JAVA WEB: UM ESTUDO COMPARATIVO ENTRE FRAMEWORKS ACTION E COMPONENT BASED

Juliano da Costa Nogueira[(1)]; Wellison Carlos Evaristo[(2)]; Leandro Duarte Pereira[(3)];

[1]Estudante, Fundação de Ensino e Pesquisa de Itajubá, Sistemas de Informação, wellisoncarlos2013@hotmail.com;[2] Estudante, Fundação de Ensino e Pesquisa de Itajubá, Sistemas de Informação, julianogueira_wb@live.com;[3] Professor, Fundação de Ensino e Pesquisa de Itajubá, Sistemas de Informação, leandro.pereira@fepi.br;

RESUMO

O presente artigo tem por objetivo elencar as facilidades e benefícios da utilização de frameworks de desenvolvimento para construções de aplicações java web, em especifico, os frameworks baseado em ação e componentes, Spring MVC e JSF respectivamente. Foram realizadas revisões da bibliografia existente para a coleta de dados que evidenciem e que justifiquem a adoção destes frameworks nos processos de desenvolvimento de softwares, bem como as características especificas de cada um dos mesmos.

Palavras-chave: Framework, java web, action e component based.

INTRODUÇÃO

Atualmente, um dos maiores desafios de um sistema de informação é assegurar que a informação possua qualidade e agilidade. Conforme Vicki (2011 apud JUNIOR, SILVA, 2011, p.12), "independente do tamanho, as organizações necessitam dos Sistemas de Informação para reagir aos problemas e oportunidades do ambiente de negócio".

Conforme o enredamento do sistema aumenta, os desafios para o gerenciamento de todos os arquivos e objetos surgem neste contexto. Sendo assim, faz se necessário a utilização de soluções que já foram utilizadas, como frameworks, para simplificar e tornar mais ágil o desenvolvimento do projeto (SOUZA, 2004).

Atualmente, existem diversos frameworks passíveis de serem utilizados na implementação de aplicações web. No contexto da Java Enterprise Edition – JEE, podemos citar: JavaServer Faces, Tapestry, Spring, Grails,Wicket, Jboss Seam, Oracle ADF e Struts, utilizando a tecnologia java (TEIXEIRA, 2008).

Devido à essa grande disponibilidade de opções, é preciso que se conheça qual o framework mais adequado a um dado domínio de aplicações. Para a realização do estudo comparativo, escolheu-se os frameworks Spring MVC e JSF, pois segundo Souza (2004), são largamente utilizados por desenvolvedores do mundo todo.

MATERIAL E MÉTODOS

O estudo engloba uma revisão descritiva com análise qualitativa dos frameworks de desenvolvimento java para web, Spring MVC e JSF. Foram realizadas pesquisas de artigos e livros para uma melhor abordagem do assunto visando uma análise justa, precisa e explicativa das ferramentas supramencionadas.

RESULTADOS E DISCUSSÃO

JavaServer Faces (JSF) é a classificação de um framework MVC baseado em componentes da plataforma Java Enterprises Edition (JEE) que proporciona a facilidade no desenvolvimento de aplicações web. Segundos os dados levantados, a primeira versão - 1.0 foi lançada em 2004, mas somente em 2006 na versão 1.2 o framework foi anexado ao JEE como uma especificação oficial da plataforma.

A tecnologia JSF constitui de um conjunto de API's para manusear componentes de interfaces com o usuário, eventos, validações de entrada, acessibilidade e internacionalização. Sua arquitetura define a separação entre a lógica da aplicação e apresentação, garantindo que cada membro do time de desenvolvimento das aplicações web se concentrem em sua parte no processo de desenvolvimento. (ORACLE, 2015).

Silveira et. al (2012) afirma que o JSF possui um ciclo de vida misto e eficiente para as requisições e componentes.
Cordeiro (2012) define as 6 fases do ciclo de vida do JSF a seguir:

• *Restore View*: A primeira fase é responsável por construir ou restaurar a árvore de componentes correspondente à tela.

• *Apply Request Values*: Esta fase é onde o JSF obtém os valores informados pelo usuário e insere nos respectivos componentes.

• *Validate*: A terceira fase é responsável por converter os valores vindos da requisição para seus respectivos tipos para, posteriormente, validá-los de acordo com as restrições.

• *Update Model*: Uma vez que os dados já foram convertidos e validados, o JSF então aplica esses valores dentro do modelo.

