

**UNIVERSIDADE FEDERAL DO PIAUÍ – UFPI
CAMPUS SENADOR HELVÍDEIO NUNES DE BARROS
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

**SISTEMA PARA GERENCIAMENTO DE IGREJAS
Sistema de Gerenciamento da Assembleia de Deus de Santo Antônio de Lisboa
– PI.**

GEDILSON PEREIRA DOS SANTOS

**PICOS – PIAUÍ
2016**

GEDILSON PEREIRA DOS SANTOS

SISTEMA PARA GERENCIAMENTO DE IGREJAS

Sistema de Gerenciamento da Assembleia de Deus de Santo Antônio de Lisboa

– PI.

Monografia submetida ao Curso de Bacharelado de Sistemas de Informação do Campus Senador Helvídio Nunes de Barros da Universidade Federal do Piauí como requisito parcial para obtenção de grau de Bacharel em Sistemas de Informação, sob a orientação do professor Leonardo Pereira de Sousa.

PICOS – PIAUÍ

2016

FICHA CATALOGRÁFICA

Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

S237s Santos, Gedilson Pereira dos.
Sistema para gerenciamento de igrejas sistema de
gerenciamento da Assembleia de Deus de Santo Antônio de
Lisboa. / Gedilson Pereira dos Santos. – 2016.
CD-ROM : 4 ¾ pol. (58f.)

Monografia (Bacharelado em Sistemas de Informação) –
Universidade Federal do Piauí,
Orientador (a): Prof. Esp. Leonardo Pereira de Sousa.

1. Sistemas Web. 2. Ruby on Rails. 3. Programa de
Computador - Igrejas. I. Título.

CDD 005.75

SISTEMA PARA GERENCIAMENTO DE IGREJA – SISTEMA DE
GERENCIAMENTO DA ASSEMBLÉIA DE DEUS DE SANTO ANTÔNIO DE
LISBOA - PI

GEDILSON PEREIRA DOS SANTOS

Monografia apresentada como exigência parcial para obtenção do grau de Bacharel em
Sistemas de Informação.

Data de Aprovação

Picos – PI. 14 de julho de 2016.



Prof. Esp. Leonardo Pereira de Sousa
Orientador



Prof. Me. Márcio Alves de Macêdo
Membro



Prof. Esp. Ismael de Holanda Leal
Membro

Dedico este trabalho a minha família, especialmente aos meus pais Gerson José dos Santos e Raimunda Pereira dos Santos, por terem sido paciente e acreditarem que esse sonho era possível.

AGRADECIMENTO

Primeiro agradeço a DEUS, pela oportunidade e pelo privilégio que me foi dado em compartilhar tamanha experiência e, ao frequentar este curso, perceber e atentar para a relevância de temas que não faziam parte da minha vida. A Ele, toda honra para sempre.

Ao meu Orientador Prof. Leonardo Pereira de Sousa pelo incentivo, simpatia e presteza no auxílio às atividades e discussões sobre o andamento e normatização desta Monografia de Conclusão de Curso.

Especialmente aos Professores Tiago José, Patrícia Medyna e Alcilene Dalília. Pelo seu espírito inovador e empreendedor na tarefa de multiplicar seus conhecimentos, pela sua forma peculiar de cada um na exposição do conhecimento e contribuição para o curso de Sistemas de Informação.

Aos demais idealizadores, coordenadores e funcionários da Universidade Federal do Piauí, Campus Senador Helvídio Nunes de Barros.

A todos os professores pelo carinho, dedicação e entusiasmo demonstrado ao longo do curso.

Particularmente ao meu amigo e Prof. Josimar José de Carvalho (Faculdade RSÁ), por sua vocação inequívoca, por não poupar esforços para nos motivar e contribuir com seus conhecimentos.

Aos colegas de classe, especialmente Itamar Egídio, Weder Moreira, João Mariano, Francilene Martins, Valdirene Araújo, Abimael Santiago, Laíse Nascimento, pela espontaneidade e alegria na troca de informações e materiais numa rara demonstração de amizade e solidariedade.

E, finalmente, aos nossos familiares pela paciência em tolerar a nossa ausência e entender o quanto me esforcei para chegar até aqui.

“Se viveres cada dia como se fosse o último, algum dia estarás muito provavelmente certo”.

Steve Jobs.

“O que torna um sonho irrealizável é a inércia de quem sonha”.

Desconhecido.

RESUMO

Em todos os setores da sociedade os sistemas de informações têm invadido e conquistado seu espaço. Isso acontece pela conscientização da eficácia dos sistemas na organização dos dados que são produzidos e mantidos por empresas, organizações, etc. O gerenciamento dessas informações deve existir em qualquer organização de forma que venham fornecer, com eficiência, maior controle na tomada de decisão. Este trabalho apresenta um sistema *web* para a igreja Assembleia de Deus em Santo Antônio de Lisboa – Piauí, que gerencia o setor financeiro de entrada e saída de recursos, registro patrimonial e principalmente controle de membresia¹, dispondo de relatórios específicos para análise e controle da instituição. Para o desenvolvimento desse *software* foram utilizados métodos e ferramentas como: a modelagem UML, a linguagem de programação *Ruby*, o *Framework Ruby on Rails* e o *Framework* de *front-end Bootstrap*.

Palavras-chave: Sistemas *Web*, *Ruby on Rails*, Sistemas para Igrejas.

¹ Membresia - Grupo de pessoas que formam uma organização com uma norma ou princípios comuns a todos os participantes.

ABSTRACT

In all sectors of society information systems have invaded and conquered their space. This is the awareness of the effectiveness of the systems in the organization of data that is produced and maintained by companies, organizations, etc. The management of this information must exist in any organization so that will provide, greater control in decision making and efficiency. This paper develops a web system for the Assembleia de Deus in Santo Antônio de Lisboa – Piauí, that manages the financial sector input and output capabilities, asset registration and mainly control membership², offering specific reports for analysis and control of the institution. For the development of this software were used tools and methods: UML modeling, the Ruby programming language, Ruby on Rails Framework and the front-end Framework Bootstrap.

Keywords: Web systems, Ruby on Rails, systems for churches.

² Membership - Group of people forming an organization with a standard or common principles to all participants.

LISTA DE ILUSTRAÇÕES

Figura 01 – <i>Acesso à internet</i>	19
Figura 02 – <i>Ciclo de vida Codifica-remenda</i>	28
Figura 03 – <i>Modelo em cascata</i>	29
Figura 04 – <i>Modelo Espiral</i>	30
Figura 05 – <i>O processo Scrum</i>	31
Figura 06 – <i>Diagrama de casos de uso do sistema</i>	38
Figura 07 – <i>Diagrama de classe do sistema</i>	40
Figura 08 – <i>Diagrama de Sequência – Cadastro</i>	41
Figura 09 – <i>Diagrama de Sequência – Alterar</i>	42
Figura 10 – <i>Diagrama de Sequência – Excluir</i>	43
Figura 11 – <i>Acesso de usuários do sistema</i>	44
Figura 12 – <i>Cadastro de funções: cargo (a), congregação (b), departamento (c), profissão (d)</i>	45
Figura 13 – <i>Cadastro de membro (a) e dados eclesiásticos (b)</i>	46
Figura 14 – <i>Cadastro de seminário (a) e participante (b)</i>	46
Figura 15 – <i>Cadastro de Patrimônio</i>	47
Figura 16 – <i>Cadastro de Receitas (a) e Despesas (b)</i>	48
Figura 17 – <i>Relatório Financeiro Específico</i>	48
Figura 18 – <i>Relatório Financeiro Geral</i>	49
Figura 19 – <i>Relatório Financeiro Anual: opções (a), por congregação (b)</i>	50
Figura 20 – <i>Relatório Financeiro Anual Geral</i>	50
Figura 21 – <i>Gerar Carteira de Membro</i>	51
Figura 22 – <i>Enviar e-mail</i>	51
Figura 23 – <i>Cadastrar: Administrador (a), Usuário Comum (b)</i>	52
Figura 24 – <i>Receita (a), Despesa (b)</i>	52
Figura 25 – <i>Tratamento de erro</i>	53
Figura 26 – <i>Diagrama Entidade e Relacionamento</i>	59

LISTA DE QUADROS

Quadro 1 – Requisitos funcionais.....	35
Quadro 2 – Requisitos não funcionais.....	36
Quadro 3 – Regra de negócios.....	36
Quadro 4 – Atores do sistema	37
Quadro 5 – Ata de entrevista.....	58

LISTA DE ABREVIATURAS E SIGLAS

CEADEPI	Convenção Estadual das Assembleias de Deus do Piauí
CoC	<i>Convention over Configuration</i>
CPE	<i>Customer Premises Equipment</i>
CRM	<i>Customer Relationship Management</i>
CSS	<i>Cascading Style Sheets</i>
DRY	<i>Don't Repeat Yourself</i>
HTML	<i>HiperText Markup Language</i>
IP	<i>Internet Protocol</i>
ISP	<i>Internet Service Providers</i>
MVC	<i>Model-View-Controller</i>
POP	<i>Point of Presence</i>
SCM	<i>Supply Chain Management</i>
SAD	Sistema de Apoio a Decisão
SAE	Sistema de Apoio ao Executivo
SGBD	Sistema de Gerenciamento de Banco de Dados
SGC	Sistemas de Gestão do Conhecimento
SI	Sistemas Integrados
SIG	Sistema de Informação Gerencial
SisGADSAL	Sistema de Gerenciamento da Assembleia de Deus de Santo Antônio de Lisboa – Piauí
SPT	Sistema de Processamento de Transações
SQL	<i>Structured Query Language</i>
UML	<i>Unified Modeling Language</i>

SUMÁRIO

1	INTRODUÇÃO.....	14
1.1	Objetivos	15
1.2	Organização do Documento	15
2	REVISÃO BIBLIOGRÁFICA	16
2.1	Sistemas de Informação	16
2.1.1	Classificação e tipos de Sistemas de Informação	17
2.2	Internet e Sistema <i>Web</i>.....	19
2.3	Tecnologias.....	20
2.3.1	Linguagem <i>Ruby</i>	21
2.3.2	<i>Framework Ruby on Rails</i>	22
2.3.3	<i>Framework Bootstrap</i>	24
2.3.4	Sistema de Gerenciamento de Banco de Dados	25
2.4	Engenharia de <i>Software</i>.....	26
2.4.1	Processo de <i>Software</i>	26
2.4.1.1	Modelos de Processos	28
2.4.1.2	Desenvolvimento Ágil de Processo	30
2.4.2	UML	32
3	ESPECIFICAÇÃO DO SISTEMA.....	34
3.4	Requisitos do Sistema.....	34
3.5	Diagrama de Casos de Uso.....	36
3.6	Diagrama de Classe.....	39
3.7	Diagrama de Sequência	40
4	FUNCIONAMENTO DO SISTEMA.....	44
4.1	Usuário Administrador	44
4.2	Usuário Comum	51
5	CONSIDERAÇÕES FINAIS.....	54
6	REFERÊNCIAS BIBLIOGRÁFICAS.....	55
	APÊNDICES.....	57

1 INTRODUÇÃO

O crescimento do número de evangélicos no nosso país tem mudado a realidade das igrejas provocando uma maior necessidade de controle, demandando um número maior de serviços no gerenciamento de informações que auxiliem na tomada de decisão.

A Igreja Evangélica Assembleia de Deus no Piauí, surgiu por volta de 1914 e foi crescendo e chegando a todas as cidades do estado, até que em 1969 chegou a Santo Antônio de Lisboa – PI. E como as estatísticas comprovam um vasto crescimento dos evangélicos no Brasil, a igreja em Santo Antônio de Lisboa vem crescendo, sendo de suma importância a prestação de contas com os membros, para isso necessita de um melhor gerenciamento do patrimônio.

Observando as atividades repetitivas que se encontram no meio administrativo da Igreja, percebe-se que é possível a realização dessas tarefas de forma mais automatizada com o uso dos sistemas de informação. Esses sistemas capturam os dados soltos dentro de um ambiente de nível menor, processando um aglomerado de dados e gerando informações, rápidas, precisas e de nível elevado através de relatórios.

Quando se fala de sistemas automatizados, que tem a capacidade de reunir dados/informações em um mesmo local, deve-se pensar em quais informações serão necessárias para melhor desempenhar as tarefas que são executadas manualmente, e assim condicionar os elementos relacionados para o processamento e a geração de informações precisas para a tomada de decisões voltadas para o planejamento e desenvolvimento das atividades.

Segundo Laudon e Laudon (2001, p. 4) consideram que “sistema de informação pode ser definido tecnicamente como um conjunto de componentes inter-relacionados que coleta (ou recupera), processa, armazena e distribui informação para dar suporte à tomada de decisão e ao controle da organização”. Os autores ainda acrescentam que “além de coordenar e apoiar a tomada de decisão, os sistemas de informação também podem ajudar os gerentes e funcionários a analisar problemas, visualizar assuntos complexos e criar novos produtos”.

O projeto proposto requer um sistema de informação que implemente pelo menos três módulos para organizar e gerenciar o atendimento aos seus membros: controle de membros, controle de patrimônio e controle financeiro.

O sistema conterà cadastro dos membros (aqueles que têm voz ativa), congregados (não podem opinar), congregações, funções desenvolvidas, listagem de usuários, membros, congregados, também com várias opções de relatórios. Além disso, o sistema terá um controle financeiro para administrar os recursos de entrada e saída (que lhe permitirá saber, por exemplo, quanto a igreja gastou com material de limpeza ou reparos realizados em um determinado mês), contas pagas e outras funcionalidades, com novas tarefas sendo adicionadas através de atualizações futuras.

