Aplicação Web para auxilio e estimulo de crianças a leitura (História Interativa)

Breno Nunes Arantes[1], **Dilson Luz Barros**[1], **Jerffeson Nascimento Gonçalves**[1], **Marcos Paulo Pereira**[1], **Maurício Mateus Oliveira de Assis**[1]

Orientador: Leonardo Santana Almeida

[1]Centro Universitário Jorge Amado – (UNIJORGE)
CEP: 40.015-060 - Rua Miguel Calmon, 42, Edf. São Paulo - Comércio - Salvador/BA

pi4sem@googlegroups.com

Abstract. *In this document we propose the contents of developing a web-based system in order to help stimulate the teaching of reading children. Through interactive books in which the reader can make a decision and/or be present in the own history. This article reports a disciplinary project, describing the need and possible benefits that will be brought to readers. However it has a technical approach regarding the development of online systems, their operation, the tools used in the design and concepts proposed by guiding this project. The application is fully interactive, with nice books for reading and dynamic. various theoretical concepts provided by the guiding thus resulting the prompt implementation were used. Thus this project can deal directly with the proposed problem is the lack of reading stimulation.*

Resumo. *Neste documento propomos o conteúdo do desenvolvimento de um sistema via web com o intuito de ajudar estimulando no ensino da leitura de crianças. Por meio de livros interativos no qual o leitor poderá tomar uma decisão e/ou estar presente na própria historia. Este artigo relata um projeto disciplinar, descrevendo a necessidade e possíveis benefícios que serão trazidos aos leitores. Contudo o mesmo tem uma abordagem técnica a respeito do desenvolvimento de sistemas online, seu funcionamento, ferramentas usadas no projeto e conceitos propostos pelos orientadores deste projeto. A aplicação está totalmente interativa, com livros agradáveis para a leitura e dinâmica. Foram utilizados vários conceitos teóricos fornecidos pelos orientadores resultando assim a aplicação pronta. Com isso este projeto pode lidar diretamente com o problema proposto que é a falta de estimulo de leitura.*

1. Introdução

Os dados do IDEB (Índice para o Desenvolvimento da Educação Básica) de 2012, divulgados pelo MEC, mostram que as 192.676 escolas de educação básica no país atendem 50.545.050 alunos. Mas, apenas perto de 64 mil delas possuem bibliotecas. Ou seja, não chega a ter uma biblioteca para cada mil alunos. Isso se reflete, é claro no déficit de leitura do brasileiro.[Garcia 2014]

A redução da leitura foi medida até entre crianças e adolescentes, que leem por dever escolar. Em 2011, crianças com idades entre 5 e 10 anos leram 5,4 livros, ante 6,9

registrados no levantamento de 2007. O mesmo ocorreu entre os pré-adolescentes de 11 a 13 anos (6,9 ante 8,5) e entre adolescente de 14 a 17 (5,9 ante 6,6 livros). [Goulart 2012]

Segundo Mariana Carvalho (2012), supervisora da Fundação Educar DPaschoal, que trabalha com incentivo a leitura, Uma razão para a queda no hábito de leitura entre o publico infanto-juvenil é a falta de estimulo da família, se em casa as crianças não encontram pais leitores, reforça-se a ideia de que ler é uma obrigação escolar.

Segundo Murilo Avellar Hingel [de Minas 1999], "A leitura pode nos conduzir por tempo, lugares e acontecimentos que não são os nossos. A leitura é necessária para a construção da cidadania, para a formação de homens livres e aptos a participar da grande obra que é a humanidade."

Diante deste problema, este trabalho tem como intuito, explicar o desenvolvimento de uma aplicação web com foco no estimulo a leitura, aplicando métodos que permeiem o interesse a leitura em crianças. Este documento foi organizado por tópicos macros representando as disciplinas curriculares, incumbidos a estas tem-se uma breve explicação do seu proposito e os sub-tópicos encarregados de especificar as tarefas que foram efetuadas para o bom cumprimento das atividades solicitadas.

2. Desenvolvimento Web

Nos últimos anos, a web tornou-se completamente integrada a malha da sociedade. A web é, atualmente, o principal veiculo para a prestação de serviços, permitindo atingir um número cada vez maior e mais diversificado de usuários e isto torna a demanda por sistemas baseados na web cada vez maior e urgente. Segundo [Conallen 2003], o termo web vem da visão do sistema como um conjunto de nós com links de interconexão. Conforme [Stair 1998], sistema é um conjunto de elementos ou componentes que interagem para se atingir os objetivos. [Oliveira 1992], define sistema como um conjunto de partes que, juntas formam um todo, para exercerem determinadas funções e atingirem determinados objetivos, acrescentando que os componentes de um sistema são as entradas, o processamento e saídas.

Um sistema ou aplicação web é desenvolvido para adicionar funcionalidades de negócio para alcançar o objetivo deste determinado negócio [Conallen 2003]. Nos seus termos mais simples, uma aplicação web é um sistema que permite a seus usuários executar a regra ou lógica do negócio com um navegador web. Um sistema web é um site em que a entrada do usuário no sistema afeta o estado do negócio. Essencialmente, um sistema baseado na web usa um site web como o front-end para aplicação de negócio. Em quase todas as aplicações web mais simples, o usuário precisa fornecer mais do que apenas informações de solicitação de navegação.

A arquitetura de um sistema web é simples e direta. Ela contém os mesmos componentes principais de um site web: um servidor web, uma conexão de rede e navegadores cliente. A inclusão do servidor permite que o sistema gerencie o estado e a regra do negócio.

2.1. Principais características do desenvolvimento WEB

Segundo [Pressman 2002] as seguintes características guiam o processo de sistemas baseados na web:

Imediatismo: aplicações baseadas na web têm um imediatismo que não é encontrado em nenhum outro tipo de software. Isto é, o prazo de colocação no mercado e disponibilização de novas informações de um site pode ser uma questão de semanas. Os desenvolvedores precisam usar métodos para planejamento, projeto, implementação e testes.

Segurança: como as aplicações web estão disponíveis através de acesso à rede, é difícil limitar a população de usuários finais que podem ter acesso à aplicação. A fim de proteger o conteúdo reservado e fornecer modos seguros de transmissão de dados, boas medidas de segurança precisam ser implementadas na infra-estrutura da aplicação propriamente dita.

Estética: uma inegável parte da atração de uma aplicação ou sistema web é o seu aspecto. Quando uma aplicação é projetada para o mercado, para vender produtos e ideias, ou para que os usuários se sintam bem e fiquem à vontade no uso da aplicação a estética pode ter tanto a ver com o sucesso quanto o projeto técnico.