• *Invoke Application*: É nesta fase que as lógicas e regras de negócio da aplicação são executadas.

• *Render Response*: A última fase do ciclo de vida é quando o JSF renderiza a resposta da requisição na tela do usuário.

No início da especificação do JavaServer Faces, uma das principais promessas era a possibilidade de escrever a interface de usuário em uma só linguagem e poder executá-la em um navegador comum (HTML), em um celular (WML) ou em Flash. No entanto, essa característica pouco se acentuou no mercado, pois, na maioria das vezes, a maior necessidade das empresas era
apenas gerar páginas HTML para navegadores comuns. (SILVEIRA et al, 2012).
De acordo com Coelho (2013), nas primeiras versões do JavaServer Faces, a principal tecnologia para exibir informações para o usuário era o JSP (JavaServer Pages). A partir da versão 2.0, porém, surgiu um novo template para
gerar páginas dinâmicas denominado Facelets.
A característica de escrever páginas HTML utilizando os componentes do JSF permite que o framework cuide de alguns detalhes trabalhosos da Web, permitindo, por exemplo, que o valor de um campo seja automaticamente convertido para o formato correto ou que, ao clicar em um botão, um
evento seja disparado para executar código Java em uma classe. (SILVEIRA at. al 2012)

Silveira et. al (2012) salienta que o código HTML gerado pelo JSF pode tornar qualquer otimização ou modificação complexa e custosa. Ao invés de uma abordagem mais livre no design e experiência de usuário, o front-end (onde ocorre a interação entre o usuário e os componentes da tela) é, no geral, adaptado para as limitações do framework.
Silveira et. al (2012) explica que, para que o JSF possa controlar seus componentes, eles devem ser organizados em uma estrutura de dados do tipo árvore. O estado de cada componente da árvore é armazenado e, a cada requisição, o framework se encarrega de atualizar as informações da árvore.
De acordo com Cordeiro (2012), o JSF armazena na memória a árvore utilizada para renderizar determinada tela. Uma vez que o framework tenha em mãos a árvore de componentes, cada componente será comparado com os atributos da requisição. Com essas informações, o JSF inicia a fase de validação dos dados enviados para verificar se são compatíveis com os disponíveis, caso contrário, a requisição se torna inválida. Esse comportamento define o JavaServer Faces como um framework stateful.
Em JSF, a comunicação entre o cliente e o servidor se dá através de classes Java denominadas Managed Beans, que podem ser consideradas como a camada controladora (Controller) da aplicação, seguindo o modelo MVC. Essas classes são responsáveis por responder aos eventos disparados
pelos componentes da View. De acordo com Cordeiro (2012), um Managed Bean não passa de uma simples classe, cuja
função é ter uma relação dos componentes contidos na tela.
Cordeiro (2012) afirma ainda que para acessar um método no Managed Bean, alguns componentes do JSF definem actions
(ações), que devem ser passadas através de EL (ExpressionLanguage), indicando qual método de qual Managed Bean deve ser executado.
Para controlar o ciclo de vida dos objetos e diminuir o acoplamento entre as classes Java, o JSF utiliza um framework de injeção de dependências e inversão de controle.
Silveira et. al (2012) explica que padrões como Inversão de Controle e Injeção de Dependências ajudam a manter os dois
princípios básicos da orientação a objetos: alta coesão e baixo acoplamento. Com o objetivo de facilitar esse trabalho, sugiram diversos frameworks de injeção de dependências,

como por exemplo o Spring, o PicoContainer e o Google Guice, além da especificação CDI (Context and Dependency Injection).

O lançamento da especificação CDI para a plataforma Java EE 6, cuja implementação de referência é o Weld, trouxe um poderoso container de injeção de dependências totalmente integrado ao JSF e às outras especificações da plataforma Java EE.

Silveira at al. (2012) destaca que o CDI é uma ferramenta completa, trazendo novas funcionalidades avançadas de injeção que antes haviam sido pouco exploradas em outras especificações da plataforma, como o EJB (Enterprise JavaBeans).

Segundo Cordeiro (2012), para que o JSF realize a conversão das Strings recebidas na requisição em objetos do tipo correto, são definidos conversores. O framework oferece conversores para todos os tipos básicos do Java, como por exemplo Integer, Float, Double, Date e Boolean. Para alguns tipos de dados, o JSF já utiliza o conversor por padrão, porém quando a conversão é mais complexa, é possível utilizar o conversor explicitamente de acordo com a necessidade, como é o caso dos conversores de número e data.