1.1 Objetivos

Este trabalho tem por objetivo desenvolver um sistema de informação para automatizar as atividades eclesiais da Igreja Evangélica Assembleia de Deus, situada na cidade de Santo Antônio de Lisboa – PI.

1.2 Organização do Documento

Após o relato da introdução sobre a necessidade de se desenvolver este sistema e o objetivo do projeto, serão expostos os seguintes capítulos que estão descritos assim:

- Capítulo 2 – REVISÃO BIBLIOGRÁFICA: Fornece a base teórica para o trabalho. São mostrados conceitos de sistemas de informações, engenharia de software e tecnologias usadas no desenvolvimento.
- Capítulo 3 – ESPECIFICAÇÃO DO SISTEMA: É apresentado a análise de requisitos, os diagramas produzidos para entendimento e desenvolvimento do sistema.
- Capítulo 4 – FUNCIONAMENTO DO SISTEMA: São mostrados usuários do sistema e outras funcionalidades específicas que o mesmo apresenta.
- Capítulo 5 – CONSIDERAÇÕES FINAIS: apresenta a conclusão do trabalho e indicação de implementações futuras.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo mostra o embasamento teórico para o desenvolvimento deste trabalho, abordando pensamentos de autores relacionados a Sistemas de Informação, Internet e Sistema *Web*, Tecnologias e Engenharia de *Software*.

2.1 Sistemas de Informação

Os sistemas de informações e suas tecnologias se tornaram componentes vitais para alcançar objetivos nas grandes empresas e organizações, e por esse motivo, tornou-se um campo de estudo para administração e gerenciamento de todos os tipos de organização.

No geral, a definição de sistemas abrange todas as áreas da sociedade podendo ser aplicado a vários seguimentos como sistemas orgânicos, políticos, matemáticos e outros. Todos são compostos de diferentes elementos que interagem entre si geram algum tipo de resultado.

O conceito geral de sistemas é definido como um grupo de elementos inter-relacionados ou em interação que formam um todo unificado. A exemplo disso, podemos citar o sistema solar e os planetas ou até mesmo o sistema socioeconômico de uma empresa. No entanto, essa definição genérica de sistemas pode tornar mais compreensível quando se descreve sistemas de informação como “um conjunto organizado de pessoas, hardware, software, redes de comunicações e recursos de dados que coleta, transforma e dissemina informações em uma organização” (O'BRIEN, 2006, p. 6). Sendo assim, esses sistemas dinâmicos possuem componentes de interação: entrada, processamento e saída.

Segundo O'Brien (2006), estes componentes são descritos como: entrada, envolve a captação e reunião de elementos que entram no sistema; Processamento, processos de transformação que convertem insumos em produtos; E Saída, transferência de elementos produzidos na transformação até seu destino final.

Para complementar essa interação, ainda são necessários dois componentes adicionais, *feedback* e controle, que tem a função de automonitoramento, auto-regulado. O *feedback*, trata de dados sobre o desempenho de um sistema. A função do controle faz ajustes necessários aos componentes de entrada e processamento para garantir a qualidade de produção.

2.1.1 Classificação e tipos de Sistemas de Informação

Segundo Laudon e Laudon (2004, p.39), de acordo com as funcionalidades podemos classificar os sistemas de informação em quatro tipos de níveis principais:

- Sistema de Níveis Operacional – Dão suporte aos gerentes operacionais, no sentido de responder questões rotineiras através do acompanhamento de atividades e transações. Destaca-se o sistema de processamento de transações.
- Sistema de Nível de Conhecimento – Dão suporte aos trabalhadores de conhecimento e de dados auxiliando a empresa comercial a integrar novas tecnologias, assim como organizar e controlar o fluxo de documentos. Destacam-se os Sistemas de Trabalhadores do Conhecimento.
- Sistema de Nível Gerencial – Atendem as necessidades de gerentes médios e assessores, desenvolvendo atividades de monitoração controle e tomada de decisão. Destaca-se Sistema de Apoio a Decisão e o Sistemas de Informações Gerenciais.
- Sistema de Nível Estratégico – Estão relacionados a gerencia sênior, é capaz de analisar questões estratégicas e tendências das empresas e do ambiente externo. Destaca-se o Sistema de Apoio Executivo.

Geralmente as organizações usam os sistemas de informações aplicando aos seus devidos fins específicos na realização de suas atividades. No entanto, de uma forma hierárquica os sistemas se relacionam entre si recebendo dados dos níveis inferiores. Segundo Laudon e Laudon (2004, p.47), “é muito vantajoso que haja um grau de integração para esses sistemas para que a informação possa fluir facilmente em diferentes partes da organização.”. Os autores apresentam com detalhes os principais tipos de sistemas para empresas:

- SIG (Sistemas de Informações Gerenciais) – contém operações básicas da empresa, atendendo as necessidades dos gerentes de nível médio de monitorar e controlar a empresa, provendo também o seu desempenho futuro.
- SAD (Sistema de Apoio à Decisão) – auxiliam gerentes de nível médio a tomar decisões que fogem da rotina. Focam em um único problema que se

altera com rapidez e para o qual não existe uma resolução totalmente predefinida.

- SAE (Sistemas de apoio ao executivo) – Ajudam os executivos na gerência sênior a tomar decisões em relação a questões como tendências de custos no setor em longo prazo e como a empresa se encaixará nesse cenário. Abordam decisões não rotineiras que exigem bom senso e capacidade de avaliação e percepção.
- ERP (Sistemas Integrados) – coleta dados de vários processos fundamentais da organização, como: manufatura e produção, finanças e contabilidade, vendas e *marketing* e recursos humanos. Esses dados estão armazenados em um único repositório central. Assim, a informação que estava fragmentada em diferentes sistemas passa a ser compartilhada em toda a empresa permitindo que os diferentes setores da informação possam cooperar de forma mais próxima.
- SCM (*Supply Chain Management* ou Sistemas de gerenciamento de Cadeia de Suprimentos) – ajudam a empresa a administrar sua relação com os fornecedores. O objetivo fundamental desses sistemas são o de levar a quantidade certa de produtos da fonte ao local de consumo do produto, com menor uso possível de recursos de tempo e custo.
- CRM (*Customer Relationship Management* ou Sistema de gerenciamento do Relacionamento com o Cliente) – fornecem informações sobre o cliente para que seja possível coordenar todos os processos de negócios a ele relacionados, em termos de venda, *marketing* e serviços. Tem como objetivo maximizar a receita, a satisfação e retenção de clientes.
- SGC (Sistemas de Gestão do Conhecimento) – possibilitam que as organizações administrem melhor seus processos, capturando e aplicando os conhecimentos. Coletam todo o conhecimento e a experiência relevantes na empresa e os tornam disponíveis onde e quando forem necessários para melhorar os processos de negócios e as decisões administrativas.
- SPT (Sistemas de Processamento de Transações) – registra as transações de rotinas necessárias para o funcionamento da empresa como, por exemplo, registro de vendas, sistemas de reservas em hotéis, folha de

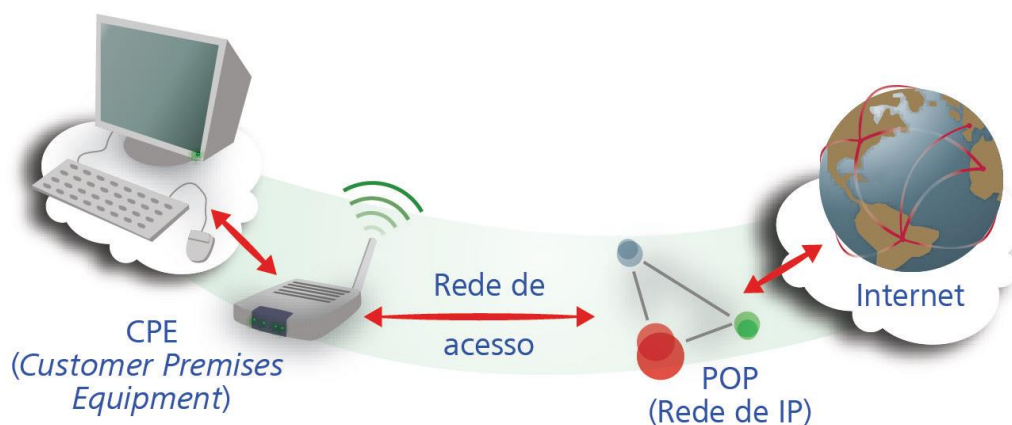
pagamento, registros de funcionários. O objetivo é responder perguntas de rotina e monitorar o fluxo de transações da organização.

2.2 Internet e Sistemas Web

Sem dúvida, a internet é a maior rede de computadores do mundo na qual integra centenas de milhares de outras redes locais, regionais e nacionais. As pessoas estão conectadas através de suas redes, em casa, no trabalho, faculdade e cada vez mais pessoas e empresas usam a internet para fins comerciais *online*, comunicação social, por meio dos provedores de serviços comerciais da internet (*Internet Service Providers – ISPs*). De acordo com Laudon e Laudon (1999, p.168), “um provedor de serviços da internet (ISP), é uma organização com uma conexão permanente com a internet que vende temporariamente conexões para assinantes”.

Segundo Boniati e Silva (2013), o acesso à internet acontece por meio de um ISP (*Internet Service Provider* ou provedor de serviço de internet) e utiliza-se de pelo menos três componentes CPE, Rede de acesso e POP ilustrados na figura 1.

Figura 1 - Acesso à internet.



Adaptado de: Boniati e Silva (2013).

- CPE (*Customer Premises Equipment*) é o equipamento que conecta o dispositivo à rede de acesso (exemplo: *modem*).
- Rede de acesso é o tipo de infraestrutura que liga o dispositivo ao provedor de internet (exemplos: cabos de cobre, fibra ótica, *Wi-fi*).

- POP (*Point of Presence*) é o ponto de presença do provedor onde estão os equipamentos que atribuem ao dispositivo um endereço IP, dando-lhe acesso à internet.

Para que se estabeleça conexão entre um computador e a internet e ainda se comunique com outros computadores, é necessário um número de identificação no qual chamamos de endereço IP (*Internet Protocol*). Fazendo uma analogia com uma linha telefônica, para se estabelecer uma conexão entre dois aparelhos é preciso o número do telefone de ambos (origem e destino). Da mesma forma acontece nas conexões da internet, quando um dispositivo qualquer (computador, *smartphone*, impressora), se conecta a um provedor de acesso, este recebe um número (número IP) que o permite comunicar-se com outros dispositivos ligados à internet.

Segundo Pauli (2013), o termo aplicação *Web* tem interpretações diferentes para pessoas diferentes. De acordo com o contexto, as pessoas farão uso de termos como aplicação *web*, *web site*, sistemas baseado em *web*, *software* baseado em *web*, podendo ter o mesmo significado. Sendo assim, sua definição para Sistema *web* baseia-se em qualquer aplicação que é processada em um servidor, no qual este recebe uma requisição e devolve uma resposta para o cliente e que exista uma interação com um banco de dados.

Rodrigues (2010) descreve a *web* como um sistema de informação em hipertexto, gráfico, distribuído, independente de plataforma, dinâmico, interativo e global, utilizado na internet. A internet como uma rede de computadores mundialmente conectados e a *web* como um conjunto de páginas acessíveis na internet através de *links*.

Os sistemas *webs* são desenvolvidos de acordo com as tecnologias existentes da *web* aproveitando-se da acessibilidade que dispõe através da internet de forma que todos podem ter acesso, se conectados à rede. Podendo ser utilizados na intranet, quando o sistema for usado internamente para uso restrito dos funcionários da organização.

2.3 Tecnologias

Este tópico mostra as tecnologias utilizadas para o desenvolvimento deste trabalho, abordando a linguagem de programação *Ruby*, *Framework Ruby on Rails*, *Framework Bootstrap* e Sistema de Gerenciamento de Banco de Dados.

2.3.1 Linguagem *Ruby*

Existem várias linguagens de programação para *web*, como por exemplo, *Ruby*, PHP (*Hypertext Preprocessor*), *Java*, *Python* e outras. Segundo Niederauer (2011), PHP é uma das linguagens mais utilizadas no mundo, com milhões de *sites* construídos. *Ruby*, por sua vez, é uma linguagem que se destaca por facilitar o desenvolvimento, possuidora de um gerenciador de pacotes e dependências que aumentam a produtividade.

Essa linguagem de programação *Ruby* foi criada pelo japonês *Yukihiro Matsumoto*, e apresentada ao público no ano de 1995. É uma linguagem multiparadigma (orientado a objeto, funcional, reflexiva e imperativa) e multiplataforma (é executável em diversos sistemas operacionais), puramente orientada a objetos, criada com o objetivo de ser lida e interpretada com simplicidade para facilitar o desenvolvimento e manutenção de sistemas para programadores familiarizados com as linguagens C e *Java* (CAELUM, 2014).

Por ser uma linguagem interpretada e não compilada, é necessário a instalação de um interpretador na máquina para executar os programas. Segundo Flanagan e Matsumoto (2008), o interpretador pode possuir a implementação de referência que define a linguagem MRI ("*Mat's Ruby implementation*") que é baseada em C ou alguma implementação alternativa como *JRuby* (baseada em *Java*), *IronRuby* (baseada em *.NET*), *Rubinius* (baseada em *Smalltalk*), *Cardinal* (baseado em *Perl*), entre outras.