2.2. Principais vantagens do desenvolvimento web

A seguir estão citadas algumas vantagens da utilização dos sistemas que são baseados na plataforma web [Conallen 2003], ressaltando os dados que afetam diretamente a aplicação desenvolvida:

1. Sistemas baseados na web podem ser executados a partir de qualquer navegador da internet, em qualquer lugar do mundo e dispositivo, para isto basta apenas o usuário possuir um sistema que consiga receber páginas web e conexão à internet.
2. Interface HTML reconhecida por uma grande gama de usuários já acostumados com o funcionamento dos navegadores.
3. Atualização dos dados e informações em tempo real para todos os usuários do sistema.
4. Desenvolvimento, manutenção e atualização centralizada da aplicação. Não é necessário sair instalando o sistema em diversos equipamentos diferentes. Basta colocá-lo no servidor para que os usuários obtenham acesso, gerando minimização dos custos, pois em qualquer situação, seja de atualização e ou alteração do sistema basta fazer apenas no servidor e a partir de então todos os usuários do sistema desfrutarão das mudanças efetuadas.
5. Redução dos custos de comunicação, sendo que se existem escritórios dispersos e o sistema de informação se baseia na internet, o custo em conversações pode ser substancialmente reduzido.
6. Exportação de dados entre usuários remotos usando o protocolo HTTP é mais simples e mais fácil do que usar outro protocolo.
7. Não é exigido muita memória e nem poderosos processadores para a execução do sistema nos terminais pois o sistema é todo executado no servidor.
8. Escalabilidade no processamento. Se houver necessidade de aumentar o poder de processamento, basta fazer isto no servidor.

2.3. Principais desvantagens do desenvolvimento web

A seguir estão citadas algumas desvantagens da utilização dos sistemas que são baseados na plataforma web [Conallen 2003], levando em consideração a aplicação que será desenvolvida:

1. Não há uma padronização entre os diversos navegadores e o sistema poderá ser exibido de uma maneira diferente a depender do navegador e da versão do mesmo.
2. O tempo de processamento da execução das tarefas depende da velocidade da conexão, entre cliente e servidor.
3. O sistema depende dos recursos do navegador usado para visualizar a aplicação. Como eles possuem recursos diferentes, existem dificuldades para prever como a aplicação vai se comportar.
4. A curva de desenvolvimento de páginas dinâmicas e formulários com interface HTML para entrada de dados é maior do que em aplicações comuns.
5. A manipulação das variáveis é um trabalho muito mais complicado, tendo em vista a possibilidade que o usuário tem de abrir e fechar janelas e "navegar" para onde bem entender
6. O desenvolvimento se torna mais complicado pois envolve três camadas onde é necessário o servidor de banco de dados, servidor de aplicativos (regras de negócios) que será o servidor de internet e o front-end com diversas validações no próprio navegador.
7. Difícil gerenciamento do estado do cliente no servidor. A natureza sem conexão das comunicações do cliente e do servidor não proporciona um modo fácil do servidor controlar a solicitação de cada cliente e associá-la à solicitação anterior, visto que cada e toda solicitação de página web estabelece e, em seguida, interrompe um conjunto completamente novo de conexões.

3. Interface Homem-Maquina

IHC (Interação Humano-Computador) é uma disciplina que abrange o projeto, implementação e avaliação de sistemas computacionais interativos para uso humano, juntamente com os fenômenos, relacionados computacionais interativos para uso humano, juntamente com os fenômenos relacionados a esse uso [ACM SIGCHI 1992]. A área de IHC tem por objetivo principal fornecer aos pesquisadores e desenvolvedores de sistemas explicações e previsões para fenômenos de interação usuário-sistema e resultados práticos para o design da interface de usuário [ACM SIGCHI 1992].

3.1. Personas

Uma persona é um personagem fictício, um modelo hipotético de um grupo de usuários reais, criado para descrever um usuário típico [Cooper A 2007] (Cooper et al., 2007; Pruitt e Adlin, 2006; Cooper, 1999). Ela é utilizada para representar um grupo de usuários finais durante discussões de design, mantendo todos focados no mesmo alvo. As personas são definidas principalmente por seus objetivos, os quais são determinados num processo de refinamentos sucessivos durante a investigação inicial do domínio de atividade do usuário [Diniz 2010].

Neste projeto foi criada 1 persona, uma criança de 9 anos de idade, a qual será o foco principal do cenário, chamada Samanta, que não tem o hábito de ler.

3.1.1. Objetivos

No momento de criação da persona é importante definir os seus objetivos, os quais, são as representações das motivações que levam o usuário a realizar suas tarefas. Sendo este

uma condição final, ao passo que uma tarefa é um processo intermediário necessário para atingir o objetivo [Diniz 2010]. Logo abaixo segui-se a persona que foi criada para o sistema com seus objetivos pessoais e práticos respectivamente.

Samanta Brito, 10 anos de idade, uma estudante do ultimo ano do ensino fundamental I. Uma criança educada, inteligente, prestativa com seus estudos e bastante comunicativa. Ela tem o hábito, de sempre que terminar de fazer suas atividades escolares, assim como muitas crianças, acessar a internet para assistir vídeos no YouTube, jogar jogos femininos e acessar o Facebook, este último algumas vezes sem premissão ou consciência dos pais. Samanta não tem o hábito da leitura, só lê os textos escolares e as postagens no Facebook, sua desculpa para não ler, é que esta é uma atividade enjoativa, pouco dinâmica e com pouco tempo perde o interesse. Contudo, caso o texto venha acompanhado de uma imagem ou figura, ela consegue se interessar e se divertir, só que mesmo assim acaba cansando desta atividade depois de um certo período.

Como Samanta já está a caminho do ensino fundamental II, seus pais sempre lhe acompanham à uma livraria para que ela venha a se interessar pela leitura, mas como eles não podem efetuar esta tarefa rotineiramente, Samanta acaba por investir seu tempo no Facebook e YouTube com assuntos triviais. Ela gostaria de se entreter e ter gosto pela leitura, para melhorar sua produtividade na escola, sem ter que se deslocar para adquirir um livro e também interagir de certa forma com o livro.

1. Objetivos pessoais:
    (a) aprender mais
    (b) se interessar pela leitura
2. Objetivos práticos:
    (a) não ter que esperar pelos pais para buscar um livro
    (b) poder ler de forma divertida no computador
    (c) ter um livro dinâmico

Tendo a persona e seus objetivos definidos, a equipe pode planejar e definir um cenário de interação, o qual esta persona estará interagindo com o ambiente atual para atingir seus objetivos.