Além dos conversores padrões do JSF, é possível criar conversores customizados para cada tipo de dado. De acordo com Coelho (2013), a necessidade de criar conversores específicos é uma tarefa bastante trivial em projetos JSF.

O framework Spring foi apresentado ao publico em 2004 através do livro *Expert One-To-One J2EE DevelopmentWithout EJ*, de Rod Johnson.

Um dos principais diferenciais do Spring em seu lançamento foi trazer para o desenvolvedor recursos de computação corporativa que até então só eram oferecidos por servidores de aplicação pesado, com isto, temos como requisito básico para seu funcionamento apenas o JVM, ao invés de servidor de aplicações ou *Servlets*. (WEISMANN, 2012).

Um ponto importante que deve ser mencionado é o fato do Spring, ao contrário de um container EJB, não ser uma solução tudo ou nada. Você usa apenas os componentes que interessam ao seu projeto, podendo ignorar tranquilamente aquilo que não precisa. Consequentemente, temos como resultado uma arquitetura bem mais enxuta e fácil de trabalhar.

A seguir discorremos acerca de alguns componentes que compõe o framework Spring, tais como: container; acesso a dados até chegarmos ao ponto principal: o Spring MVC.

Para trabalharmos com o Spring, temos um módulo obrigatório dentro dos componentes deste framework, denominado *Container*. Os módulos *core* e *beans* definem o núcleo do framework onde serão implementados o suporte à injeção de dependecias e inversão de controle.

No módulo *context* está a implementação do *ApplicationContext*, onde há dois tipos de *containers:* BeanFactory e ApplicationContext.

O BeanFactory consite numa implementação sofisticada do padrão Factory, baseada em configuração. Apenas o suporte básico a IoC e DI é fornecido na primeira versão. Na segunda versão, baseada no *BeanFactory*, oferece recursos corporativos mais avançados, como exemplos o gerenciamento de recursos e internacionalização.

A utilização deste *container* é desencorajada pelo próprio time de desenvolvimento do Spring. Apesar de raro, seu uso é recomendado em ambientes computacionais extremamente restritos, como applets ou dispositivos móveis. (WEISMANN, 2012).

Temos também o módulo *Expression Language* (SpEL). Segundo Weismann (2012), uma forma bastante utilizada na configuração do Spring é utilizando documentos XML. A SpEL fornece uma linguagem próxima a EL que utilizadas com JSPs, no entanto, voltada para a configuração do container. Os arquivos de configuração tornam-se vivos, pois a partir de sua utilização é possível determinar valores de configuração em tempo de execução, ao invés de configuração.

O Spring oferece suporte às principais tecnologias de persistência adotadas por desenvolvedores Java. Há suporte para JDBC, ORMs como Hibernate, iBatis, JPA, JDO e OXM. O suporte a estas tecnologias se da através de templates, que garantem a redução significativa da quantidade de código repetidos, ou seja, de infraestrutura, que precisamos escrever nestas situações, como por exemplo abrir e fechar conexões, iniciar e finalizar transações, entre outras.

O framework Spring implementa o padrão MVC de uma forma altamente produtiva: o Spring MVC.

Este módulo não surgiu inicialmente de forma planejada, mas sim conforme a equipe de desenvolvimento foi percebendo que o Struts, que era um framework de respeito e que dominava o mercado, apresentava um conjunto de limitações que dificultavam bastante a divisão de interesses entre as camadas MVC.

O componente responsável por orquestrar o funcionamento do Spring MVC é o Dispatcher Servlet, que é a implementação do padrão *Front Controller*, largamente utilizado na escrita de frameworks voltados para criação de aplicações web. Este padrão objetiva fornecer um ponto de entrada central para todas as requisições que são direcionadas à aplicação. Portanto, sua função é interpretar estes requisitos e decidir qual o componente fara seu processamento e eventual retorno para o usuário. (WEISMANN, 2012).

Tudo começa no momento em que uma requisição chega ao Dispatcher Servlet. Esta requisição terá sua assinatura analisada e enviada ao Mapeador de requisições (HandlerMapping), - componente responsável por descobrir qual controlador deve ser acionado.

Com o controlador alvo obtido, este é executado e um nome lógico do template de visualização a ser renderizado como resposta ao usuário (view) é retornado ao Dispatcher Servlet e o conjunto de variáveis (model) que serão expostas nesta renderização.