Ruby possui um gerenciador de pacotes e dependências chamado *Ruby-Gems*. Nele contém as bibliotecas, nos arquivos *gems*, com extensão ".gem", também códigos fontes e outras informações.

Mais algumas características *RUBY - LANG* (2014):

- Capacidade de tratamento de exceções, tal como em *Java* ou *Python*, de forma a facilitar o tratamento de erros.

- Um verdadeiro *garbage collector mark-and-sweep* para todos os objetos *Ruby*. Não é necessário manter contadores de referência em bibliotecas de extensão (*extension libraries*).
- Escrever extensões C em *Ruby* é mais fácil do que em *Perl* ou *Python*, com uma API refinada para chamar *Ruby* a partir do código C.
- O *Ruby* pode carregar bibliotecas de extensão (*extension libraries*) dinamicamente se o Sistema Operacional permitir.
- O *Ruby* tem um sistema de *threading*³ independente do Sistema Operacional.
- O *Ruby* é altamente portátil: é desenvolvido principalmente em ambiente GNU/Linux, mas trabalha em muitos tipos de ambientes *UNIX*, Mac OS X, Windows 95/98/Me/NT/2000/XP, DOS, BeOS, OS/2, etc.

2.3.2 Framework Ruby on Rails

Existem vários *frameworks* que fazem uso da linguagem *Ruby*, como *Merb*, *Padrino*, *RailsBricks*, *Ruby on Rails*, *Sinatra* e outros. Porém, *Ruby on Rails*⁴ se destaca pela facilidade e praticidade que proporciona ao usá-lo para desenvolvimento de aplicações *web* com curto prazo de entrega.

Framework é uma coleção de bibliotecas e ferramentas que fornece de forma básica, mais completa, a infraestrutura para construção de aplicações de modo a facilitar o desenvolvimento. *Ruby on Rails* (ou apenas *Rails*) é um *framework* de código aberto para desenvolvimento de aplicações *web*, criado por David Heinemeier Hansson (ou "DHH") e lançado em 2004. Como pilares da agilidade *Rails*, dispõe de dois conceitos (CAELUM, 2014):

- *Don't Repeat Yourself* (DRY) - Em português "Não repita você mesmo", diz que não se deve repetir código e sim modularizá-lo. Ou seja, escreve-se trechos de código que serão reutilizáveis.
- *Convention over Configuration* (CoC) - Em português "Convenção acima de Configuração", diz que só devemos precisar fazer uma configuração nos casos excepcionais: ao utilizar os padrões oferecidos pelo *Rails*, configura-

³ Threading é um conjunto de tarefas existentes em um ou mais programas, executadas ao mesmo tempo pelo processador.

⁴ *Ruby on Rails* – <http://rubyonrails.org/>

se apenas o que estiver fora do padrão, diminuindo bastante a quantidade de configuração necessária para a grande maioria das aplicações.

O desenvolvimento das aplicações em *Rails* são implementadas usando a arquitetura MVC⁵ (*Model View Controller*). Cada parte do padrão MVC é uma entidade capaz de ser construída e testada separadamente. O modelo (*model*) representa os dados (é a informação que a aplicação trabalha); a visão (*view*) representa a interface do usuário (é a representação) e o controle (*controller*) comanda as ações, (é o diretor da interação entre eles) (SOUZA, 2014).

Ruby, Thomas e Hansson (2011) definem esses componentes:

- Modelo (*Model*): É responsável por manter o estado dos dados. Também impõe todas as regras de negócio que se aplicam aos dados;
- Apresentação (*View*): É responsável por gerar uma *interface* de usuário, normalmente com base em dados do modelo;
- Controle (*Controller*): É responsável por orquestrar a aplicação. Recebem eventos do mundo exterior, interagem com o modelo e mostram através da apresentação ao usuário.

Organizados dessa forma, as aplicações desenvolvidas em *Ruby on Rails* modulariza os processos agilizando a construção do projeto e futuras atualizações, se necessário.

Na camada de apresentação, *Ruby on Rails* faz uso das linguagens tais como: HTML (*HiperText Markup Language*), CSS (*Cascading Style Sheets*) e *Javascript*.

HTML em português traduz-se para linguagem para marcação de hipertexto. De acordo com Silva (2011, p. 20), “podemos resumir hipertexto como todo conteúdo inserido para *web* e que tem como principal característica a possibilidade de se interligar a outros documentos da *web*”. Sendo assim, os documentos marcados com HTML podem ser constituídos de títulos, parágrafos, listas, tabelas, figuras, entre outros.

A aparência da página *web* é muito importante para a usabilidade e legibilidade, para isso existe uma linguagem de folha de estilo, CSS (*Cascading Style Sheet*) responsáveis pela formatação dos documentos marcados com HTML, expondo como esses documentos serão apresentados.

⁵ Saber mais sobre MVC – <https://msdn.microsoft.com/en-us/library/ff649643.aspx>

Segundo Boniati e Silva (2013), CSS é um mecanismo com a função de adicionar estilos (fontes, cores, espaçamentos, bordas, sombras, etc.) aos elementos de documentos codificados em HTML. Todas as funções de apresentações, aspecto visual de documentos fazem parte da finalidade do CSS.

Além de proporcionar melhor aparência para páginas *webs*, o CSS foi introduzido para resolver problemas e economizar tempo. O princípio da folha de estilo consiste em agrupar características de formatação associadas a grupos de elementos em um mesmo documento (RODRIGUES, 2010).

O *JavaScript* é uma linguagem utilizada na camada de comportamento e é uma linguagem de *stripting*, o que significa que tem de ser utilizada em conjunto com outra linguagem, como por exemplo HTML.

De acordo com Santos (2009), *JavaScript* é uma linguagem de *script* que permite adicionar novos níveis de interatividades e funções as páginas *webs*, se responsabilizando pelo comportamento da página em HTML. Segundo a mesma, o *script* é uma sequência de instruções que segue padrões da linguagem e são interpretados por um *software*.

Esta camada permite que a aplicação tenha um comportamento mais dinâmico. Podemos exemplificar seu uso na mudança da imagem à medida que o mouse se movimenta sobre elas; abrir nova janela com controle programático sobre seu tamanho, posição e atributos; ou validar valores de formulários para garantir que são aceitáveis antes de enviar ao servidor. Isso é possível pelo uso das variedades de bibliotecas e suas funcionalidades específicas que facilitam o desenvolvimento e utilização da linguagem.

2.3.3 *Framework Bootstrap*

O *Framework Bootstrap* foi criado por Mark Otto e Jacob Thornton, desenvolvedores que até então trabalhavam no *Twitter*⁶, anunciaram ao mundo no dia 19 de agosto de 2011 o mais popular *framework JavaScript*, HTML e CSS para desenvolvimento de *sites* e aplicações *web* responsivas⁷. Indicado para

⁶ Twitter – <https://twitter.com/>

⁷ Web Responsiva é quando um site se adapta automaticamente a largura de tela do dispositivo no qual está sendo visualizado.

desenvolvedores de todos os níveis, além de tornar o desenvolvimento *front-end* muito mais rápido e fácil (SILVA, 2015).

O *Bootstrap* veio para facilitar a vida de desenvolvedores de aplicação *web*, pois o mesmo se utiliza da tecnologia responsiva iniciando o projeto de certo ponto onde se tem linhas de códigos prontas evitando toda a programação desde o início. Segundo Magno (2013), com o uso do *Bootstrap* não se faz necessário partir do zero o desenvolvimento de elementos básicos na *interface*, o mesmo dispõe de elementos comumente usados por *designers*, além de possuir um *kit* de ferramentas com as convenções padrão de classes com documentação limpa e prática para dar vida a códigos que estão prontos para serem usados e personalizados de acordo com cada desenvolvedor. Se adapta a diversos ambientes de desenvolvimento e possui ferramentas que agilizam o processo de implementação.

2.3.4 Sistema de Gerenciamento de Banco de Dados

De acordo com Elmasri e Navathe (2005, p. 3), “ Um banco de dados é uma coleção de dados relacionados. Os dados são fatos que podem ser gravados e que possuem um significado implícito”. Essa é uma definição genérica, mas podemos acrescentar que a partir de uma base de dados organizada, é possível gerar informações de forma lógica e estruturada.

Quem realmente organiza e armazena uma base de dados são os Sistemas de Gerenciamento de Banco de Dados (SGBD), que possuem uma coleção de programas que dão acesso aos dados, com objetivo de facilitar a forma de armazenamento e recuperação dos dados, gerindo uma massa de informações eficiente.

Segundo a Documentação do PostgreSQL 8.0.0 (2005), o *PostgreSQL* é um sistema de gerenciamento de banco de dados objeto-relacional (SGBDOR), baseado no *POSTGRES* versão 4.0, desenvolvido pelo Departamento de Ciência da Computação da Universidade da Califórnia em Berkeley. É um SGBD com código fonte aberto, de uso livre, que pode ser usado pela maioria das linguagens de programação.

O *PostgreSQL* oferece recursos como consultas complexas, chave estrangeira, gatilhos, visões, controle de simultaneidade multiversão, além de se estender com novos tipos de dados, funções e métodos de índice. Outro aspecto

notável, é que juntamente com a MySQL⁸, é um dos sistemas mais utilizados de banco de dados relacionais. Disponibiliza ainda, um recurso para manipulação dos dados usando linha de comandos para o principal cliente de terminal interativo, o *psql*, permitindo a execução de comandos SQL⁹ no servidor.

2.4 Engenharia de Software

O *software* está incorporado em todos os setores da sociedade, desde as áreas de transportes as industriais quanto em telecomunicações e máquinas de escritórios. Com o passar do tempo os *softwares* requerem aperfeiçoamento, adaptações, correções e isso requer tempo, esforço e recursos.

Pressman (2011, p. 30), faz relatos do crescimento desse mercado nos últimos anos:

Conforme aumenta a importância do *software*, a comunidade da área tenta desenvolver tecnologias que tornem mais fácil, mais rápido e mais barato desenvolver e manter programas de computadores de alta qualidade. Algumas dessas tecnologias são direcionadas a um campo de aplicação específico (por exemplo, projeto e implementação de *sites*); outras são focadas em um campo de tecnologias (por exemplo, sistemas orientados a objetos ou programação orientada a aspectos); e ainda outras são de bases mais amplas (sistemas operacionais como o Linux).

Ainda segundo Pressman (2011), a engenharia de *software* é uma tecnologia com base nas seguintes camadas: ferramentas, métodos, processos e foco na qualidade. Dentre essas, a camada de processo se destaca como a base da engenharia de *software*, por manter a coesão, possibilitando um desenvolvimento de forma racional e em prazos pré-estabelecidos.

2.4.1 Processo de Software

Sommerville (2011, p. 18), define processo de *software* como “um conjunto de atividades relacionadas que leva a produção de um produto de *software*”. Ainda segundo o autor, o desenvolvimento dessas atividades envolve o uso de uma

⁸ MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL como interface – <https://www.mysql.com/>

⁹ Linguagem de Consulta Estruturada ou SQL, é a linguagem de pesquisa declarativa padrão para banco de dados relacional.

linguagem de programação desde o início. Dentre vários processos existentes, o autor afirma que todos devem incluir quatro atividades fundamentais:

1. Especificação do *software*: definição das funcionalidades e as restrições fundamentais.
2. Projeto e implementação de *software*: deve ser produzido para atender as especificações.
3. Validação do *software*: deve ser validado para garantir que atenda as demandas do cliente.
4. Evolução do *software*: deve evoluir para atender as necessidades de mudanças do cliente.

Na prática essas atividades estão ligadas ao desenvolvimento do *software*, além de outras como: validação de requisitos, documentação, gerenciamento, etc.

Para Pressman (2011, p. 40), “processo é um conjunto de atividades, ações, e tarefas realizadas na criação de algum produto de trabalho”. Essas atividades, ações e tarefas trabalham paralelamente de forma adaptável a fim de produzir resultados tangíveis.

Considerado como a base de um processo de engenharia de *software*, a metodologia de processo estabelece meios para identificar atividades estruturais aplicáveis, independentemente do tamanho ou complexidade do projeto (PRESSMAN, 2011). Essa metodologia compreende cinco atividades, resumidamente, descritas abaixo:

1. Comunicação: antes de iniciar qualquer trabalho técnico, é de vital importância comunicar-se e colaborar com o cliente.
2. Planejamento: é a atividade de planejamento que cria um “mapa” que ajuda a jornada de trabalho. Esse “mapa” é o plano de projeto de *software*.
3. Modelagem: um engenheiro de *software* cria modelos para melhor entender as necessidades do *software* e o projeto que irá atender a essas necessidades.
4. Construção: essa atividade combina a geração de código e testes necessários para revelar erros na codificação.
5. Emprego: o *software* é entregue ao cliente, que avalia o produto e fornece *feedback*, baseado na avaliação.

São atividades metodológicas que podem ser usadas para o desenvolvimento de qualquer *software*, tanto pequenos como maiores e complexos.

De forma que cada sistema desenvolvido terá suas particularidades, mas sempre a mesma metodologia.