3.2. Cenário de Interação

Cenário se resume na narrativa sobre pessoas realizando uma atividade para alcançar um ou mais objetivos.[Rosson 2002]. O cenário de interação busca especificar em mais detalhes as ações do usuário e as respectivas respostas (feedback) do sistema necessárias para alcançar os objetivos apoiados pelo sistema[Diniz 2010]. No parágrafo a seguir tem-se o cenário de interação do projeto, representando como Samanta iria buscar uma forma de leitura na realidade que se encontra e os atores, que são as pessoas que compõem este cenário.

Atores: Caroline Soares (atendente), Samanta Brito (leitora) e a Marisa Brito (mãe de Samanta).

Samanta e sua mãe se dirigem até uma livraria a qual contem uma série de livros onde elas podem escolher alguns livros, podendo ler naquele ambiente ou pegar emprestado. Para isso, elas precisam inicialmente solicitar uma lista dos livros disponíveis na

livraria a uma atendente. Esta atendente solicita o estilo de livro que elas desejam, por exemplo aventura e depois a idade da criança, caso Marisa tenha interesse em limitar a busca pela faixa de idade. Neste momento a atendente faz uma busca dos livros disponíveis na livraria pelo seu computador, informando o(s) estilo(s) escolhido a idade. No final desta busca, o sistema da livraria entrega uma lista com todos os livros disponíveis.

Depois de tomarem posse desta lista, a qual contem o nome do livro e um breve descritivo sobre ele, Samanta e sua mãe informam o código do(s) livro(s) desejado(s) à atendente, para que ela possa buscar esse(s) livro(s) nas prateleiras.

Quando retorna com o(s) livro(s), a atendente pergunta se Marisa tem cadastro na livraria, caso ela não tenha, a atendente solicita alguns dados pessoais dela como RG, CPF, nome, sobrenome e endereço completo (logradouro, nome da rua, número, bairro, cidade e estado) e pede para ela digitar uma senha de acesso. Depois de ter efetuado o cadastro, atendente faz a reserva do livro no nome de Marisa, em seguida o livro é entregue a elas com um prazo de 1 semana para devolução ou renovação. Caso já tenha cadastro basta informar o número do CPF e depois digitar a senha de acesso para receber o livro.

Após ter criado o cenário de interação e a persona, a equipe está apta a criar a representação da interação como uma conversa entre entre usuário (U) e preposto do designer (D), o qual é representado por toda a equipe.

3.3. Design Centrado na Comunicação

O design centrado na comunicação visa elaborar a metacomunicação entre usuário e designer de modo a evitar rupturas comunicativas durante a interação do usuário com o preposto do designer. Como essa interação é vista como uma conversa, projetar a interação significa definir as conversas que o usuário poderá travar com o preposto do designer para alcançar seus objetivos [Diniz 2010].

Toda conversa tem um tópico, que é o assunto geral por ela endereçado. Essa conversa pode se desdobrar em diálogos, que endereçam subtópicos relacionados ao tópico da conversa [Diniz 2010]. Na Figura 1 é mostrado o dialogo desenvolvido para o cenário apresentado acima, com base no perfil da persona (usuário) e seus objetivos.

3.4. Signos

A utilização de imagens ou ilustrações, como a de um envelope ou impressora, para vincular o conhecimento que o usuário possui da imagem a um comando ou função do sistema é, de forma simplista, a ideia por traz de um signo. Para Peirce[apud Silva 1992], signo é “uma coisa como a imagem de uma impressora) que serve para veicular conhecimentos de uma outra coisa, que representa como a função imprimir” . Em interfaces, um signo é uma mensagem codificada pelo designer para se comunicar com usuário. Existem três tipos de signos: estáticos, dinâmicos e metalinguísticos[RAMOS 2016].

A tabela de signos desta aplicação foi desenvolvida aparte do senário de interação, previamente elaborado com objetivo de recriar ações necessárias em processo de ler um livro na biblioteca. Neste contexto foram retirados palavras chaves ”signos” que iram surgir na interface, bem como também ações e características que estes signos assumiram, a tabela estar anexada neste artigo.

Tópico > Subtópico o Subtópico do Subtópico	Falas e Signos
Obter um livro	U: Eu quero ler um livro.
➢ Informar características do livro	D: Qual estilo do livro? Tem faixa etária? Qual? U: Aventura... tem, 10 anos
➢ Escolher um livro da lista	D: Aqui os livros disponíveis. Escolha um. U: Apenas um? D: Sim. U: Quero esse.
➢ Informe seus dados de acesso para ler	D: Me informe seus dados de acesso para poder ler o livro. U: Não tenho dados de acesso. E agora? D: Vamos cadastrar.
➢ Cadastrar no sistema o Informar dados de cadastro	D: Me diga seu nome, sobrenome, um nome de acesso e uma senha. U: Aqui está, e agora? D: Aguarde um momento.
➢ Acessar o sistema	D: Agora que está cadastrado use seu nome de acesso e senha sempre que quiser um livro. U: Certo
Ler o livro	U: Acessei o sistema e agora? D: Tome o livro que você pediu para ler.

Figure 1. Diálogo entre usuário (U) e designer (D)

3.5. MoLIC

MoLIC, Modeling Language for Interaction as Conversation, é uma linguagem que os designers de IHC podem utilizar para modelar a interação dos usuários com sistemas computacionais, seguindo a metáfora de interação como conversa. Modelagem de interação é modelar a conversa entre o usuário e o preposto (porta -voz) do cristalizado na interface. Molic foi criado em 2003[de Paulo 2003] , em 2005 por Silva e Barbosa , lançaram sua segunda edição[inf.puc rio 2016].

Neste projeto o Molic foi utilizado para conceber, a interação do leitor (usuário), com a interface do software. Com finalidade de prevenção de possíveis rupturas entre a comunicação utilitário – leitor, como é possível perceber nas figuras, questão anexadas neste artigo. A pois o leitor iniciar a comunicação com o designer (aplicação), este guiar para que possa atingir o objetivo do usuário, que é ler o livro (historia interativa). No Molic estar todos os objetos que estarão presente na interface, como também as ligações entre eles, neste ponto possibilita prever as rupturas que poderiam ocorrer, assim viabilizando planejamento das ações necessária, para impedir que haja a ruptura da conversa entre leitor e designer.

4. Banco de Dados

Afirma-se que um banco de dados, é uma estrutura para se armazenar informações. Banco de dados e sistemas de banco de dados são componentes essenciais da vida na sociedade moderna.Geralmente encontra-se diariamente diversas atividades que envolvem alguma interação com um banco de dados. Por exemplo, quando compra-se algo online - um livro ou um brinquedo -, essas atividades envolverão alguém, ou algum programa de computador que acessa um banco de dados [Ramez Elmasri 2011].