Com o nome lógico da visualização e o modelo obtidos, entra em cena o Gerenciador de visualização (Vi e w Res ol ver), que possui a função de, com base no nome da view, retornar ao Servlet Dispatcher qual elemento de visualização - mais comumente uma página JSP - deverá ser renderizada de volta ao usuário que fez a requisição. (WEISMANN, 2012).

CONCLUSÕES

Em virtude dos fatos mencionados, tanto o JSF quanto o Spring MVC são ferramentas muito importantes e de profunda importância para o bom desenvolvimento de uma aplicação Web, mais é claro, tais quais com suas características. O JSF que é um framework baseado em componentes formado por um conjunto de API's para manusear componentes de interfaces com o usuário, eventos, validações de entrada, acessibilidade e internacionalização, e o Spring MVC que é um framework baseado em ações que em geral é muito útil para qualquer tipo de aplicação web, inclusive as que o JSF se propõe a resolver, você usa apenas os componentes que interessam ao seu projeto, podendo ignorar tranquilamente aquilo que não precisa. Consequentemente, temos como resultado uma arquitetura bem mais enxuta e fácil de trabalhar. Com isso faz-se necessário que o desenvolvedor selecione o framework que melhor atenda sua aplicação pois podemos dizer que os frameworks *component based* são mais centrados nas *views* (com seus componentes que mapeiam o modelo e os dados do usuário), enquanto os *action based* são mais centrados nos *controllers* (que recebem parâmetros via *request*).

REFERÊNCIAS

COELHO, H. **JSF Eficaz: As melhores práticas para o desenvolvedor web Java**. Casa do Código, 2013.

CORDEIRO, G. **Aplicações Java para a web com JSF e JPA**. Ed. Casa do Código, 2012. Fórum de discussões DevMedia. Disponível em: <http://www.devmedia.com.br/forum>. Acessado em: 10.ago 2017.

JUNIOR, I. B. R. S; SILVA, L. H. **Sistemas de Apoio a cooperativas de crédito**. 2011. 93 f. Trabalho de Conclusão de Curso – FEPI – Centro Universitário de Itajubá, 2011.

ORACLE**, JavaServer Faces Technology Overview.** Disponível em: <http://www.oracle.com/technetwork/java/javaee/overview-140548.html>. Acessado em: 10 ago. 2017.

SILVEIRA, P. et al. **Introdução à Arquitetura e Design de Software: Uma Visão Sobre a Plataforma Java**. Rio de Janeiro: Elsevier, 2012.

SOUZA, M. V. B. **Estudo comparativo entre frameworks java para construção de aplicações web**. 2004. 57 f. Universidade Federal de Santa Maria, 2004. Disponível em: <http://www-usr.inf.ufsm.br/~marvin/monografia.pdf>. Acesso em: 10 abr. 2017.

TEIXEIRA, M. **Estudo da Utilização de Frameworks no Desenvolvimento de Aplicações Web**. 2008. 52 f. Universidade Estadual do Oeste do Paraná – Campus de Cascavel, 2008.

WEISSMANN, H. L. **Vire o jogo com Spring Framework**. São Paulo: Casa do Código, 2012.

VIII CONGRESSO DE INICIAÇÃO CIENTÍFICA DA FEPI
PESQUISA CIENTÍFICA E DESENVOLVIMENTO SOCIAL
18 de outubro de 2017

INVENTÁRIOS DE ATIVOS MÓVEIS:
Um software de contagem e consultas RFID – IntellInvent

Caio da Silva Ribeiro [(1)]; Felipe Correa Pinto [(2)]; João Paulo Chaves Barbosa [(3)]

[1]Graduando, Fundação de Ensino e Pesquisa de Itajubá, Sistemas de Informação, caio.wb@hotmail.com;
[2]Graduando, Fundação de Ensino e Pesquisa de Itajubá, Sistemas de Informação, felipe_correapinto@hotmail.com;
[3]Docente, Fundação de Ensino e Pesquisa de Itajubá, Sistemas de Informação, joao.chaves@fepi.br;

RESUMO

O presente artigo tem como objetivo demonstrar como a utilização de tecnologias podem ser importantes na realização de tarefas de uma organização. O enfoque principal será na utilização de identificação por rádio frequência (RFID) na realização da tarefa de controle patrimonial de uma organização, melhorando sua assertividade e eficiência. Foram realizadas revisões da bibliografia existente para a coleta de dados que justifiquem a realização do controle patrimonial pelas organizações e também como acontece o funcionamento da tecnologia RFID.