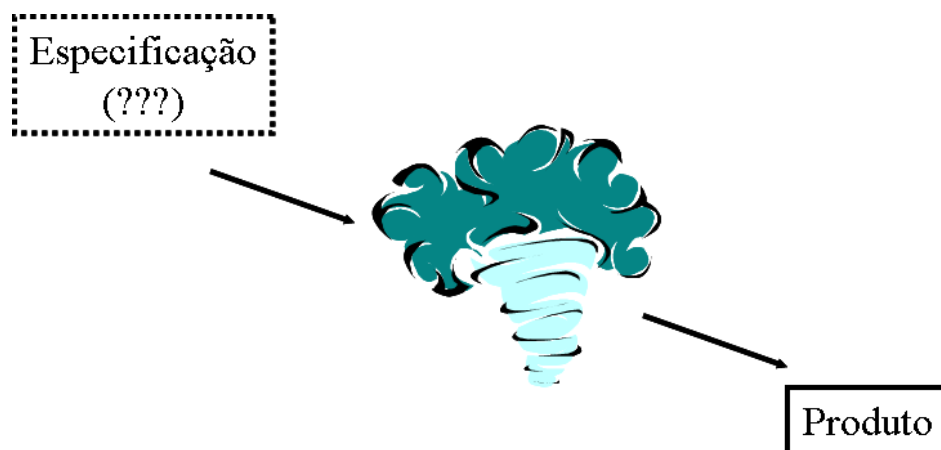
2.4.1.1 Modelos de Processo

Um processo no geral, pode ser definido como um conjunto de passos parcialmente ordenados, constituídos por atividades, métodos, maneira de agir ou conjunto de medidas tomadas para atingir algum objetivo.

De acordo com Pádua (2000), em engenharia de *software* os processos são definidos para atividades como desenvolvimento, manutenção, aquisição e contratação de *software*, podendo definir subprocessos para cada um destes. O ponto inicial para desenvolver um processo de *software* é a escolha de um modelo de ciclo de vida.

O ciclo de vida ou modelo de processo mais desordenado é o “Codificação-remenda” (Figura 2). Não se faz nenhuma especificação, já começa o desenvolvimento do sistema, codificando e corrigindo erros à medida que vão sendo descobertos. É um dos modelos mais utilizados por desenvolvedores, pois não exige nenhuma técnica de gerenciamento (PÁDUA, 2000).

Figura 02 – *Ciclo de vida Codifica-remenda.*



Adaptado de: (PÁDUA, 2000)

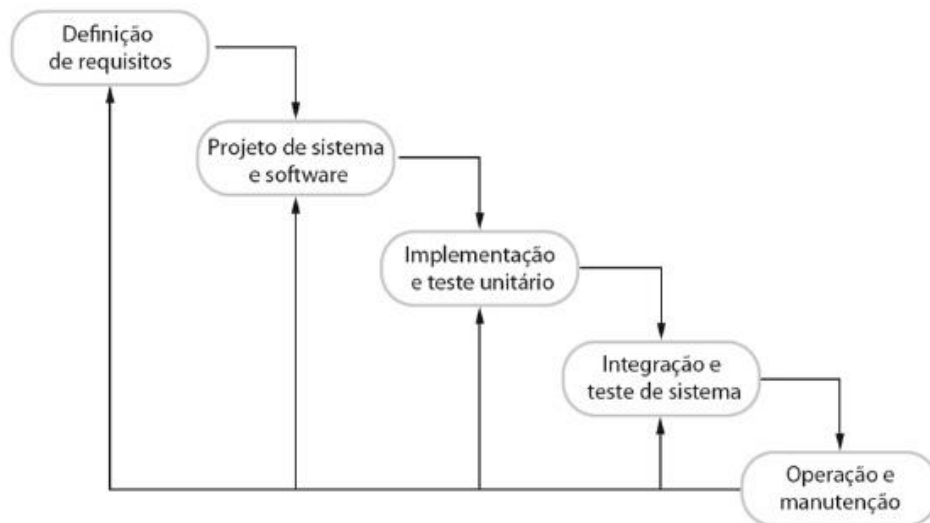
Sommerville (2011), aborda alguns modelos e dentre eles está o Modelo em Cascata (Figura 3). Esse modelo considera as atividades fundamentais do processo de especificação, desenvolvimento, validação e evolução e representa cada uma

delas como fases distintas, como: especificação de requisitos, projeto de *software*, implementação, teste e assim por diante.

Outro modelo de *software*, proposto por *Boehm* no ano de 1988, é em Espiral (Figura 4). Cada fase do processo de interações corresponde uma volta no espiral, assim o produto é desenvolvido. Pádua (2000) afirma que o produto é construído em prazos mais curtos, agregando novas características e recursos à medida que surge uma necessidade. Além disso, requer um gerenciamento mais preciso para manter a confiabilidade do processo.

Sommerville (2011, p. 32), ainda acrescenta que “o modelo em espiral combina prevenção e tolerância a mudanças, assume que mudanças são um resultado de risco de projeto e inclui atividades explícitas de gerenciamento de riscos para sua redução”.

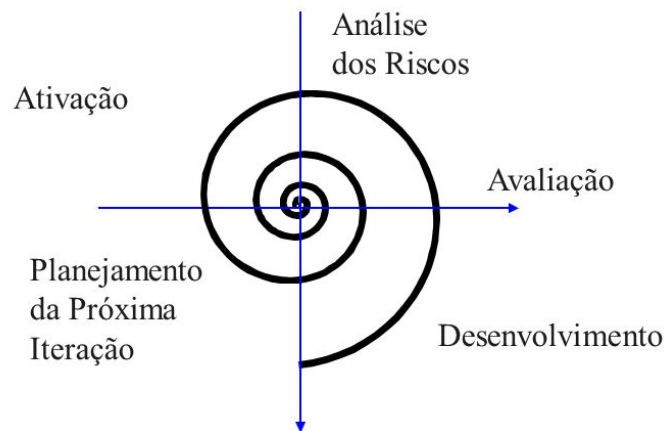
Figura 03 - Modelo em cascata.



Adaptado de: (SOMMERVILLE, 2011)

A engenharia de *software* tem a relevante importância no desenvolvimento de aplicações e sistemas computacionais em geral. Assim, fica evidente que todo *software* deve ser especificado, projetado, implementado e validado com critérios específicos para cada caso. Quando isso não acontece, existe uma enorme chance de o projeto falhar e o usuário rejeitar o *software* por não atender as necessidades pretendidas.

Figura 4 - Modelo espiral.



Adaptado de: (PÁDUA, 2000)

2.4.1.2 Desenvolvimento Ágil de Processo

O crescente mercado global exige das empresas mudanças rápidas na prestação de serviços e os *softwares* fazem parte de quase todas as operações de negócios. O desenvolvimento de *software* de forma ágil e entrega no prazo tornou-se um requisito ainda mais crítico para as empresas.

De acordo com Sommerville (2011, p. 39), "os processos de desenvolvimento rápido de *software* são concebidos para reduzir, rapidamente, softwares uteis". Sendo assim, seu desenvolvimento é feito com uma série de incrementos que inclui uma nova funcionalidade no sistema e não como parte única. Dentre muitas abordagens de desenvolvimento rápido de *software*, o autor cita algumas características fundamentais:

- Os processos de especificação, projeto e implementação são intercalados;
- O sistema é desenvolvido em uma série de versões. Os usuários finais e outros *stakeholders*¹⁰ do sistema são envolvidos na especificação e avaliação de cada versão.
- *Interfaces* de usuários do sistema são geralmente com um sistema interativo de desenvolvimento que permite a criação rápida do projeto de *interface* por meio de desenho e posicionamento de ícones.

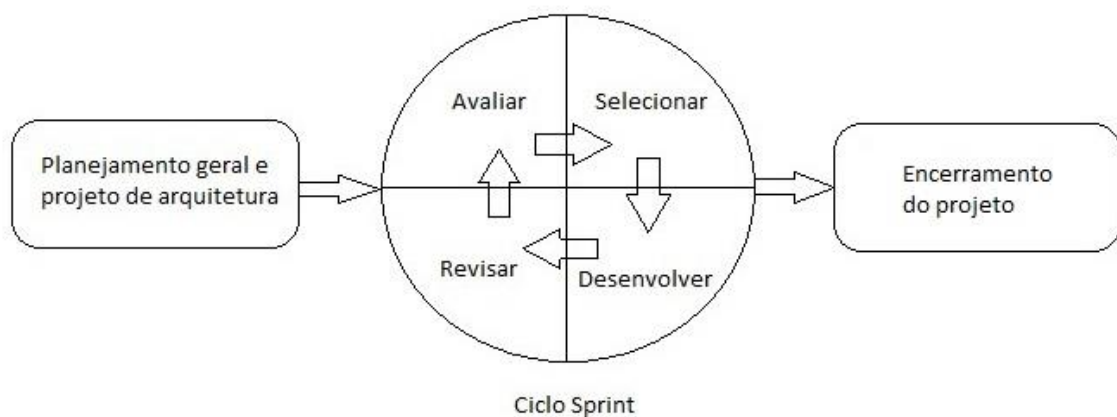
¹⁰ *Stakeholder* é uma pessoa ou grupo que possui participação, investimento ou ações e que possui interesse em uma determinada empresa ou negócio.

Os métodos ágeis envolvem os clientes com o objetivo de obter um *feedback* rápido para evolução dos requisitos. É realizado através de pequenos incrementos e entregue para ser avaliado no máximo em três semanas.

A exemplo do modelo ágil temos o *Scrum*, que segundo Schwaber, Schwaber e Beedle (2004 apud SOMMERVILLE 2011, p. 50), “é um método ágil geral, mas seu foco está no gerenciamento do desenvolvimento interativo, ao invés das abordagens técnicas específicas da engenharia de *software* ágil”. É uma metodologia que funciona como gestor estrategista em desenvolvimento de *software*.

Na Figura 5 mostra o processo *Scrum* de gerenciamento, que se divide em três fases: na primeira se estabelece os objetivos gerais do projeto e da arquitetura de *software*. Em seguida, ocorre uma série de ciclo *Sprint*, sendo que cada ciclo desenvolve um incremento do sistema. A última fase encerra o projeto, completa a documentação exigida, com quadros de ajuda do sistema, manuais do usuário e avalia as lições aprendidas com o projeto. (SOMMERVILLE ,2011).

Figura 05 – O processo Scrum.



Adaptado de: (SOMMERVILLE, 2011)

Ainda segundo Sommerville (2011), a fase central, “*Sprint*” do *Scrum* é ainda mais inovadora. No *Sprint* é onde é feita a avaliação da tarefa, seleção de recursos para desenvolvimento e implementação. Algumas características desse processo são:

- *Sprints* são de comprimento fixo, normalmente duas a quatro semanas.
- O ponto de partida para o planejamento é o *backlog* do produto, que é a lista de trabalho a ser feito no projeto.

- A fase de seleção envolve todos da equipe do projeto, que trabalham com o cliente para selecionar os recursos e a funcionalidade a ser desenvolvida durante o *Sprint*.
- Reuniões diárias rápidas, envolvendo todos os membros da equipe, são realizadas para analisar os progressos e, se necessário repriorizar o trabalho.
- No fim do *Sprint*, o trabalho é revisto e apresentado aos *stakeholders*.

2.4.3 UML

A Linguagem de Modelagem Unificada - UML (do inglês, *Unified Modeling Language*), é uma linguagem visual utilizada para modelar *softwares* baseados no paradigma de orientação a objetos. Utilizada pela engenharia de *software* em todo o planeta, ela auxilia a modelagem em todos os domínios de uma aplicação, de forma que é possível a visualização do desenho e a comunicação entre objetos (GUEDES, 2011).

Essa linguagem trabalha com um conjunto de diagramas que permite uma notação clara e consistente. Com os diagramas é possível a comunicação entre os desenvolvedores da aplicação e os usuários, contribuindo de forma preventiva com inconsistências no projeto. Estimula novas ideias na elaboração do projeto, além de possibilitar a documentação de artefatos do sistema (GÓES, 2014).

Nota-se a importância da modelagem do sistema antes da codificação, pois torna mais compreensível o que se pretende fazer. Quanto maior o sistema, mais complexo e propício à ocorrência de erros tornando indispensável o uso da UML. Segundo Góes (2014), a UML (versão 2.0) possui treze tipos de diagramas divididos em três categorias:

- Diagrama de estrutura – incluem o Diagrama de classe, Diagrama de objetos, Diagrama de componentes, Diagrama de estrutura composta, Diagrama de pacotes e Diagrama de implementação.
- Diagramas de comportamento – incluem o Diagrama de casos de uso, Diagrama de atividades e Diagramas de máquina de estado.

- Diagramas de interação – todos os derivados do diagrama de comportamento mais geral – incluem Diagrama de sequência, Diagrama de comunicação, Diagrama de tempo e Diagrama de visão geral de interação.

Segundo Sommerville (2011, p. 74), o diagrama de caso de uso é um dos modelos mais relevantes da linguagem de modelagem unificada (UML), “um caso de uso identifica os atores envolvidos em uma interação e dá nome ao tipo de interação. Essa é, então, suplementada por informações adicionais que descrevem a interação como sistema”. As informações suplementares podem ser uma descrição através de textos e ainda outros modelos da UML, como por exemplo o diagrama de sequência.

É um dos diagramas que simplifica o comportamento do sistema para o usuário, permitindo a compreensão das funcionalidades que ele dispõe, com o objetivo de uma apresentação informal e flexível para qualquer usuário. Guedes (2011, p. 52) afirma que “o diagrama de casos de uso tenta identificar os tipos de usuários que irão interagir com o sistema, quais papéis esses usuários irão assumir e quais funções esses usuários poderão requisitar”.

No próximo capítulo serão abordados, com mais detalhes, os respectivos diagramas de cada categoria utilizados no sistema SisGADSAL¹¹, no qual este trabalho está focado.

¹¹ Sistema de Gerenciamento da Assembleia de Deus de Santo Antônio de Lisboa.

