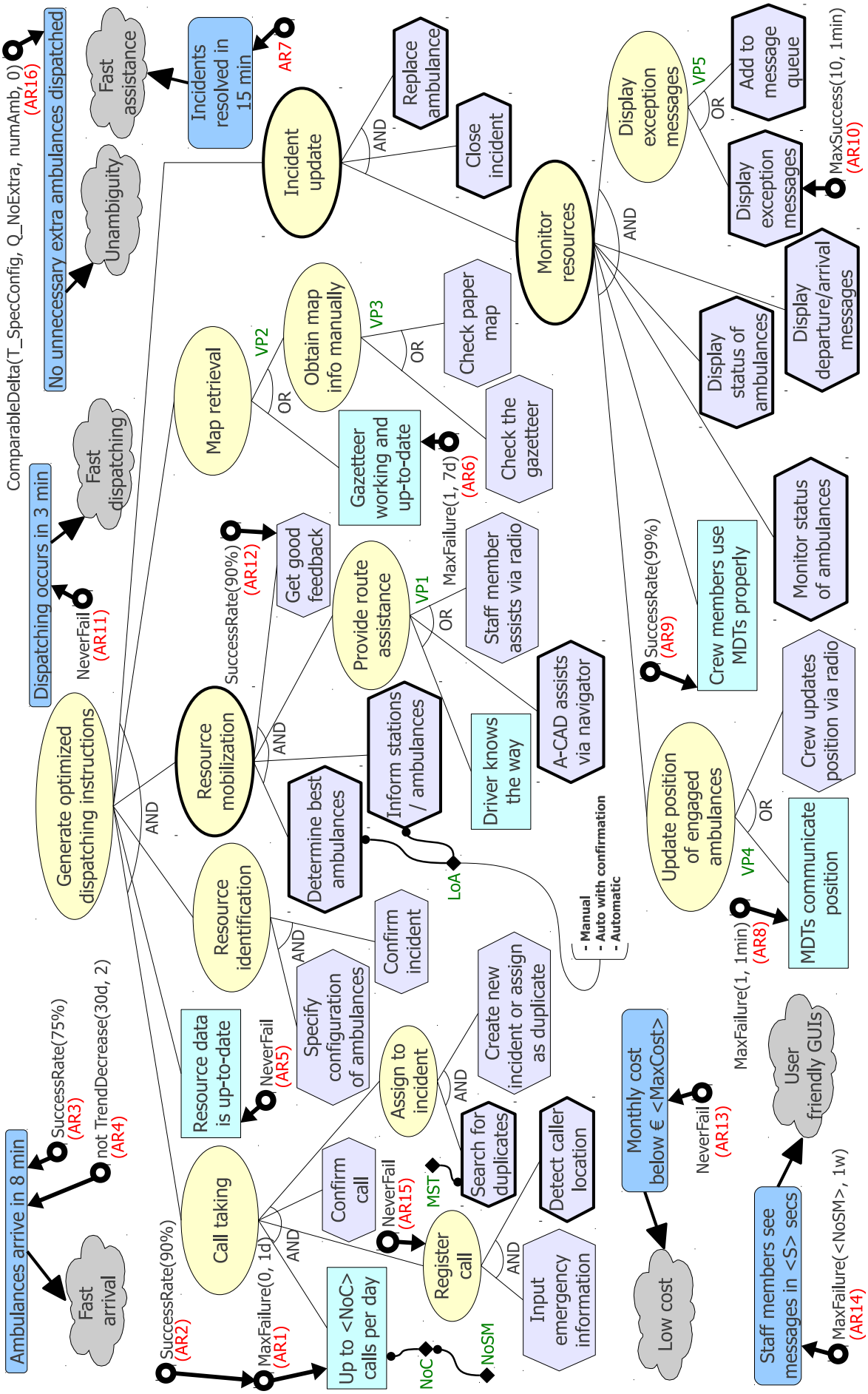

Figura 2 – Exemplo de figura em modo paisagem: um modelo de objetivos (SOUZA; MYLOPOULOS, 2013).