Palavras-chave: Controle Patrimonial, RFID, identificação por rádio frequência, Tecnologia da Informação, Inventário.

INTRODUÇÃO

As tecnologias da informação são grandes aliadas na composição das organizações no contexto atual da economia mundial. Sendo que cada vez mais, conforme Lucas (2006), assumem papéis fundamentais e estratégicos no dia a dia das empresas para aumentar sua vantagem competitiva. Nesse cenário, surge uma demanda cada vez maior pelo desenvolvimento de novas ferramentas da tecnologia da informação, para auxiliar nos processos organizacionais e tornar mais eficientes as atividades gerenciais e operacionais.Esse caminhar global rumo à tecnologia da informação impulsiona o mercado a se preocupar com a qualidade e a eficácia das aplicações, bem como à segurança e a veracidade das informações, pois, ainda segundo Lucas (2006, p. 2) "A tecnologia permite à organização melhorar significativamente seu modelo de negócio e alterar sua estrutura". Sendo assim, toda organização deve passar a valorizar a informação como recurso fundamental à sua sobrevivência. É pela gestão da informação que as organizações tornam-se estruturadas, competitivas e aptas a responder às mudanças demandadas pelo cenário mundial (BEUREN; MARTINS, 2001).A demanda de confiabilidade e eficiência das informações também se aplica à gestão patrimonial das organizações, tarefa essa que é de suma importância em suas atividades, pois trata-se diretamente do controle de seus ativos e deve ser bem gerido para que não haja perdas, prejuízos ou déficits.

MATERIAL E MÉTODOS

O estudo envolve uma revisão da importância da realização do controle de ativos moveis por parte de uma organização e também conceitos da tecnologia RFID, e sua forma de aplicação. Foram utilizados livros, artigos e citações de empresas conceituadas em áreas relacionadas, para que o assunto pudesse ser abordado da melhor maneira possível, tendo em vista enfatizar como a tecnologia pode ser uma ferramenta muito útil no dia-a-dia das organizações.

RESULTADOS E DISCUSSÃO

Pozo (2015) comenta que os recursos patrimoniais das organizações são vitais para que ela possa desenvolver e oferecer seus produtos e serviços para terceiros, com isso, para o sucesso de uma organização é preciso que haja controle e manutenção de seus ativos móveis.
Todos os itens que colaboram de alguma forma para que a organização possa ofertar seus produtos e serviços, podem ser classificados como recursos patrimoniais. Esses recursos têm por finalidade fazer com que a satisfação do cliente seja atendida (POZO, 2015). Controle dos recursos patrimoniais, pode ser

definido como conjunto de atividades que por meio de relatórios e registros garantem dados referentes à existência, identificação, quantidade, localização, condições de uso e histórico dos bens patrimoniais, a partir de sua inserção até sua remoção do patrimônio de uma organização (TORRES; MARTINS, 2003).Diante dessa situação podemos perceber o quão importante e fundamental é a realização do controle patrimonial por parte das organizações, mesmo diante desse cenário, muitas organizações ainda realizam seu inventariado de forma pouco eficiente, podendo ser manualmente e em algumas empresas é empregado a utilização de códigos de barras, no entanto esses métodos não são nada eficientes e não garantem a veracidade das informações obtidas (RIBEIRO, 2016).

As organizações estão em uma busca constante da melhoria de suas operações, visando cada vez mais o lucro. Para que esse objetivo possa ser alcançado, as tecnologias e os sistemas de informação estão entre as principais ferramentas utilizadas (LAUDON; LAUDON, 2013).

Tendo isso em mente, a utilização da tecnologia de rádio frequência na realização do controle patrimonial, poderia fazer com que a organização realizasse a verificação de seus ativos móveis em menos tempo e alocando menor número de pessoas para essa atividade.

Essa tecnologia já é utilizada nos dias de hoje nos mais diversos tipos de aplicação, desde rastreamento de produtos até identificação de objetos, com isso ela se encaixa perfeitamente no nosso cenário de estudo.

A identificação por rádio frequência ou mais conhecida como RFID, consiste basicamente em equipamentos eletrônicos, que emitem e recebem ondas magnéticas ou eletromagnéticas, fazendo assim com que sejam acessados dados que estão armazenados em um microchip que está acoplado a uma antena, desta maneira, é possível a identificação dos objetos que possuam um identificador RFID fixado nele (CHIESA, et al., 2002).