3 ESPECIFICAÇÃO DO SISTEMA

Este capítulo apresenta as especificações do projeto que foi desenvolvido, um sistema para gerenciamento de igrejas. Serão abordados assuntos desde o levantamento de requisitos, metodologia aplicada, alguns diagramas compostos na UML como diagrama de casos de uso, diagrama de classes e diagrama de sequência e outros detalhes do sistema.

O SisGADSAL é um sistema de informação gerencial via *web*. A linguagem de programação *Ruby*, juntamente com o *framework Ruby on Rails*, foi escolhida para o desenvolvimento desse projeto pensando na facilidade de uso da sintaxe, com orientação a objetos, tornando mais produtiva a implementação e revisões futuras. O uso do *framework Ruby on Rails* facilita o desenvolvimento *web*, bem como a criação, implantação e manutenção. Segue conceitos de desenvolvimento Ágil, que permite maior agilidade, dispondo de recursos que facilitam a conexão com banco de dados, desempenho e qualidade do *software*. Com foco no projeto, a maior parte do tempo gasto são em *layout*, usabilidade, modelagem, testes, em coisas que agregam valor ao sistema e menos em configurações de bibliotecas (*GEMs*).

A metodologia para o desenvolvimento do sistema foi o *Scrum*, devido as suas características específicas ao projeto. Dentre elas está a de acompanhamento da equipe de desenvolvedores e usuários, revisando e validando cada ciclo.

Além dos estudos sobre a linguagem de programação *Ruby*, o *framework Ruby on Rails* e o *framework bootstrap*, foram realizadas outras atividades complementares para que pudesse obter o levantamento de requisitos, elaboração de entrevista e os diagramas de modelagem UML a partir dessas informações.

3.1 Requisitos do Sistema

Uma das fases iniciais para obtenção de informações do sistema de informação que se pretende desenvolver, consiste no levantamento de requisitos. Para isso, foi realizado reuniões e entrevistas (APÊNDICE A) com pessoas ligadas a administração da igreja para que pudesse compreender as necessidades e o processo de informatização. Com isso, foi possível determinar o que o *software* deve fazer, identificar os problemas e sugerir soluções cabíveis de implementação.

Através destas reuniões foi possível coletar um conjunto de requisitos importantes, que são: Funcionais (correspondem ao que o cliente quer que o sistema realize), Não-Funcionais (correspondem a restrições, consistências, validações que devem ser levadas a efeito sobre os requisitos funcionais) e Regra de Negócio (condições estabelecidas que devem ser seguidas na execução de uma funcionalidade).

O quadro a seguir (Quadro 1) mostra os requisitos funcionais com sua respectiva identificação, descrição e dependências.

Quadro 1 – Requisitos funcionais

Identificador	Descrição	Depende
RF01	O sistema deverá permitir cadastro de usuários.	
RF02	O sistema deverá permitir autenticação de usuário através de <i>e-mail</i> e senha.	RF01
RF03	O sistema deve controlar o cadastro de membros, congregados e congregações, e dados particulares.	RF02
RF04	O sistema deve registrar doações, indicando os itens doados e a data de doação.	RF03
RF05	O sistema deve registrar os pagamentos de dízimos e ofertas.	RF02, RF03
RF06	O sistema deve registrar as despesas de contas a pagar mensalmente.	RF05
RF07	O sistema emitirá relatórios contendo previsão de contas a pagar, realizar a movimentação do caixa e ter um controle efetivo das despesas.	RF05, RF06
RF08	O sistema emitirá certificados de seminários realizados.	RF02
RF09	O sistema deve permitir desativar e reativar membros.	RF03
RF10	O sistema deve emitir carteira de membros.	RF03

O Quadro 2 mostra os requisitos não funcionais com sua respectiva identificação, descrição, a categoria que pertence e dependências.

O Quadro 3 mostra as regras de negócio com sua respectiva identificação, descrição e dependências.

Quadro 2 – Requisitos não funcionais

Identificador	Descrição	Categoria	Depende
RNF01	O sistema deve controlar o acesso às funcionalidades, como: cadastro, relatórios e impressão (disponíveis através dos usuários do sistema).	Segurança de Acesso	RF02
RNF02	Identidade Visual e Interfaces amigáveis	Usabilidade	
RNF03	O sistema deve estar acessível na internet através dos principais navegadores do mercado.	Portabilidade	
RNF04	O tempo de resposta às requisições do sistema deve ser inferior a 3 segundos.	Performance	
RNF05	A persistência das informações deve ser implementada em um Sistema Gerenciador de Bancos de Dados Relacionais (SGBDR) livre (Postgres ou MySQL).	Manutenibilidade	

Quadro 3 – Regras de negócio

Identificador	Descrição	Depende
RN01	Não será permitido o cadastro de membros pelo usuário comum, a não ser pela congregação que lhe for permitido.	
RN02	Somente o administrador poderá excluir um cadastro.	RF03
RN03	Pagamentos de dízimos devem ser feitos em dinheiro ou donativos.	
RN04	Ofertas serão inseridas de acordo com a avaliação do valor da mesma.	

Concluído a análise dos requisitos funcionais, não funcionais e regra de negócio, passa-se por um processo de validação para tratar da consistência dos requisitos levantados. É um importante processo na engenharia de *software*, onde permite minimizar o tempo gasto, correção de incoerência, além ajudar na construção de importantes diagramas, como é o exemplo do diagrama de casos de uso.

3.2 Diagrama de Casos de Uso

O Quadro 4 mostra os atores identificados no sistema e uma descrição sobre a tarefa que cada um pode realizar.

Quadro 4 – Atores do sistema

Ator	Descrição
Administrador	Este tem total acesso as funcionalidades do sistema, gerenciamento de cadastros, exclusão de dados, impressão de relatórios em qualquer módulo do sistema.
Usuário	Tem acesso restrito as informações, podendo gerenciar apenas os dados da congregação a qual foi designado.

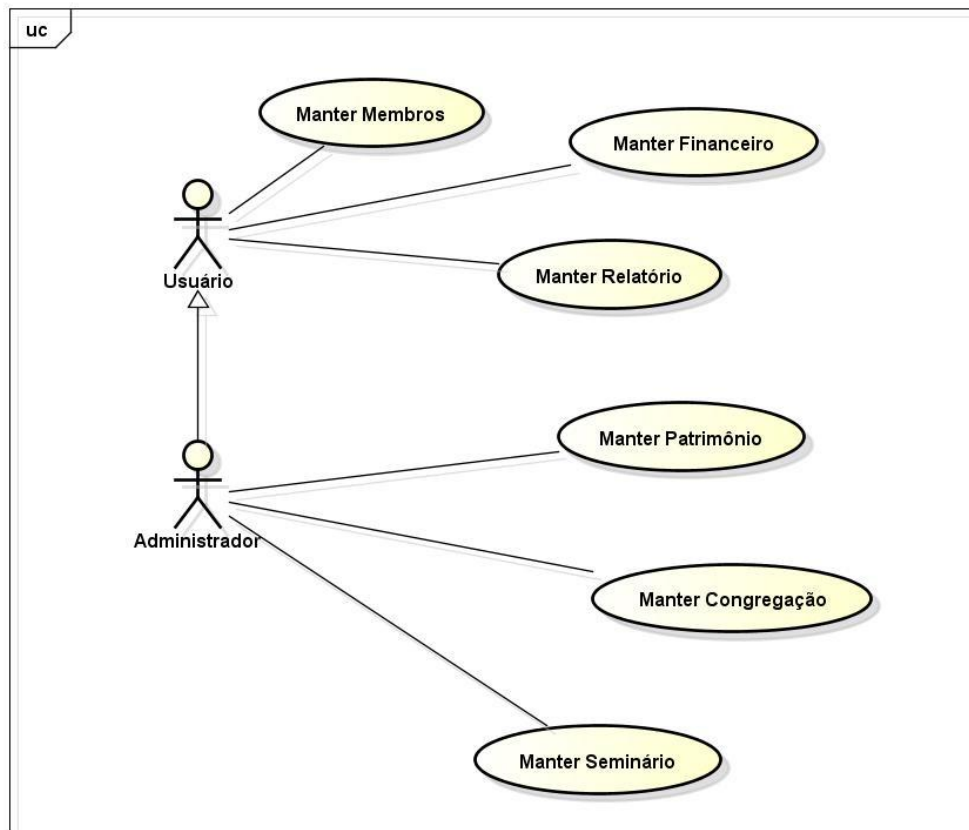
A representação visual do diagrama de casos de uso é feita através dos atores e casos de uso. Os atores são os “bonecos palito” com uma descrição abaixo e os casos de uso representados por uma elipse contendo uma descrição dentro. A ligações entre os “bonecos” e as elipses são chamadas de associações, que representam o relacionamento entre eles. Outros tipos de associação são: Generalização/Especialização, Inclusão (é representada por uma linha tracejada com uma seta em uma das extremidades, que aponta para o caso de uso incluído) e Extensão (semelhante ao de inclusão, sendo que este aponta para o caso de uso que se utiliza do caso de uso estendido).

Para melhor entendimento do diagrama de casos de uso (Figura 6), mostraremos detalhes das interações entre atores, casos de uso e os requisitos que estão relacionados.

- **Caso de Uso 01 – Manter membros:**
 - **Ator:** Usuário e administrador;
 - **Requisitos:** RF03;
 - **Descrição:** Realizar cadastro, exclusão, edição e listagem de membros do sistema (sendo que alguns dados estão restritos ao administrador). Toda saída do sistema parte dos registros das pessoas, tornando a principal informação.
- **Caso de Uso 02 – Manter congregação:**
 - **Ator:** Administrador;
 - **Requisitos:** RF03;

- **Descrição:** Realizar cadastro, exclusão, edição e listagem de congregação. Dá permissão para determinados usuários gerenciar algumas destas congregações.

Figura 6 – Diagrama de casos de uso do sistema



Fonte: O autor (2016).

- **Caso de Uso 03 – Manter patrimônio:**
 - **Ator:** Administrador;
 - **Requisitos:** RF03 e RF04;
 - **Descrição:** Realizar cadastro, exclusão, edição e listagem de patrimônio. Se refere as aquisições realizadas por cada congregação e atribuída ao sistema pelo administrador.
- **Caso de Uso 04 – Manter financeiro:**
 - **Ator:** Usuário e Administrador;
 - **Requisitos:** RF04 e RF05;
 - **Descrição:** Realizar cadastro, exclusão, edição e listagem de entrada e saída. Se refere as doações conforme especificado no requisito.
- **Caso de Uso 05 – Manter seminário:**

- **Ator:** Administrador;
- **Requisitos:** RF08;
- **Descrição:** Realizar cadastro, exclusão, edição e listagem de seminário. Torna possível a impressão de certificações dos participantes de palestras.
- **Caso de Uso 06 – Manter relatório:**
 - **Ator:** Usuário, Administrador;
 - **Requisitos:** RF07;
 - **Descrição:** Permite gerar relatório das informações gerais do sistema, em cada relatório é proposto uma abordagem específica dos dados.

3.3 Diagrama de Classe

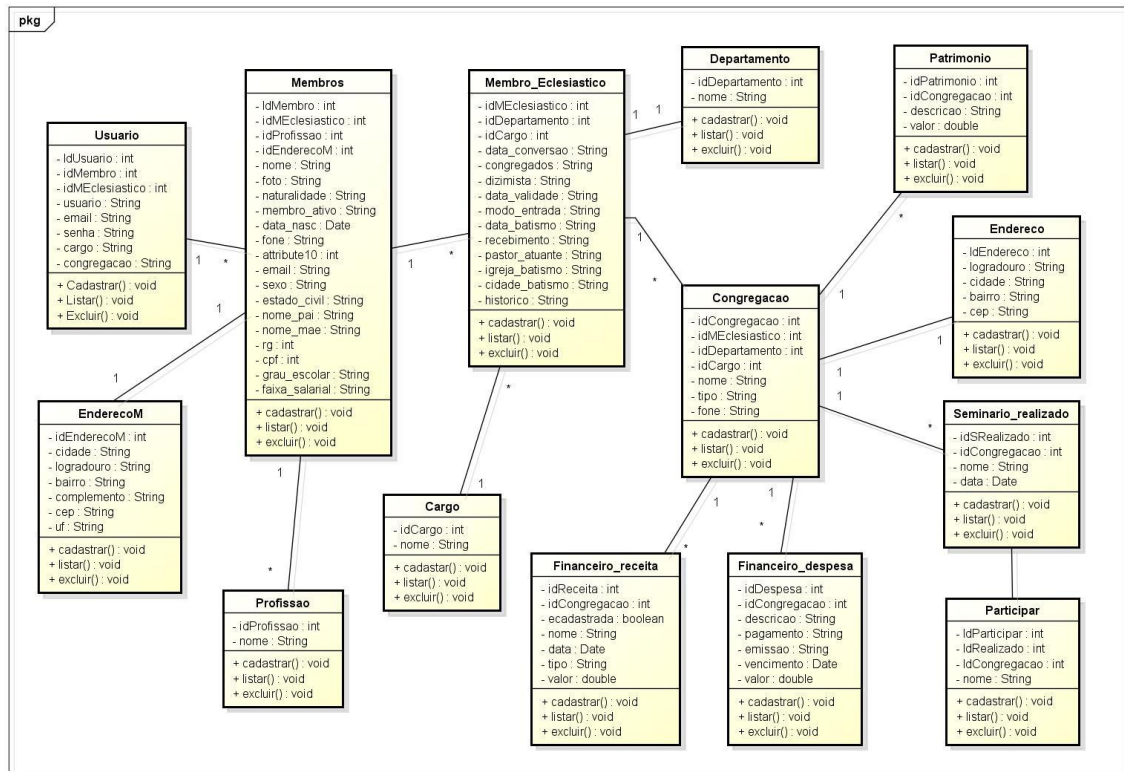
Esse diagrama mostra um conjunto de classes e seus relacionamentos, sendo um dos principais diagramas orientado a objetos da UML, com uma visão estática das classes, que serve de referência para o desenvolvimento de outros diagramas. As classes são representadas por um retângulo que inclui: nome, atributos e métodos. Os relacionamentos é a comunicação entre as classes do diagrama, podendo ser: associação (agregação ou composição), generalização e dependência. Com isso, são definidas as responsabilidades de cada classe do sistema.