Listagem 2.4 – Código LaTeX utilizado para inclusão das tabelas 2 e 3.

```
% Exemplo de tabela 01:
\begin{table}
\caption{Exemplo de tabela com diferentes alinhamentos de conteudo.}
\label{tbl-intro-exemplo01}
\centering
\begin{tabular}{ | c | l | r | p{40mm} |}\hline
\textbf{Centralizado} & \textbf{Esquerda} & \textbf{Direita} & \textbf{Parágrafo }\\\hline
C & L & R & Alinhamento de tipo parágrafo especifica largura da coluna e quebra o texto automaticamente.\\
\hline
Linha 2 & Linha 2 & Linha 2 & Linha 2\\
\hline
\end{tabular}
\end{table}

% Exemplo de tabela 02:
\begin{table}
\caption{Exemplo que especifica largura de coluna e usa lista enumerada (adaptada de~\cite{souza-mylopoulos:spe13}).}
\label{tbl-intro-exemplo02}
\centering
\renewcommand{\arraystretch}{1.2}
\begin{small}
\begin{tabular}{ | p{15mm} | p{77mm} | p{55mm} |}\hline
\textbf{\textit{AwReq}} & \textbf{Adaptation strategies} & \textbf{Applicability conditions}\\\hline

AR1 &
\vspace{-2mm}\begin{enumerate}[topsep=0cm, partopsep=0cm, itemsep=0cm, parsep=0cm, leftmargin=0.5cm]
\item \textit{Warning(``AS Management'')}
\item \textit{Reconfigure($\varnothing$)}
\end{enumerate}\vspace{-4mm} &
\vspace{-2mm}\begin{enumerate}[topsep=0cm, partopsep=0cm, itemsep=0cm, parsep=0cm, leftmargin=0.5cm]
\item Once per adaptation session;
\item Always.
\end{enumerate}\vspace{-4mm}
\\\hline

AR2 &
\vspace{-2mm}\begin{enumerate}[topsep=0cm, partopsep=0cm, itemsep=0cm, parsep=0cm, leftmargin=0.5cm]
\item \textit{Warning(``AS Management'')}
\item \textit{Reconfigure($\varnothing$)}
\end{enumerate}\vspace{-4mm} &
\vspace{-2mm}\begin{enumerate}[topsep=0cm, partopsep=0cm, itemsep=0cm, parsep=0cm, leftmargin=0.5cm]
\item Once per adaptation session;
\item Always.
\end{enumerate}\vspace{-4mm}
\\\hline
\end{tabular}
\end{small}
\end{table}
```

Listagem 2.5 – Código LaTeX utilizado para inclusão da Tabela 4.

```
% Exemplo de tabela 03:
\begin{table}
\caption{Exemplo que mostra equações em duas colunas (adaptada de~\cite{souza-
    mylopoulos:spe13}).}
\label{tbl-intro-exemplo03}
\centering
\vspace{1mm}
\fbox{\begin{minipage}{.98\linewidth}
\begin{minipage}{0.51\linewidth}
\vspace{-4mm}
\begin{eqnarray}
\Delta \left( I_{AR1} / NoSM \right) \left[ 0, maxSM \right] > 0\\
\Delta \left( I_{AR2} / NoSM \right) \left[ 0, maxSM \right] > 0\\
\Delta \left( I_{AR3} / LoA \right) < 0\\
\end{eqnarray}
\vspace{-6mm}
\end{minipage}
\hspace{2mm}
\vline
\begin{minipage}{0.41\linewidth}
\vspace{-4mm}
\begin{eqnarray}
\Delta \left( I_{AR11} / VP2 \right) < 0\\
\Delta \left( I_{AR12} / VP2 \right) > 0\\
\Delta \left( I_{AR6} / VP3 \right) > 0\\
\end{eqnarray}
\vspace{-6mm}
\end{minipage}
\end{minipage}}
\end{table}
```

Tabela 2 – Exemplo de tabela com diferentes alinhamentos de conteúdo.