4.1. Levantamento de requisitos

Levantamento de requisitos é umas das partes mais importantes do processo que resultará no desenvolvimento de um sistema. Entender aquilo que o cliente deseja ou o que o cliente

acredita que precisa e as regras do negócio ou processos do negócio. Isso é o cerne que move essa importante função que faz parte da engenharia de requisitos [Veríssimo 2007].

As Figuras 2 e 3 mostram quais são os requisitos funcionais e não funcionais, respectivamente, do projeto.

Requisitos Funcioanais		
Referência	**Titulo**	**Descrição**
R1	Cadastrar no sistema	A utilização da aplicação requer o cadastro do usuário no sistema.
R1.1	Logar no Sistema	O usuário só poderá logar no sistema caso esteja cadastrado. Assim podendo utilizar vários recursos do site.
R1.1.1	Ver Livros	Livros estarão visivelmente disponiveis para a leitura após o login do usuário.
R1.1.2	Ler Livros	Logado no sistema, o usuário pode ler quaisquer livros disponiveis.
R1.1.2.1	Comentar	Para comentar sobre o livro, o usuário deverá terminar a leitura do livro especifico.
R1.1.2.2	Enquete sobre o livro	Perguntas sobre a satisfação do cliente sobre o produto especifico.
R1.1.3	Marcar Favoritos	Enquanto logado no sistema, o usuário pode marcar os livros mais interessantes para ele como favoritos.
R1.1.4	Compartilhar	Ao escolher um livro, ou marca-lo como favorito, com suas redes sociais será permitido o compartilhar o livro que o usuário deseja(caso o cadastro dele seja por alguma rede social) especificas.
R1.1.5	Enquete sobre o Site	Para os usuários poderem dizer sobre o nível de satisfação deles sobre o conteúdo e a maneira que foi desenvolvido o site e os livros.

Figure 2. Requisitos Funcionais

Requisitos não funcionais		
Refêrencias	**Titulo**	**Descição**
R1	Banco de dados	A aplicação será feita na "web", por isso, será utilizado o banco MySQL para o armazenamento dos dados tanto dos livros como dos clientes.
R1.1	Armazenamento	Os dados que vão ser armazenados no banco de dados são (adicionar os dados).
R2	Segurança/Confiabilidade	Política de segurança que estabelece algumas regras para os membros da equipe visando a integridade do código.
R2.1	Criptografia	Alguns dados da aplicação terá de ser criptografado para a segurança dos clientes e do site, como por exemplo: login, senha.
R3	Padrões de projeto	Padrões que serão adotados no projeto para facilitar o seu manuseio, desempenho e manutenção.
R3.1	(padrão)	
R3.2	(padrão)	
R4	Portabilidade	O projeto irá rodar na plataforma Web.

Figure 3. Requisitos não Funcionais

4.2. Modelo de entidade e relacionamento - MER

Consiste em mapear o mundo real do sistema em um modelo gráfico que irá representar o modelo e o relacionamento existente entre os dados. Propõe definições e regras para o projeto e a implementação do banco de dados [Siena 2013].

A Figura 4 representa o MER do projeto junto com os seus atributos. A entidade Usuário está se relacionando com a entidade Livros. No caso, um usuário tem a possibilidade de ler vários ou nenhum livro, e um livro pode ser lido por vários ou nenhum usuário.

Cada um deles tem seus respectivos atributos que servirá para identifica-los fazendo consultas no banco. A entidade autor está se relacionando com livros, nesse caso o autor pode escrever nenhum ou vários livros, e um livro pode ser escrito por um ou vários autores.

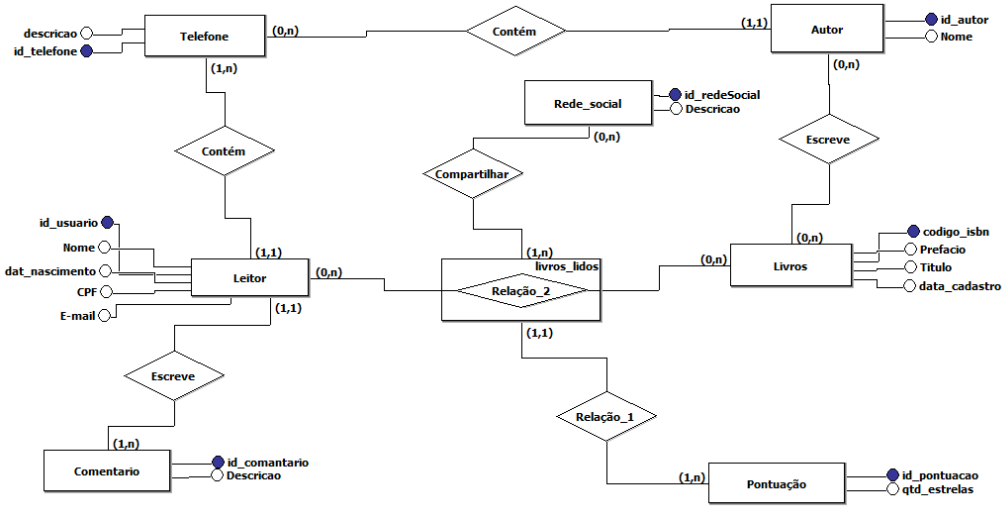

Figure 4. Modelo de entidade e relacionamento

4.3. Modelo Relacional

O modelo relacional foi criado por Edgar F. Codd, nos anos 70 e começou a ser usado com o advento dos bancos de dados relacionais, nos anos 80. A ideia de modelo relacional se baseia no princípio de que as informações em uma base de dados porem ser consideradas como relações matemáticas e que podem ser representadas, de maneira uniforme, através do uso de tabelas onde as linhas representam as ocorrências de uma entidade e as colunas representam os atributos de uma entidade do modelo conceitual.[Siqueira 2014]

Segundo [Siqueira 2014] as relações no modelo relacional são conjuntos de dados vistos como tabelas cujas operações são baseadas na álgebra relacional (projeção, produto cartesiano, seleção, junção, união e subtração) e que manipulam conjuntos de dados ao invês de um único registro, isto é, cada operação realizada afeta um conjunto de linhas e não apenas uma única linha. Da mesma forma, a resposta das operações de consulta são sempre na forma de uma tabela. As operações da álgebra relacional são implementadas por linguagens não procedurais de alto nível, sendo a SQL a linguagem padrão para os bancos de dados relacionais e universalmente usada, tendo sido padronizada pelo ANSI (American National Standard Institute).

Na Figura 5 apresenta o modelo relacional do projeto destacado e seus relacionamentos. Como foi explicado na sessão acima, as entidades usuário e livro se relacionam. Formando assim uma tabela associativa que foi nomeada de livros lidos. Nessa tabela será armazenado a chave primária das entidades usuário e livro, fazendo um relacionamento entre as tabelas.