O diagrama de classe (Figura 7) mostra como as classes estão distribuídas e relacionadas. De forma semelhante a esta, são as tabelas do banco de dados do sistema, mostrado em detalhes (APÊNDICE B).

A classe “membros” mantém dados cadastrais de pessoas vinculadas no sistema, assim como em “Membro_Eclesiastico” que são dados específicos ligados à igreja. As classes “Endereco” e “Profissao” ligadas a classe de “Membros” e as classes “Cargo” e “Departamento” ligadas a classe de “Membro_Eclesiastico” seguem como o padrão de normalização para melhor aplicação no banco de dados. Seguindo com a classe “Congregacao” que registra as igrejas como sede (matriz) ou congregações (filiais) ligados ao setor financeiro que são as classes de receitas “Financeiro_Receita” e despesas “Financeiro_Despesa”. Também ligada a classe de “Congregacao” está a classe que tem o objetivo de criar certificações para

seminários, chamada “Seminario_Realizado”. E finalmente a classe “Patrimonio” que é designada para registro das aquisições da instituição.

Figura 7 – Diagrama de classe do sistema



Fonte: O autor (2016).

3.4 Diagrama de Sequencia

O diagrama de sequência faz parte dos diagramas de interação da UML, focado em como as mensagens são enviadas no decorrer do tempo. Esse diagrama é baseado em casos de uso, podendo ser criado um diagrama de sequência para cada caso de uso, já que este refere-se a um processo disparado por um ator. Também depende do diagrama de classe, visto que as classes dos objetos estão descritas nele.

Segundo Guedes (2011, p. 192), “Este é um diagrama comportamental que procura determinar a sequência de eventos que ocorrem em um determinado processo, identificando quais mensagens devem ser disparadas entre os elementos envolvidos e em que ordem”. Sendo assim, o objetivo desse diagrama é estabelecer a ordem dos eventos, mensagens, métodos e objetos dentro de um processo.

O ator é responsável pelo início do processo, como está representado pelo “Administrador” na Figura 8, sua ação acarreta no disparo de um método em um dos

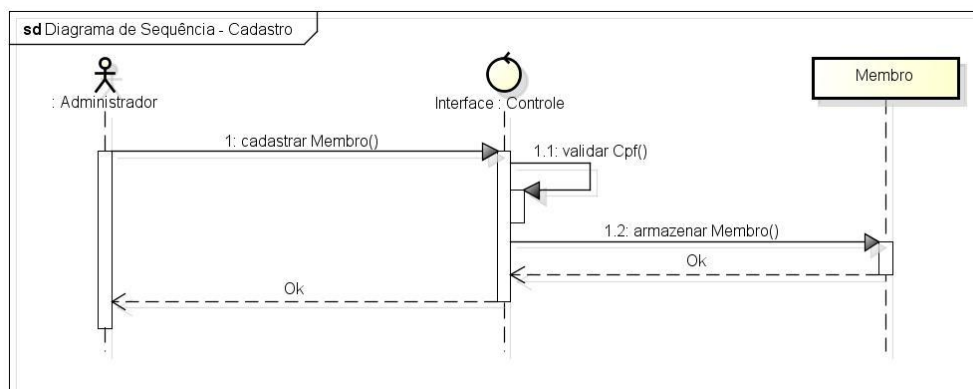
objetos (Interface) e conseqüentemente ativa outros métodos que acionam a classe (Membro) que é representado pelo triangulo. Entre esses eventos acontece a troca de mensagens indicado pela linha entre dois componentes, contendo setas indicando quem enviou a mensagem e quem recebeu.

Ao aplicar o diagrama de seqüência no caso de uso “manter membros” da Figura 6, podemos descrever os eventos de cadastro, alteração e exclusão. Esta modelagem pode ser usada em todos os outros casos de uso do sistema para detalhar os eventos sequenciais de cada caso.

Para realizar o cadastro de membros no sistema, deve-se seguir os seguintes passos:

- 1 – O administrador realiza o cadastro de um membro na interface;
- 2 – O sistema verifica se o CPF é valido;
- 3 – Os dados do membro são armazenados;
- 4 – O sistema retorna uma mensagem de confirmação de cadastro.

Figura 8 – Diagrama de Sequência - Cadastro



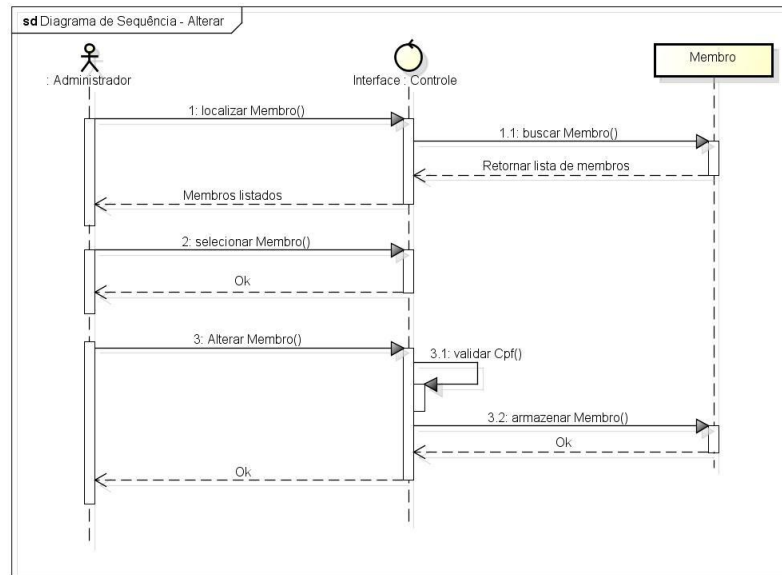
Fonte: O autor (2016).

A Figura 9 mostra o diagrama de seqüência, com o objetivo de apresentar a alteração do registro de um membro no sistema que segue os seguintes passos:

- 1 – O administrador localiza um membro na interface do sistema;
- 2 – O sistema busca a classe com os dados e apresenta à interface os dados solicitados;
- 3 – O membro seleciona o membro desejado para alteração, que retorna uma mensagem de confirmação;
- 5 – É feita a alteração do membro pelo usuário;

6 – O sistema valida os dados, armazena as alterações e retorna uma mensagem para interface.

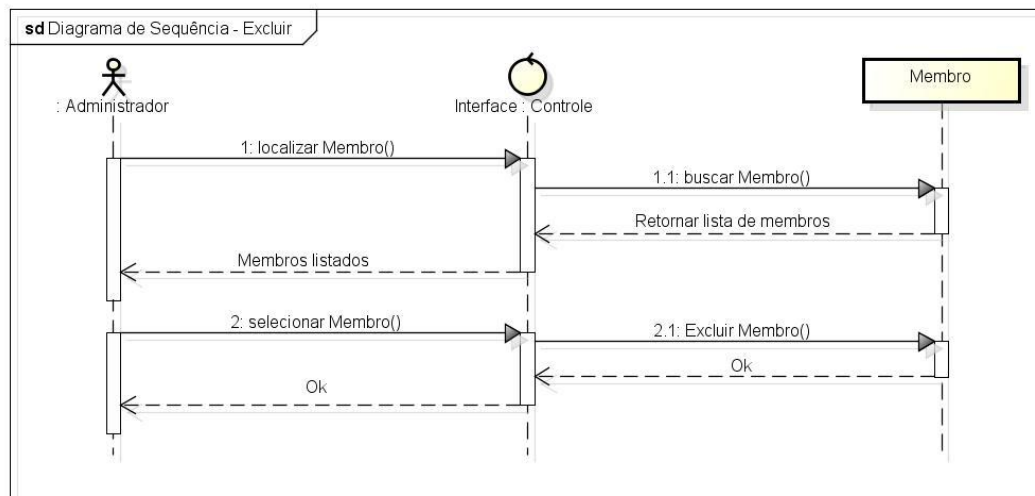
Figura 9 – Diagrama de Sequência - Alterar



Fonte: O autor (2016).

A Figura 10 mostra o diagrama de sequência, com o objetivo de apresentar a exclusão do registro de um membro no sistema que segue os seguintes passos:

- 1 – O administrador localiza um membro na interface do sistema;
- 2 – O sistema busca a classe com os dados e apresenta à interface os dados solicitados;
- 3 – O membro seleciona o membro desejado para exclusão;
- 5 – É feita a remoção do membro pelo usuário;
- 6 – O sistema retorna uma mensagem de confirmação para interface.

Figura 10 – Diagrama de Sequência - Excluir

Fonte: O autor (2016).

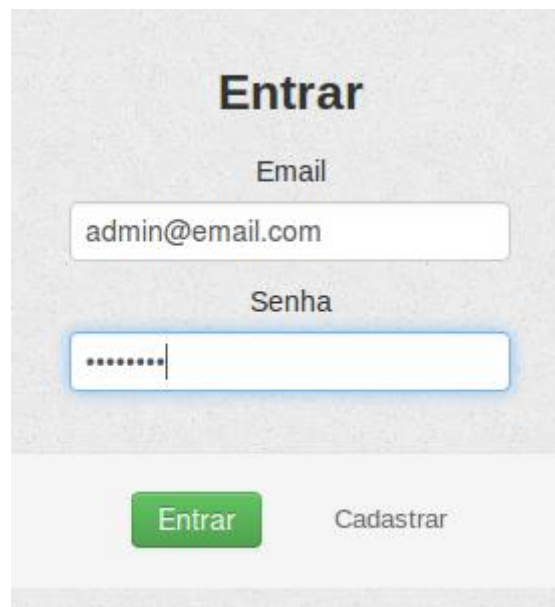
4 FUNCIONAMENTO DO SISTEMA

Esse capítulo explica as principais funcionalidades do SisGADSAL. Serão abordados os tipos de usuários que possuem acesso e as funções que os mesmos usarão no sistema. Depois serão apresentadas as telas do sistema de acordo com cada tipo de usuário, detalhando cada funcionalidade.

4.1 Usuários Administrador

Este é responsável por gerenciar todas as funcionalidades do sistema. Ao iniciar o sistema, o usuário “Administrador” poderá fazer uso de todas as funcionalidades. Uma vez feito *login*, será permitido o gerenciamento das funcionalidades do sistema, através do *e-mail* e senha cadastrados previamente (Figura 11), caso contrário o sistema não permitirá o acesso mostrando uma mensagem de erro do *e-mail* ou senha.

Figura 11 – Acesso de usuários do sistema



A imagem mostra a interface de login do sistema. No topo, o título "Entrar" está centralizado. Abaixo dele, há um campo rotulado "Email" com o endereço "admin@email.com" preenchido. Logo abaixo, há um campo rotulado "Senha" com pontos para ocultar o texto. Na base da interface, há dois botões: um verde com o texto "Entrar" e um cinza com o texto "Cadastrar".

Fonte: O autor (2016).

Inicialmente, o usuário “Administrador” deverá realizar os cadastros secundários que são: “Cargo”, “Congregação”, “Departamento” e “Profissão”. Esses cadastros irão complementar os campos de cadastro dos membros que serão solicitados nas tabelas de “Membro” e “Membro Eclesiástico” usando-os como

dependentes. É essencial que essas tabelas sejam preenchidas antes que se faça qualquer outro tipo de cadastro, principalmente dos membros (pessoa física). A Figura 12 mostra cada tabela de cadastro, cargo Figura 12(a), congregação Figura 12(b), departamento Figura 12(c), e profissão Figura 12(d).

Figura 12 – Cadastro de funções: cargo (a), congregação (b), departamento (c), profissão (d)

The figure displays four separate web forms for data entry, arranged in a 2x2 grid. Each form has a title bar with the form name and two buttons: 'Salvar' (Save) and 'Cancelar' (Cancel).
 (a) 'Novo Cargo': The 'Nome' field contains the text 'Membro'.
 (b) 'Nova Congregação': The 'Nome da congregação' field contains 'Assembleia de Deus', the 'Telefone' field contains '89 34491288', and the 'Setor ou bairro' field contains 'Sede'.
 (c) 'Novo Departamento': The 'Nome' field contains 'Jovens'.
 (d) 'Nova Profissão': The 'Nome' field contains 'Estudante'.

Fonte: O autor (2016).

Na Figura 13(a) mostra a tela de “cadastro de membro”, com parte dos dados que o sistema precisa para gerar alguns relatórios. Os dados da tela de “dados eclesiais” Figura 13(b), são informações que pertencem a igreja e que se faz necessário para o gerenciamento.

Além dessas funcionalidades, o sistema possui as funções de cadastro de “Seminário” Figura 14(a), que permite o cadastramento de seminários que a igreja deseja realizar, inserindo a descrição, data, carga horária e congregação que será ministrado, e também, a função “Participante em Seminário”, Figura 14(b), que permite cadastrar qualquer pessoa para fazer parte do seminário a ser ministrado pela igreja, podendo receber a certificação impressa no final do curso.