Centralizado	**Esquerda**	**Direita**	**Parágrafo**
C	L	R	Alinhamento de tipo parágrafo especifica largura da coluna e quebra o texto automaticamente.
Linha 2	Linha 2	Linha 2	Linha 2

Tabela 3 – Exemplo que especifica largura de coluna e usa lista enumerada (adaptada de (SOUZA; MYLOPOULOS, 2013)).

AwReq	**Adaptation strategies**	**Applicability conditions**
AR1	1. *Warning("AS Management")* 2. *Reconfigure(∅)*	1. Once per adaptation session; 2. Always.
AR2	1. *Warning("AS Management")* 2. *Reconfigure(∅)*	1. Once per adaptation session; 2. Always.

Tabela 4 – Exemplo que mostra equações em duas colunas (adaptada de (SOUZA; MYLOPOULOS, 2013)).

$\Delta\left(I_{AR1}/NoSM\right)[0, maxSM] > 0$	(2.1)	$\Delta\left(I_{AR11}/VP2\right) < 0$	(2.5)
$\Delta\left(I_{AR2}/NoSM\right)[0, maxSM] > 0$	(2.2)	$\Delta\left(I_{AR12}/VP2\right) > 0$	(2.6)
$\Delta\left(I_{AR3}/LoA\right) < 0$	(2.3)	$\Delta\left(I_{AR6}/VP3\right) > 0$	(2.7)
	(2.4)		(2.8)

Referências

ALBERT, R.; JEONG, H.; BARABÁSI, A.-L. Internet: Diameter of the world-wide web. *nature*, Nature Publishing Group, v. 401, n. 6749, p. 130, 1999. Citado 2 vezes nas páginas 7 e 9.

CHANE, V. A. *Especificação de Requisitos do Sistema Marvin*. 2023. Documento interno. Disponível na pasta de documentação do projeto. Citado 4 vezes nas páginas 2, 3, 4 e 6.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. *Numerische mathematik*, Springer, v. 1, n. 1, p. 269–271, 1959. Citado na página 9.

GELLERSEN, H.-W.; GAEDKE, M. Web engineering: Modelling and architectures. In: KAPPEL, G. et al. (Ed.). *Web Engineering*. [S.l.]: John Wiley & Sons, 2002. p. 3–24. Citado na página 6.

LATTES, C. M. G.; OCCHIALINI, G. P.; POWELL, C. F. Observations on the tracks of slow mesons in photographic emulsions. *Nature*, Nature Publishing Group, v. 160, n. 4067, p. 486, 1947. Citado na página 9.

NIELSEN, J. *Usability Engineering*. San Francisco: Morgan Kaufmann, 1994. Citado na página 5.

NORMAN, D. A. *The Design of Everyday Things*. Revised and expanded edition. New York: Basic Books, 2013. Citado na página 5.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 8. ed. Porto Alegre: AMGH Editora, 2016. Citado 2 vezes nas páginas 5 e 6.

SANDHU, R. S. et al. Role-based access control models. *IEEE Computer*, IEEE, v. 29, n. 2, p. 38–47, 1996. Citado na página 5.

SOMMERVILLE, I. *Engenharia de Software*. 9. ed. São Paulo: Pearson Prentice Hall, 2011. Citado na página 6.

SOUZA, V. E. S. *Projeto do Sistema Marvin*. 2024. Documento interno. Disponível na pasta de documentação do projeto. Citado na página 6.

SOUZA, V. E. S.; MYLOPOULOS, J. Designing an adaptive computer-aided ambulance dispatch system with Zanshin: an experience report. *Software: Practice and Experience (online first: http://dx.doi.org/10.1002/spe.2245)*, Wiley, 2013. Citado 4 vezes nas páginas 7, 12, 14 e 15.