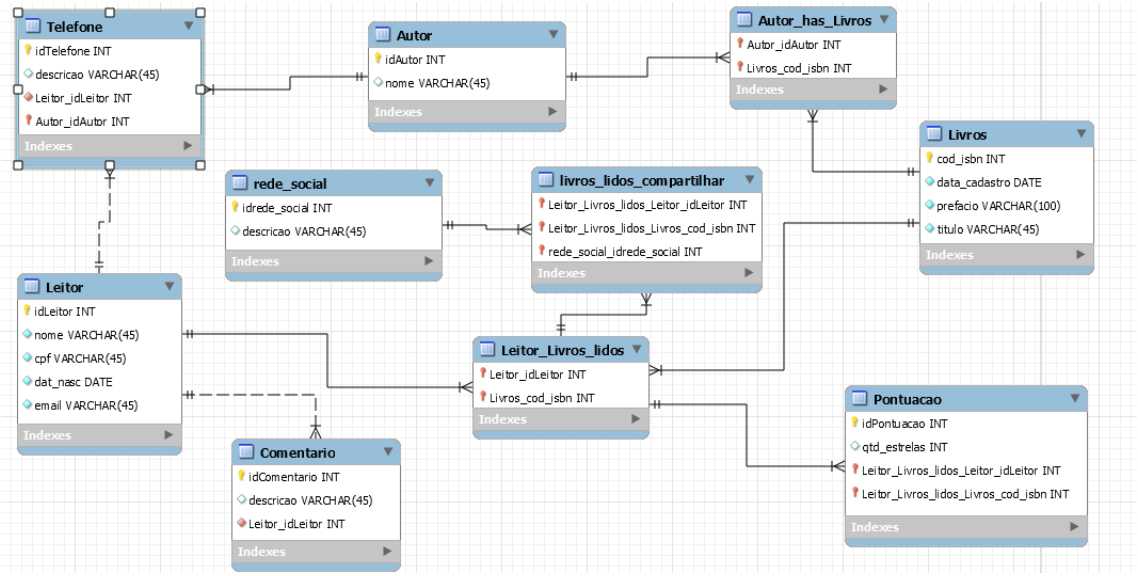

Figure 5. Modelo Relacional

5. Arquitetura de sistemas

Uma arquitetura de sistema é uma representação de um sistema em que existe um mapeamento de funcionalidade para componentes de hardware e software, um mapeamento da arquitetura de software de hardware para a arquitetura de hardware e uma interação humana com esses componentes [Copyright IBM Corp 2006].

A arquitetura de Sistemas de Informação tem como foco principal a análise das necessidades dos usuários dentro de um possível sistema a ser desenvolvido. Para isto, a mesma procura não se aprofundar em detalhes tecnológicos, mas sim se concentrar em que o cliente realmente precisa, levando em conta ainda as características do negócio em que o mesmo está inserido [Groffe 2016].

A arquitetura define como são organizados esses elementos para montar o software e o arquiteto é o responsável pela sua criação. Mas o mais importante é saber quais as possíveis escolhas e que fatores influenciam na sua seleção. Aí entra a caixa de ferramentas do arquiteto (conjunto de possíveis escolhas/soluções), e os requisitos que o software deve atender [Xavier 2013].

Segundo[IBM Corp 2006] o termo arquitetura é, hoje em dia, amplamente utilizado fora de seu sentido tradicional de construção e existem muitas definições de ”Arquitetura” (contexto de sistemas) e ”Arquitetura de sistema (ou sistemas)”:

”Arquitetura de sistemas: A estrutura fundamental e unificadora do sistema definida sob o ponto de vista de elementos do sistema, interfaces, processos, restrições e comportamentos.” [INCOSE 1996].

”A arquitetura de sistema inclui as principais propriedades físicas, estilo, estrutura, interações e finalidade de um sistema.” [Pirbhai 2000].

”Arquitetura: Os conceitos e regras que definem a estrutura, o comportamento semântico e os relacionamentos entre as partes de um sistema; um plano de algo a ser construído. Inclui os elementos que constituem o fato, os relacionamentos entre esses

elementos, as restrições que afetam esses relacionamentos, um foco nas partes do fato e um foco no fato como um todo." [Putman 2001].

No contexto apresentado, arquitetura de sistema pode ser vista de formas diferentes, a depender do que se trata, ou refere-se. Se tratando da arquitetura do sistema está relacionado como ele será montado, ferramentas usadas, lavando em consideração hardware e processos que serão efetuados, bem como a sua interface. Porem partindo do contexto da arquitetura do sistemas prioriza se mais como o software atendera as necessidades do cliente, que pode ser caracterizada pela as regras de negócio do cliente.

5.1. Padrões de Arquitetura de Sistema

Um padrão de arquitetura é um conjunto de decisões para arquitetura que são aplicáveis em um problema de design recorrente. Os padrões arquiteturais expressam formas de organizar a estrutura fundamental do sistema, permitem a construção de uma arquitetura aderente a certas propriedades. O conhecimento de padrões arquiteturais ajuda na definição da arquitetura do sistema [Figueiredo 2016].

5.1.1. Tipos de Padrões Arquiteturais para Sistemas

Padrão de arquitetura, tem como objetivo central isolar ao máximo a camada de apresentação de um sistema. Além disso, em termos de nomenclatura, todas coincidem em suas duas letras iniciais: M e V, que se referem às camadas Model (representação do domínio do sistema) e View (representação gráfica do modelo), respectivamente. Dessa forma, presume-se que a variação entre os padrões é o conceito em torno da outra letra restante [Kawata 2016].

O padrão arquitetural Model-View-Controller (MVC) é uma forma de quebrar uma aplicação, ou até mesmo um pedaço da interface de uma aplicação, em três partes: o modelo, a visão e o controlador [Medeiros 2016].

O MVC inicialmente foi desenvolvido no intuito de mapear o método tradicional de entrada, processamento, e saída que os diversos programas baseados em GUI utilizavam. No padrão MVC, teríamos então o mapeamento de cada uma dessas três partes para o padrão MVC [Medeiros 2016]. O controlador (Controller) que interpreta as entradas do mouse ou do teclado enviado pelo usuário e mapeia essas ações do usuário em comandos que são enviados para o modelo (Model) e/ou para a janela de visualização (View) para efetuar a alteração apropriada [Medeiros 2016].

O modelo (Model) gerencia um ou mais elementos de dados, responde a perguntas sobre o seu estado e responde a instruções para mudar de estado. O modelo sabe o que o aplicativo quer fazer e é a principal estrutura computacional da arquitetura, pois é ele quem modela o problema que está se tentando resolver [Medeiros 2016].