Uma forma de controlar os bens físicos é registrando cada aquisição. Foi com essa finalidade que o modulo de registro de patrimônio foi desenvolvido, para que cada congregação pudesse registrar seus bens patrimoniais, com apenas uma

descrição, valor e congregação que pertence (Figura 15). Cada congregação é responsável pelos bens adquiridos para uso interno, fazendo uso dessa funcionalidade para controle junto à administração.

Figura 13 – Cadastro de membro (a) e dados eclesiásticos (b)

Novo Membro Salvar Cancelar

Nome Joaquim Silva Velozo	Foto Browse... foto 3x4.jpg	Ativo Sim
Cpf 073.764.868-60	Data nascimento 16/04/1989	Naturalidade Valença do Piauí
Rg 2934933	Estado civil Solteiro	Nome pai Mario Pereira
Orgão expedidor SSP	Faixa salario Entre 1 e 3 salários mínimo.	Nome mae Joaquina Pereira
Email joaquim@gmail.com	Grau escolar Ensino Superior	Profissao Estudante
Sexo Masculino		

(a)

Novo Membro Eclesiastico Salvar Cancelar

Data conversão 02/01/2008	Pastor atuante Arlindo Melo	Cargo Membro
Congregação Sede	Recebimento	Historico
Data de batismo 04/05/2016	Igreja batismo Assembleia de Deus	
Dizimista Sim	Cidade de batismo Santo António de Lisboa	
Modo de entrada Decisão	Departamento Jovens	

(b)

Fonte: O autor (2016).

Figura 14 – Cadastro de seminário (a) e participante (b)

Novo Seminário Salvar Cancelar

Nome / descrição
A SEGUNDA VINDA DE JESUS

Data
01/06/2016

Carga horária
20

Congregação
Sede

(a)

Novo Participante Salvar Cancelar

Nome
Gerson José

Cpf
822.153.347-54

Seminário
A SEGUNDA VINDA DE JESU

(b)

Fonte: O autor (2016).

Figura 15 – Cadastro de Patrimônio

Novo Patrimônio Salvar Cancelar

Descrição
Aparelhagem de som

Valor
5000,00

Congregação
Sede

Fonte: O autor (2016).

O sistema dispõe de um módulo para controle de finanças, no qual registra a movimentação de receitas e despesas que é gerado a cada mês. Os registros mensais são contabilizados juntamente com algumas despesas fixas, as quais a igreja presta conta mensalmente à CEADEPI (Convenção Estadual das Assembleias de Deus do Piauí).

Nesse módulo estão as funcionalidades de receitas (dízimos e ofertas) e despesas. A Figura 16 mostra como é feita a inserção de valores no SisGADSAL. De acordo com o período corrente, a pessoa que irá contribuir pode estar cadastrada ou não, fazendo a doação voluntária de qualquer valor em uma determinada congregação Figura 16(a). De forma semelhante o cadastro de despesas, após confirmar o período, o usuário descreverá a conta a pagar, destinatário, valor, vencimento e a congregação Figura 16(b).

Em todas as funcionalidades cadastráveis no sistema permitem fazer uma listagem com opção para gerar um arquivo no formato “.PDF”, que facilita a impressão das informações desejadas.

Os relatórios são um conjunto de informações que foram inseridos ao longo de um período de uso do sistema, nos quais contém resultados gerados por determinada atividade que serve de base para tomada de decisão dentro de uma organização. Nesse caso, o sistema SisGADSAL aborda em seus relatórios informações relevantes para análise de desempenho, para servir de auxílio para a liderança e prestação de contas com os fiéis.

Figura 16 – Cadastro de Receitas (a) e Despesas (b)

Nova Receita Salvar Cancelar

Período ativo
01 /06 /2016 - 30 /06 /2016

ESTÁ CADASTRADA?
 Sim
 Não

Nome da pessoa
Rosa maria ▾

Tipo
Oferta ▾

Valor
100,00

Data
15/06/2016

Congregação
Acampamento ▾

Nova Despesa Salvar Cancelar

Período ativo
01 /06 /2016 - 30 /06 /2016

Descrição
Combustível

Pagar para
Posto Capital do Cajú

Valor
200,00

Vencimento
30/06/2016

Congregação
Sede ▾

(a)
(b)

Fonte: O autor (2016).

O usuário “Administrador” visualiza todos os relatórios específicos de uma congregação e a seleciona para listar as informações. O exemplo abaixo mostra que o usuário selecionou a congregação “Sede” para obter as informações daquele período, de 01 de junho a 30 de junho de 2016 (Figura 17). O relatório mostra os valores, em reais, de dízimos e ofertas no mês, despesas do período e o saldo líquido, que neste caso foi negativo. Permite também gerar e salvar um arquivo no formato “.PDF” para impressão.

Figura 17 – Relatório Financeiro Específico

Financeiro Gerar PDF Página principal	
Relatório financeiro	
Igreja Evangélica Assembleia de Deus - SEDE Rua Cirilo Lura, Nº 109 Centro CEP 64640-000	
Período referente: 01 /06 /2016 - 30 /06 /2016, da Assembleia de Deus, na congregação Sede Valores em R\$	
Total de Dízimos e Ofertas	200.00
CEADEP - Convenção Estadual das Assembleia de Deus	20.00
FP - Fundo de Penção	9.00
Outras Despesas p/ CEADEP	8.55
Despesas Cadastradas da Igreja	350.00
Somatório de Despesas	387.55
Valor Líquido	-187.55

Fonte: O autor (2016).

O próximo relatório mostra as informações financeiras de todas as congregações, o somatório naquele período, de 01 de junho a 30 de junho de 2016 (Figura 18). Permite também gerar e salvar um arquivo no formato “.PDF” para impressão.

Figura 18 – Relatório Financeiro Geral

Relatório financeiro	
Igreja Evangélica Assembleia de Deus - SEDE Rua Cirilo Lura, Nº 109 Centro CEP 64640-000	
Período a que se refere os dados: 01 /06 /2016 - 30 /06 /2016 Valores em R\$	
Total de Dízimos e Ofertas	300.00
CEADEP - Convenção Estadual das Assembleia de Deus	30.00
FP - Fundo de Penção	13.50
Outras Despesas p/ CEADEP	12.82
Despesas Cadastradas da Igreja	350.00
Somatório de Despesas	406.32
Valor Líquido	-106.32

Fonte: O autor (2016).

O relatório de finanças anual mostra as informações registradas de todos os períodos cadastrados no ano e realiza somatório de tudo. Quando selecionado no “menu” a opção de “Relatório Anual”, será aberto a página da Figura 19(a), que permite escolher o *link* desejado para relatório anual de uma congregação Figura 19(b) ou o relatório anual geral de todas as congregações (Figura 20). Para o usuário administrador não haverá exceção de congregações que ele possa visualizar, diferentemente de um usuário comum que terá acesso à sua congregação.

A geração da carteira de membro é uma funcionalidade bastante relevante, na qual, é obtida através das informações inseridas no cadastro do membro (Figura 21). Com essa “carteirinha” é realizado a identificação de uma pessoa como pertencente a uma determinada igreja, ou seja, é um documento interno que pode ser usado pelo fiel em outras igrejas (denominações) de uma outra cidade. Todos os membros que estão cadastrados podem possuir uma carteira de membro, podendo ser renovada ou não, dependendo da sua participação com as normas estabelecidas da igreja.

Figura 19 – Relatório Financeiro Anual: opções (a), por congregação (b)

Períodos [Página principal](#)

Ano	Opção
2016	Relatório Financeiro por Congregação Relatório Financeiro Geral

(a)

Congregações [Página Principal](#)

Selecione abaixo a CONGREGAÇÃO para gerar relatório financeiro

Nome	CEP	Bairro	Cidade	Logradouro	UF	Fone	Tipo	Selecionar
Assembleia de Deus	64640-000	Acampamento	Santo Antonio de Lisboa	justino lima	Piauí - PI	89 34491288	Acampamento	↓
Assembleia de Deus	64640000	BR 316 - km 300	Santo Antonio de Lisboa	rua cirilo lura	Piauí - PI	89 34491288	Bem-te-vi	↓
Assembleia de Deus	64640-000	Região urbana	Santo Antonio de Lisboa	sítio salvador	Piauí - PI	11111111	Sítio Salvador	↓
Assembleia de Deus	64640000	barro branco	Santo Antonio de Lisboa	rua do barro branco	Piauí - PI	89 34491288	Barro branco	↓
Assembleia de Deus	64640-000	Centro	Santo Antonio de Lisboa	rua cirilo lura	Piauí - PI	89 34491288	Sede	↓

(b)

Fonte: O autor (2016).

Figura 20 – Relatório Financeiro Anual Geral

Somatório de Despesas	231.43
Valor Líquido	468.57
PERÍODO 01 /12 /2016 - 31 /12 /2016	
Total de Dízimos e Ofertas	400.00
CEADEP - Convenção Estadual das Assembleia de Deus	40.00
FP - Fundo de Penção	18.00
Outras Despesas p/ CEADEP	17.10
Despesas Cadastradas da Igreja	200.00
Somatório de Despesas	275.10
Valor Líquido	124.90
Calculo de todos os periodos do ano de 2016!	
Total de Dízimos e Ofertas	7282.00
CEADEP - Convenção Estadual das Assembleia de Deus	728.20
FP - Fundo de Penção	327.69
Outras Despesas p/ CEADEP	311.31
Despesas Cadastradas da Igreja	1850.00
Somatório de Despesas	3217.20
Valor Líquido	4064.80

Fonte: O autor (2016).

O sistema dispõe de uma funcionalidade que facilita a comunicação com todos os membros da igreja, que é o envio de *e-mail* (Figura 22). Com isso, é possível alcançar todos os membros e mantê-los informados sobre reuniões em que todos devem participar. Para isso, é necessário que esteja devidamente cadastrado um *e-mail* de contato para cada membro.

Figura 21 – Gerar Carteira de Membro

Gerar próxima **Imprimir** Cancelar

Gere as carteirinhas de Membros sequencialmente!

Congregação: SEDE	Nº de Ordem: 00016		Igreja Evangélica Assembleia de Deus Rua Cirilo Lura, Nº 109 - Centro CEP 64640-000
Filiação: Misael da Silva Raimunda da Silva			CARTEIRA DE MEMBRO
Dt Nasc: 27 /06 /1976	Dt Batismo: 22 /06 /2016	Validade: 31/12/2016	Nome: Miguel da Silva
<small>Esta carteira é de uso pessoal e intransferível. Se você mudou de estado civil, residência ou telefone, atualize seus dados na secretaria da igreja. Identificação válida apenas quando acompanhada de carta de recomendação atualizada.</small>			RG: CPF: 212121 SSP 763.346.885-88
			Assinatura: <input type="text"/>

Fonte: O autor (2016).

Figura 22 – Enviar e-mail

Novo Email Cancelar

Mensagem

Aviso!
Amanhã haverá reunião
às 14hrs, não perca.

Enviar

Fonte: O autor (2016).

4.2 Usuário Comum

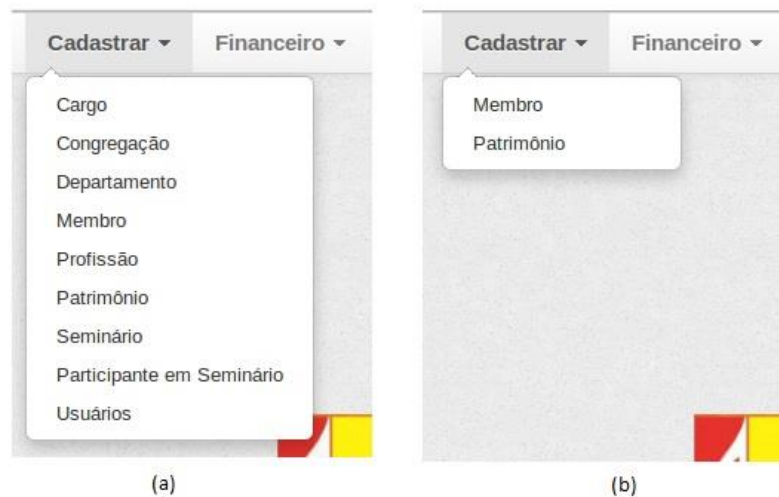
As funcionalidades apresentadas no item anterior são atribuídas ao usuário administrador do sistema. Nesta sessão, serão abordadas funções desempenhadas por um usuário comum ou específico destinado a uma congregação.

O usuário administrador terá autonomia para criar outros usuários, até mesmo administrador. Mas neste caso, se trata de um usuário comum que possuirá autonomia para executar o sistema nas funcionalidades cabíveis à congregação que ele é responsável.

O “Usuário Comum” tem algumas restrições no sistema para não interferir em dados de outra congregação. Os itens que ele pode cadastrar foram reduzidos,

permitindo o cadastramento de pessoas somente na congregação a qual foi definida pelo administrador (Figura 23). Sendo assim, ao cadastrar um membro, um patrimônio ou finanças, este usuário estará restrito à sua área de gerenciamento.

Figura 23 – Cadastrar: Administrador (a), Usuário Comum (b)



Fonte: O autor (2016).

Da mesma forma acontece ao acessar os relatórios, listagem de membros, listagem de patrimônio, listagem de receitas e despesas. Esse usuário só terá a sua disposição as informações que forem referentes à sua congregação. A figura a seguir mostra um exemplo que a “congregação” permanece a mesma em períodos diferentes para esse usuário (Figura 24).