Booman e Hanson (2006) afirmam que um sistema RFID é composto por quatro elementos: Etiquetas RFID (*tags* ou *transponders*); Leitores RFID; *middleware* RFID e a aplicação RFID, que trata as informações, alinhando-as à proposta de negócio.

As etiquetas ou tags, tem a função de identificar cada produto ou objeto.

Os leitores, que são o segundo componente da estrutura, têm como finalidade efetuar a leitura das informações que se encontram armazenadas nas etiquetas (DA COSTA, 2008).

O leitor cria e envia sinais de rádio frequência através de uma antena, com isso recebe uma resposta da *tag* interrogada com as informações nela contidas, o leitor então decodifica esses dados e os organiza para que possam ser processados por algum computador (GLOVER, 2006).

O terceiro componente da arquitetura são as antenas, essas por sua vez são responsáveis por realizar a comunicação entre etiqueta e leitor. As antenas tem a finalidade de emitir as ondas eletromagnéticas que atingem a etiqueta, também são responsáveis por capitar a sua reposta. As antenas podem ser encontradas de vários tipos, podendo ser interna, integrada diretamente ao aparelho de leitura ou externa (TEIXEIRA, 2011).

O último componente do sistema é o *middleware*, cujo mesmo está integrado ao software compondo assim um sistema que trata as informações obtidas. “Os dados das etiquetas devem passar por um software capaz de filtrar, converter, corrigir e reencaminhar para sistemas ERP. Esta camada de software é o *middleware*. ” (DA COSTA, 2008, p. 18).

CONCLUSÕES

A tecnologia está cada vez mais presente nas atividades das organizações, desde a criação de um simples documento, até a construção de seu produto final, com isso, podemos observar que a disponibilidade das organizações em investir em aplicações que possam automatizar seus processos, vem crescendo cada vez mais. Diante dos fatos supracitados, o desenvolvimento de um software capaz de realizar a função de manipulador de dados que estão contidos em uma etiqueta RFID, seria extremamente conveniente para que, usado em conjunto com um coletor de dados, possa auxiliar as organizações no processo de verificação de seus ativos móveis. Tornando esse um processo mais eficiente e eficaz.

REFERÊNCIAS

BOMAN, Robert C.; HANSON, Brian. **Personal item monitor using radio frequency identification**. U.S. Patent n. 7,034,684, 25 abr. 2006.

CHIESA, M., et al. **RFID**: a weeklong survey on the technology and its potential. Harnessing Technology Project. 2002. Disponível em:

<http://people.interactionivrea.org/c.noessel/RFID/RFID_research.pdf >. Acesso em 25 ago. 2017.

DA COSTA, R. M. M. **Sistema Automático de Identificação e Colocação de Informação em Garrafas de Gás**. 2008. Dissertação (Mestrado em Engenharia Electrotécnica e de Computadores Major Automação) - Faculdade de Engenharia da Universidade do Porto, Porto. Disponível em:<http://repositorioaberto.up.pt/bitstream/10216/59731/1/000136644.pdf>. Acesso em 30 de mai. 2017.

GLOVER, B.; BHATT, H. **Fundamentos de RFID**. Rio de Janeiro: Alta Books, 2007. 240 p. ISBN 978-85-7608-139-5.

LAUDON, K.; LAUDON, J. **Sistemas de Informação Gerenciais**. 2. ed. São Paulo: Pearson Education do Brasil Ltda, 2013. 429p.

LUCAS, Henry C. **Tecnologia da informação**: tomada de decisão estratégica para administradores. Rio de Janeiro: LTC, 2006.

POZO, H. **Administração de recursos materiais e patrimoniais**: Uma abordagem logística. 7. ed. São Paulo: Atlas, 2015. 456p.

RIBEIRO, Carlos. Avaliação Patrimonial com RFID. **RFID Journal Brasil**, 10 nov. 2016. Disponível em <http://brasil.rfidjournal.com/artigos/vision?15209/> Acesso em 18 ago. 2017.

'TEIXEIRA, Tiago. **Controle de Fluxo de Pessoas Usando RFId**. 2011. 73 f. TCC (Graduação) - Curso de Tecnologia em Sistemas de Telecomunicações, Instituto Federal de Santa Catarina, São José, 2011.