A visão (View) gerencia a área retangular do display e é responsável por apresentar as informações para o usuário através de uma combinação de gráficos e textos. A visão não sabe nada sobre o que a aplicação está atualmente fazendo, tudo que ela realmente faz é receber instruções do controle e informações do modelo e então exibir elas. A visão também se comunica de volta com o modelo e com o controlador para reportar o seu estado [Medeiros 2016].

O padrão arquitetural Model–view–presenter (MVP) é uma derivação do padrão de software model-view-controller (MVC), usado também para construir principalmente interfaces gráficas [Junior 2015].

O apresentador (Presenter) assume a função de mediadora (executada pelo Controller em MVC). Além disso, a View é responsável por manipular os eventos UI (como mouseDown, keyDown, etc.), que era o trabalho da Controller. O presenter age como intermediário entre a view e o model. Ele retira os dados do modelo e retorna para a view. Mas, diferente de típicos MVC, ele também decide o que acontece quando usuário interage com a view [Junior 2015].

O padrão MVVM é denominado de Model-View-ViewModel, foi desenvolvido em 2005 por John Gossman, um dos arquitetos da Microsoft, portanto é muito utilizado por programadores que desenvolvem com esta tecnologia [Junior 2015].

Modelo (Model) Inclui todo o código que implementa a lógica central do aplicativo e define os tipos necessários para modelar o domínio do aplicativo. Essa camada é completamente independente das camadas view e view model [Junior 2015].

Visão (View) define a interface do usuário usando a marcação declarativa. A marcação de vinculação de dados define a conexão entre os componentes específicos da interface do usuário e vários membros de view model [Junior 2015].

Visão Modelo (Visão Modelo) fornece destinos de vinculação de dados para a camada view. Em muitos casos, view model expõe a camada model diretamente ou fornece membros que encapsulam membros específicos da camada model. A camada view model também pode definir membros para controlar os dados que sejam relevantes à interface do usuário, mas não à camada model [Junior 2015].

5.2. Vantagens e Desvantagens dos Padrões Arquiteturais

Favorece o modelo de desenvolvimento incremental, as camadas podem ser facilmente substituídas por equivalentes (Requer interfaces estáveis). Mudanças em uma camada teoricamente só impacta a camada superior, camadas superiores podem ser independentes de plataforma/hardware [Figueiredo 2016].

As desvantagem de estruturar o sistema em camadas está na sua construção, pois não é trivial pode ser difícil identificar quais os serviços elementares das camadas inferiores. Muitas camadas podem comprometer o desempenho do sistema, a requisição tem que trafegar pelas várias camadas até ser atendida [Figueiredo 2016].

6. Arquitetura Montada para o Software

A estrutura montada para o utilitário é MVC e DAO (Objeto de acesso a dados), pois proporciona uma grande vantagem na construção software. Segundo [Figueiredo 2016] esse tipo de padrão possibilita possíveis manutenções futuras no APP, sem causa grandes ou nem impacto nas outras camadas, além de uma melhor organização entre partes do utilitário. Como o sistema está decomposto por três camadas, ou seja, modelo; visão e controle, possibilita ganhos de desenvolvimento e segurança. Pois a equipe trabalha em um mesmo software, mas cada componente do grupo tem foco por nível do programa, enriquecendo agilmente a construção. Desvantagem nesse modelo arquitetural é

que requer camadas bem definidas, como diz [Figueiredo 2016] tornas se mais complicado recognição de quais serviços estão nas camadas inferiores do software. Á uma perda também no desempenho e ocorre aumento no tamanho do utilitário.

Objeto de acesso a dados (ou simplesmente DAO, acrônimo de Data Access Object), é um padrão para persistência de dados que permite separar regras de negócio das regras de acesso a banco de dados[Kotaro Takai 2008].

Este padrão permite criar as classes de dados independentemente da fonte de dados ser um BD relacional. Para isso, encapsula os mecanismos de acesso a dados e cria uma interface de cliente genérica para acessar os dados. Dessa forma, o DAO permite que os mecanismos de acesso a dados mudem independentemente do código que usa o dado[Kotaro Takai 2008].

O DAO é utilizado na camada Molde (modelo), para fazer todas canecões necessária ao banco de dados, sendo responsável por ler e grava dados no banco.

7. Atores Do Software

Atores são usuários e/ou outros meios externos que desenvolvem algum papel em relação ao sistema. Os meios externos são hardwares e/ou softwares que, assim como os usuários, geram informações para o sistema ou necessitam de informações geradas a partir do sistema [Nogueira 2006].

Na Figura 7 esta representado as ações dois atores do software, que são usuário e administrador. Os atores se comunicação entre se, porém o usuário não pode interferir nas ações do administrador, assim também esta para o usuário.

No software há apenas dois autores, sendo estes o Administrador e Usuário, neste meio as ações dos tratantes geram saídas e entradas de informações no utilitário. Na Figura 9.1 o diagrama de caso de uso, há uma atuação entres atores do software.

Ator	Atuação	Ator	Atuação	
USUARIO	Ver Livros	ADMINISTRADOR	Gerir Livro/Leitor	Cadastrar
	Logar Sistema			
	Cadasto Sistema			
	Responder Pesquisa			Deletar
	Marcar Favoritos			
	Compartilhar			Deletar
	Comentar			
	Ler livros			

Figure 6. Quadro de atuação

8. Segurança e Auditoria de sistemas

A auditoria de sistemas de informação visa verificar a conformidade não dos aspectos contábeis da organização, mas sim do próprio ambiente informatizado, garantindo a integridade dos dados manipulados pelo computador. Assim, ela estabelece e mantém procedimentos documentados para planejamento e utilização dos recursos computacionais da empresa, verificando aspectos de segurança e qualidade. O trabalho da auditoria de sistemas acontece com o estabelecimento de metodologias, objetivos de controle e procedi-

mentos a serem adotados por todos aqueles que operam ou são responsáveis por equipamentos de TI e/ou sistemas dentro da organização.[Fonseca 2012]

8.1. Politica de segurança

A Política de segurança da informação, na A EMPRESA, aplica-se a todos os funcionários, prestadores de serviços, sistemas e serviços, incluindo trabalhos executados externamente ou por terceiros, que utilizem o ambiente de processamento da Companhia, ou acesso a informações pertencentes à A EMPRESA. [Basto 2015]

Todo e qualquer usuário de recursos computadorizados da Companhia tem a responsabilidade de proteger a segurança e a integridade das informações e dos equipamentos de informática. [Basto 2015]

A politica de segurança do projeto abordado está em um documento externo.Para conseguir acessar esse documento, clique nesse link:

8.2. Criptografia

Trata-se de um conjunto de conceitos e técnicas que visa codificar uma informação de forma que somente o emissor e o receptor possam acessá-la, evitando que um intruso consiga interpretá-la. Para isso, uma série de técnicas são usadas e muitas outras surgem com o passar do tempo.[Alecrim 2005]