Figura 24 – Receita (a), Despesa (b)

Pessoa	Tipo	Valor	Data	Congregação	Período
Abdias	Oferta	300.0	28 /01 /2016	Sítio Salvador	01 /01 /2016 - 31 /01 /2016
Valdo José	Oferta	200.0	18 /05 /2016	Sítio Salvador	01 /05 /2016 - 31 /05 /2016
Josue pereira	Dízimo	150.0	18 /02 /2016	Sítio Salvador	01 /02 /2016 - 29 /02 /2016
jose do sitio	Dízimo	200.0	18 /02 /2016	Sítio Salvador	01 /02 /2016 - 29 /02 /2016
Josue pereira	Dízimo	50.0	19 /08 /2016	Sítio Salvador	01 /08 /2016 - 31 /08 /2016
Gedilson	Oferta	300.0	26 /08 /2016	Sítio Salvador	01 /08 /2016 - 31 /08 /2016

(a)

Descrição	Pagar para	Valor	Vencimento	Congregação	Período
luz	eletrobras	50.0	31 /08 /2016	Sítio Salvador	01 /08 /2016 - 31 /08 /2016
Aluguel	Sr. Abraão	150.0	31 /08 /2016	Sítio Salvador	01 /08 /2016 - 31 /08 /2016

(b)

Fonte: O autor (2016).

O sistema possui um tratamento de erros para alguns campos obrigatórios, garantindo a inserção de valores corretamente. Se o usuário exceder o número de caracteres ou deixar um campo obrigatório vazio, certamente notificará um erro, como no exemplo a seguir da Figura 25. Essa funcionalidade é para todo usuário do sistema, comum e administrador.

Figura 25 – Tratamento de erro

A imagem mostra uma interface de usuário para o cadastro de um novo membro. No topo, há uma barra com o título "Novo Membro" e dois botões: "Salvar" e "Cancelar". Abaixo, o formulário é dividido em duas colunas. A coluna da esquerda contém os campos "Nome" (preenchido com "Joaquim Silva Velozo"), "Cpf" (preenchido com "0737648686000" e destacado por um retângulo vermelho), e "Rg" (preenchido com "2934933"). A coluna da direita contém os campos "Foto" (com um botão "Browse..." e o texto "No file selected."), "Data nascimento" (preenchido com "16/04/1989"), "Estado civil" (menu suspenso com "Solteiro" selecionado) e "Faixa salario" (menu suspenso com "Entre 1 e 3 salários mínimo." selecionado). No topo da área de formulário, uma mensagem de erro é exibida em um retângulo vermelho: "▪ Cpf é muito longo (máximo: 11 caracteres)".

Fonte: O autor (2016).

5 CONSIDERAÇÕES FINAIS

Atualmente, poucas igrejas evangélicas utilizam Sistemas de Informação para gerenciar suas tarefas rotineiras. A forma que as igrejas armazenam suas informações é em fichas de papel, anotações manuais, que se tornam vulneráveis a extravios e perda dos dados, além de aumentar a probabilidade de informações incorretas.

Tendo em vista estes problemas, surgiu a ideia de informatizar a igreja desenvolvendo o SisGADSAL, para que pudesse melhorar a maneira que as informações são mantidas. Esse sistema foi desenvolvido para a plataforma *web*, permitindo acessá-lo de qualquer lugar que tenha acesso à internet.

Foram descritas as etapas que nortearam a construção do sistema, identificando a real necessidade, pela qual foi constatada através de entrevistas com funcionários, pesquisa sobre sistemas de informação gerenciais, sistemas *web*, para aplicação destes conhecimentos. Em seguida, foram mostrados métodos abordados, tecnologias utilizadas e, finalmente, foram apontadas as principais funcionalidades do sistema.

Os resultados obtidos com a pesquisa e desenvolvimento do sistema de *software* são descrições das funcionalidades que foram desenvolvidas e testadas para atenderem os requisitos solicitados, as quais foram mostradas nas imagens como as principais telas do sistema.

Quanto a trabalhos futuros, algumas sugestões de implementação para o sistema de *e-mail*, fazendo com que envie mensagens de texto para um grupo específico. Outro quesito é a criação de um relatório individual para atender à solicitação de um membro, enviando para o *e-mail* do solicitante. Também, a criação de uma página *web* para divulgação de informações da igreja, tornando conhecidas suas atividades internas e sociais.

6 REFERÊNCIAS BIBLIOGRÁFICAS

BONIATI, B. B.; SILVA, T. L.; **Fundamentos de desenvolvimento web.** Westphalen, 2013.

CAELUM, Apostila do curso: **Desenvolvimento Ágil para Web com Ruby on Rails, 2014.** Disponível em: < <https://www.caelum.com.br/apostila-ruby-on-rails/>> Acesso em 24 de maio de 2016.

DOCUMENTAÇÃO DO POSTGRESQL 8.0.0. **The PostgreSQL Global Development Group.** California: Copyright © 1996-2005 The PostgreSQL Global Development Group.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de banco de dados.** 4a ed. São Paulo: Pearson Addison Wesley, 2005.

FLANAGAN, David; MATSUMOTO, Yukihiro. **The Ruby Programming Language.** 1 ed. Sebastopol: O'Reilly Media, 2008.

GÓES, Wilson M. **Aprenda UML por meio de um estudo de caso.** Novatec, São Paulo, Brasil, 2014.

GUEDES, Gilleanes T. A. **UML 2: Uma Abordagem Prática.** Novatec, São Paulo, Brasil, 2011.

LAUDON, K. C.; LAUDON, J. P. **Sistemas de informação: com internet.** 4. ed. São Paulo: Prentice hall, 1999.

LAUDON, K. C.; LAUDON, J. P. **Gerenciamento de sistemas de informação.** 3. ed. Rio de Janeiro, 2001.

LAUDON, K. C.; LAUDON, J. P. **Sistemas de informação gerenciais.** 5. ed. São Paulo: Prentice hall, 2004.

MAGNO, Alexandre. **Mobile First Bootstrap** Develop advanced websites optimized for mobile devices using the Mobile First feature of Bootstrap. 1 ed. Birmingham: Packt Publishing, 2013.

NIEDERAUER, Juliano. **Desenvolvimento Websites com PHP.** 2. Ed. São Paulo: Novatec Editora, 2011.

O'BRIEN, James A. **Sistemas de Informação e as decisões gerenciais na era da Internet.** 2. ed. São Paulo: Saraiva, 2006.

PAULI, Josh. **Introdução ao Web Hacking:** Ferramentas e técnicas para invasão de aplicações web. 1 ed. São Paulo: Novatec, 2013.

PRESSMAN, Roger S. **Engenharia de Software:** Uma Abordagem Profissional. Bookman, Porto Alegre, Brasil, 2011.

RODRIGUES, Andréa. **Desenvolvimento para Internet**, Curitiba: LT, 2010.

RUBY, Sam; THOMAS, Dave; HANSSON, David Heinemeier. **Agile Web Development with Rails**. 4 ed. Dallas: Pragmatic Bookshelf, 2011.

RUBY-LANG.ORG. About Ruby. **Ruby - A Programmer's Best Friend**, julho 2014.

Disponível em: <<http://www.ruby-lang.org/pt/about/>>. Acesso em: 24 de Maio 2016.

SOUZA, Nathália. **Introdução ao Framework Ruby on Rails**, setembro 2014.

Disponível em: <<http://www.devmedia.com.br/introducao-ao-framework-ruby-on-rails/31285>>. Acesso em 22 de Maio 2016.

SANTOS, Elisabete da Silva. **Apostila JavaScript**. FATEC, São Paulo, Brasil, 2009.

SILVA, Mauricio S. **Bootstrap 3.3.5**: Aprenda a usar o framework Bootstrap para criar layouts CSS complexos e responsivos. Novatec, São Paulo, Brasil, 2015.

SILVA, Mauricio S. **HTML 5**: A linguagem de marcação que revolucionou a web.

Novatec, São Paulo, Brasil, 2011.

APÊNDICES

APÊNDICE A – Ata de entrevista realizada com secretário da Igreja.

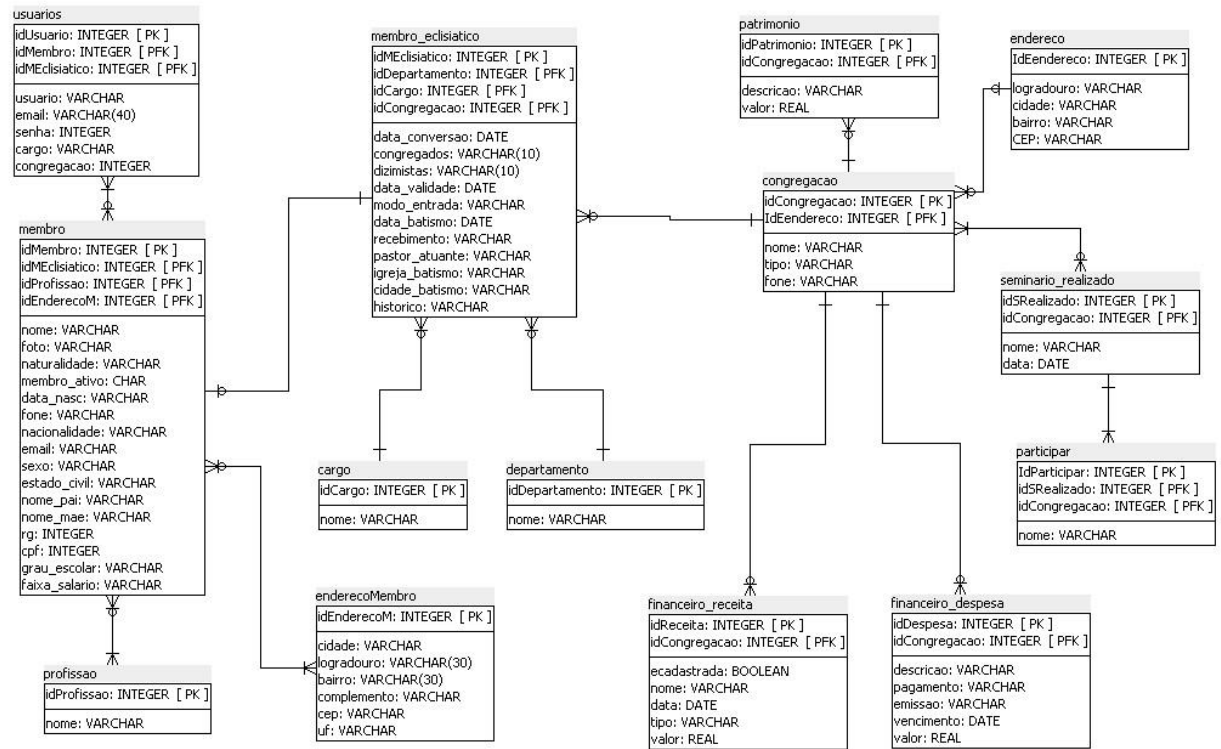
Quadro 5 – Ata de entrevista

Data: 21/04/2016	Entrevistado: Josimar Jose de Carvalho (faz parte da administração)	Entrevistador Gedilson Pereira dos Santos	Duração: 01h 30min: 00
Assunto: Implantação de Sistema de Informação em Igrejas Evangélicas; Processo de gerenciamento da Igreja; Rotinas de trabalho e atividades desenvolvidas na Igreja;			
Objetivos: <ul style="list-style-type: none"> · Conhecer as atividades que podem ser desenvolvidas no software. · Analisar como é o gerenciamento da igreja. · Observar as dificuldades de gerenciamento e as possíveis mudanças. · Como funciona o setor financeiro. · Como é feita as entradas e saídas de produtos da Instituição. 			
Perguntas Formuladas: <ul style="list-style-type: none"> · Quais as funções básicas que você julga necessário implementar no sistema? · Que módulos de gerenciamento das atividades deve ser desenvolvida? · Fale das atividades de gerenciamento? · Como é o gerenciamento das receitas e despesas? · Como é feito o controle financeiro e patrimonial? · Como é feito o controle dos membros (fieis)? · O que não pode faltar nos relatórios? 			
Pontos Discutidos: <p>A entrevista ocorreu de forma dinâmica, discutindo a respeito das atividades que podem ser realizadas no sistema dando mais agilidade no processo de controle administrativo da igreja.</p> <p>O controle de todos os membros e congregados foi um ponto forte na entrevista, pois dados importantes serão retirados deles. A emissão de relatórios foi uma das questões exigidas, visto que será gerada carteira de membros pelo sistema.</p> <p>O controle de patrimônio e financeiro é de suma importância, pois necessita-se de uma administração aberta para que todos os clientes (membros e congregados) vejam o que está sendo adquirido.</p>			

APÊNDICE B – Diagrama Entidade Relacionamento

Diagrama entidade relacionamento do sistema para gerenciamento de Igrejas.

Figura 26 – Diagrama Entidade e Relacionamento



Fonte: O autor (2016)



**TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA
“JOSÉ ALBANO DE MACEDO”**

Identificação do Tipo de Documento

- () Tese
() Dissertação
(X) Monografia
() Artigo

Eu, Gedilson Pereira dos Santos, autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação SISTEMA PARA GERENCIAMENTO DE IGREJAS Sistema de Gerenciamento da Assembleia de Deus de Santo Antônio de Lisboa – PI, de minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos – PI 10 de agosto de 2016.

Gedilson Pereira dos Santos
Assinatura