Há dois tipos de chaves criptográficas: chaves simétricas e chaves assimétricas: Chave simétrica e Assimétrica. As chaves simétricas, é um tipo de chave mais simples, onde o emissor e o receptor fazem uso da mesma chave, isto é, uma única chave é usada na codificação e na decodificação da informação. Já a assimétrica, também conhecida como ”chave pública” trabalha com duas chaves: uma denominada privada e outra denominada pública. Neste método, um emissor deve criar uma chave de codificação e enviá-la ao receptor. Essa é a chave pública. Uma outra chave deve ser criada para a decodificação. Esta, a chave privada, é secreta.[Alecrim 2005]

O processo clássico para fazer a criptografia é guardar um hash (chamado também de digest nesse caso) da senha do usuário, usando algum algoritmo de hash unidirecional. Isso pode ser feito utilizando uma chamada de função na query do banco de dados (como MD5()no MySQL), ou, o mais utilizado para não ter de trafegar a senha entre o servidor web e o banco de dados: com o MessageDigest do javax.security.[Ricardo Nakamura 2010] Através dessa classe você pode facilmente gerar o hash de uma senha, conforme o exemplo abaixo:

```
MessageDigest algorithm = MessageDigest.getInstance("MD5");
byte messageDigest[] =
   algorithm.digest("senha".getBytes("UTF-8"));
```

Neste projeto o hash 5 (MD5) será utilizado para fazer a criptografia do sistema de login e senha.

MD5 (Message-Digest algorithm 5) é um algoritmo de “hash” de 128 bits unidirecional. O MD5 é um método que é possível transformar uma palavra num MD5 hash, porém com um MD5 hash nunca é possível encontrar a palavra que o originou.[do Nascimento 2012]

O MD5 é muito útil quando você quer arquivar senhas. Pois uma senha como por exemplo “12345678” gera o MD5 hash “25d55ad283aa400af464c76d713c07ad” (sempre sem aspas) porém se alguém descobrir o MD5 hash, será incapaz (teoricamente) de encontrar a senha que o gerou.[do Nascimento 2012]

9. Desenvolvimento do sistema

Este capítulo trata do desenvolvimento do sistema web. São apresentados os requisitos do sistema, a especificação, e a implementação do sistema.

9.1. Funcionalidades do sistemas

Para a especificação do sistema foi utilizada a ferramenta Astah Community. A seguir é apresentado o diagrama de caso de uso, a partir do qual se pode ter uma visão macro das funcionalidades do sistema. Conforme [Bezzera 2002], os diagramas de caso de uso (DCU) correspondem a uma visão externa do sistema e representa graficamente os atores envolvidos, casos de uso e relacionamento entre esses elementos. O diagrama de caso de uso tem objetivo de ilustrar em um nível alto de abstração quais elementos externos interagem com que funcionalidades do sistema. O diagrama de casos de uso apresentado na Figura 5 contém as funcionalidades essenciais do sistema realizadas pelos seus atores.

A seguir são descritos brevemente as principais funcionalidades:

1. cadastro sistema: ao acessar o sistema o usuário terá que realizar um cadastro no sistemas.
2. logar sistema: após realizar o cadastro o usuário poderá logar no sistema.
3. Ver livro: devidamente logado no sistema o usuário terá a disposição uma biblioteca digital contento os livros.
4. ler livros: usuário poderá ler qualquer livro disponibilizado pelo sistema.
5. responder pesquisa: usuário terá a disposição questionários de avaliação do sistema.
6. marcar favoritos: usuário após escolher um livro poderá marca alguns como favoritos.
7. compartilhar: o sistema disponibilizará a opção de poder compartilhar os livros.
8. comentar: para comentar sobre o livro, o usuário deverá terminar a leitura do livro especifico.
9. gerir livros/leitor: o sistema ira fazer o gerenciamento do cadastro dos livros e dos usuários cadastrados.

9.2. Apresentação do sistema

O sistema será baseado na web, e após informado o endereço na internet é aberto a página da aplicação.

Quando o usuário acessa a pagina inicial, o mesmo terá que realizar o login no sistema para visualizar a biblioteca de livros disponíveis, realizar algumas alteração no seu cadastro.

Ao tentar efetuar o acesso, é feita uma validação do usuário e da senha informados. Caso o usuário não tenha cadastro, ele poderá o utilizar a opção de criar uma conta.

Ao efetuar o acesso com sucesso (login e senha válidos), o menu na lateral esquerda apresenta as opções (funcionalidades) para o perfil do usuário que está logado. No menu de opções o usuário terá as seguintes opções:

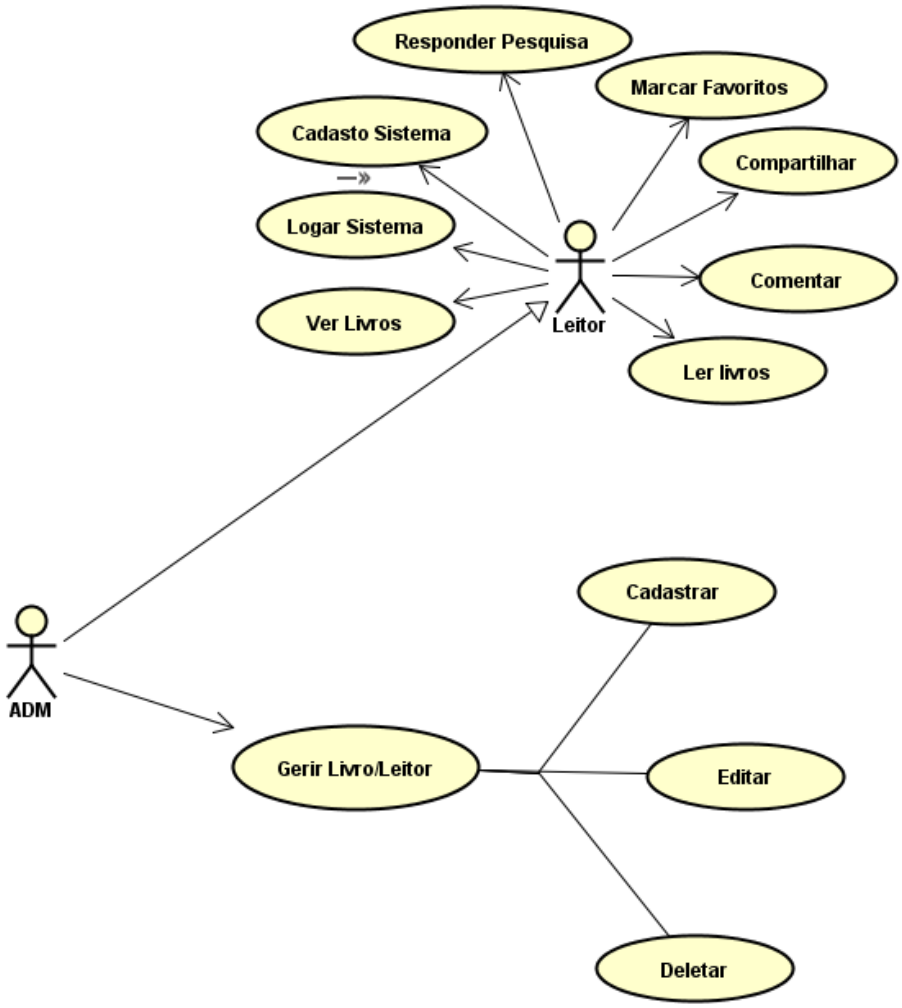

Figure 7. Diagrama de casos de uso

1. Home: onde o sistema retorna na tela inicial;
2. Componentes: onde o usuário poderá visualizar a equipe de produção do sistema.
3. Biblioteca de livros: onde consta toda a edições de livros do sistema.
4. Livros futuros: constará os futuros livros que serão adicionado na biblioteca de livros.
5. Contato: local onde o usuário poderá enviar reclamações, sugestões outro tipo de informação referente ao sistema.
6. Sub-menu: onde constará mas opção do sistema, como, responder a pesquisa, alterar dados cadastrais, compartilhar e outros.

References

ACM SIGCHI, H. (1992). Curricula for human-computer interaction. Disponível: http://www.acm.org/sigchi/. Acesso: 15 Set. 2016.

Alecrim, E. (2005). Criptografia. Disponível: http://www.infowester.com/criptografia.php. Acesso: 07 Out. 2016.

apud Silva, B. (1992). *MoLIC – modelagem da interação*.

Basto, F. (2015). Politica da segurança da informação – como fazer ? Disponível: http://analistati.com/politica-de-seguranca-da-informacao-como-fazer/. Acesso: 20 Set. 2016.

Bezzera, E. (2002). *Principios de Analise e Projeto de Sistemas com UML*. São Paulo.

Conallen, J. (2003). *Desenvolvendo aplicações web*. Campus, Rio de Janeiro.

Cooper A, Reimann R, C. D. (2007). *About Face 3: The Essentials of Interaction Design*. John Wiley & Sons, New York.

Copyright IBM Corp, . (2006). *Conceito Arquitetura de Sistema*.

de Minas, L. (1999). In da Educação de Mina, S., editor, *Tempo Escolar: Hora de Refletir, Planejar e Construir a Escola Sagaran*, page 132. Lições de Minas.

de Paulo, B. (2003). *MoLIC – modelagem da interação*.

Diniz, Simone Junqueira Barbosa & Santana, B. d. S. (2010). *Interação Humano-Computador*.

do Nascimento, M. (2012). Um pouco mais sobre o hash md5 para o ccna. Disponível: http://www.dltec.com.br/blog/cisco/um-pouco-mais-sobre-o-hash-md5-para-o-ccna/. Acesso: 26 Out. 2016.

Figueiredo, E. (2016). *Padrões Arquiteturais*.

Fonseca, G. (2012). Auditoria de sistemas de informação – conheça mais sobre o assunto. Disponível: https://www.profissionaisti.com.br/2012/04/auditoria-de-sistemas-de-informacao-conheca-mais-sobre-o-assunto/. Acesso: 22 Set. 2016.

Garcia, M. (2014). Interesse pela leitura no brasileiro é muito pequeno. Disponível: http://professormarcusgarcia.com.br/brasileiro-le-pouco/. Acesso: 07 Set. 2016.

Goulart, N. (2012). Hábito de leitura cai no brasil, revela pesquisa. Disponível: http://veja.abril.com.br/educacao/habito-de-leitura-cai-no-brasil-revela-pesquisa/. Acesso: 28 ago. 2016.

Groffe, R. J. (2016). *Arquitetura de sistemas de informação uma visão geral*.

IBM Corp, . (2006). *Conceito Arquitetura de Sistema*.

INCOSE, I. . (1996). *Definição de linha de base aprovada pelo Grupo de Trabalho de Arquitetura de Sistemas*. Boston, MA.

inf.puc rio, . (2016). *MoLIC*.

Junior, B. (2015). *Angular MVC,MVVM, MVP ou MVW*.

Kawata, H. (2016). *Os padrões de arquitetura MVC MVP e MVVM no delphi*.

Kotaro Takai, O. (2008). *Persistência de Objetos*.

Medeiros, H. (2016). *Introdução ao padrão MVC*.

Nogueira, A. (2006). *Atores*.

Oliveira, D. d. P. R. (1992). *Sistemas de informações gerenciais*. Atlas, São Paulo.

Pirbhai, D. H. &. P. H. &. I. (2000). *Process for System Architecture and Requirements Engineering*. Dorset House Publishing.

Pressman, R. S. (2002). *Engenharia de software*. McGraw-Hill.

Putman, Janis& Hall, P. (2001). *Architecting with RM-ODP*. Referencia o ISO/IEC 10746-2: Information Technology - Open Distributed Processing - Reference Model: Foundations como a origem.

Ramez Elmasri, S. N. (2011). *Sistema de banco de dados*. Pearson, São Paulo.

RAMOS, M. (2016). *MoLIC – modelagem da interação*.

Ricardo Nakamura, T. F. (2010). Guardando senhas criptografadas em java. Disponível: http://blog.caelum.com.br/guardando-senhas-criptografadas-em-java/. Acesso: 07 Out. 2016.

Rosson, M.B. & Carrol, J. M. (2002). Scenario-based development of human-computer interaction. CA: Morgan Kaufmann Publishers.

Siena, R. (2013). Tecnologias e linguagens de banco de dados. Disponível: http://pt.slideshare.net/professor-rade. Acesso: 16 Set. 2016.

Siqueira, F. (2014). Banco de dados —. Disponível: https://sites.google.com/site/uniplibancodedados1/aulas/modelo-relacional. Acesso: 16 Set. 2016.

Stair, R. M. (1998). *Princípios de sistemas de informação*. LTC – Livros Técnicos e Científicos Editora S.A, Rio de Janeiro. Tradução de Maria Lúcia Lecker Vieira e Dalton Conde de Alencar.

Veríssimo, R. (2007). Levantamento de requisitos e mapeamento de processos. Disponível: http://www.baguete.com.br/artigos/296/ricardo-verissimo/05/11/2007/levantamento-de-requisitos-e-mapeamento-de-processos. Acesso: 15 Set. 2016.

Xavier, K. (2013). *O que e arquitetura de software*.