Revisão sistemática da evolução MVC na base ACM

Valéria Martins da Silva

Ciência da Computação
Universidade Federal do Tocantins, Brasil
valeriasilva@uft.edu.br

Resumo

O MVC (*Model-View-Controller*) é um padrão ou arquitetura de desenvolvimento que particionou o processo de criação e manutenção de sistemas buscando a escalabilidade e eficiência da aplicação. No entanto, um projeto pode ter peculiaridades e necessidades que a arquitetura MVC não satisfaz, o que induz a geração de estudos que analisam a aplicabilidade de evoluções propostas neste padrão. Por esta razão, foi realizada uma revisão sistemática com o objetivo de reunir os estudos relevantes sobre a evolução do MVC na base da ACM (*Association for Computing Machinery*). Esta iniciativa compõe parte dos requisitos da disciplina de Metodologia Científica do curso de Ciência da Computação da Universidade Federal do Tocantins, Brasil. A procura e seleção dos estudos foi baseada em técnicas de busca e critérios de escolha. Com relação aos estudos encontrados, percebemos que a evolução com base na arquitetura MVC atende às diversas carências no desenvolvimento e manutenção de um software, e se torna necessária nos casos em que as camadas Modelo, Visão e Controle atingem alguns resultados, porém, a criação de novas camadas ou funcionalidades completaria a satisfação das exigências do sistema. Concluímos que a revisão sistemática gerou resultados positivos no aprendizado, visto que superou as expectativas na síntese de dados relevantes extraídos de publicações distintas.

Abstract

The MVC (Model-View-Controller) is a standard architecture or development that partitioned the process of creating and maintaining systems seeking scalability and efficiency of the application. However, a project may have quirks and needs that do not satisfy the MVC architecture, which induces the generation of studies that examine the applicability of proposed changes in this pattern. For this reason, we performed a systematic review in order to gather the relevant studies on the evolution of the MVC at the base of the ACM (Association for Computing Machinery). This initiative consists of the re-quisitos the discipline of scientific methodology of the course of Computer Science, Federal University of Tocantins, Brazil. The search and selection of studies was based on search techniques and selection criteria. Regarding the studies, we realize that evolution based on MVC architecture meets the diverse

needs in developing and maintaining software, and is necessary in cases where the layers Model, View and control reach some results, however, the creation new layers or features complement to meet the requirements of the system. We conclude that the systematic review has generated positive results in learning, since exceeded expectations in the synthesis of relevant data extracted from various publications.

Keywords: Systematic review, developments MVC, software engineering.

1 Introdução

A crescente busca pela qualidade e facilidade no desenvolvimento e manutenção de software desperta o interesse na criação de medidas e técnicas que auxiliam na criação de sistemas sustentáveis capazes de acomodar mudanças sem muita dificuldade e trabalho. O padrão MVC (*Model-View-Controller*) sugere uma arquitetura de software dividida em componentes, viabilizando com clareza o desenvolvimento de um código organizado e enxuto, e posteriormente, a reciclagem e manutenção do sistema sem dificuldade e com segurança. Porém, a independência dos componentes só será atingida se houver uma organização do sistema em camadas para garantir a escalabilidade, eficiência e a reusabilidade.

A separação de componentes tem por objetivo primário a separação da lógica e negócio de apresentação, ou seja, haverá a divisão entre a interface do usuário e a lógica do sistema. As camadas Modelo, Visão e Controle exercem esta divisão de funcionalidades ao desenvolver e executar um software. No padrão MVC, o Modelo trabalha na manipulação dos dados internos de uma aplicação, e se comunica especialmente com o armazenamento de dados. A camada de Visão ou apresentação trabalha na interface do usuário, capturando as suas ações e enviando ao Controlador, acessa os dados do Modelo através do Controlador e aplica a apresentação desses dados conforme o evento. Por fim, a camada de Controle exerce funcionalidades que envolvem o comportamento da aplicação; controla os fluxos entre as camadas de Visão e Modelo, e gera a resposta ao usuário.

Devido às vantagens na utilização da arquitetura, uma série de tecnologias e frameworks web têm sido implementados com o MVC. Este padrão se tornou popular no desenvolvimento de sistemas complexos, cada qual com suas exigências, metas e peculiaridades. Naturalmente, o MVC passou a ser objeto de estudo e pesquisa para a análise de sua aplicabilidade e o seu potencial evolutivo, especialmente do incremento de sua essência para melhor se encaixar aos quesitos de cada sistema específico. Portanto, buscou-se levantar o status da utilização do MVC e suas evoluções diversas, para isto foi realizada a seguinte revisão sistemática utilizando a base da ACM.

Organizamos este artigo de acordo com a estrutura IMRAD: introdução, métodos, resultados e discussão, o que é adotado como parte dos Requisitos Uniformes para Manuscritos Submetidos a Revistas Biomédicas do Comitê Internacional de Editores de Revistas Médicas, atualização 2008 . Acreditamos que a adoção desta estrutura ajudaria mecanismos de busca em bases de dados internacionais para armazenar e

recuperar informações em trabalhos de pesquisa a fim de facilitar as meta-análises e revisões sistemáticas.

2 Métodos

A revisão sistemática é um procedimento satisfatório para reunir uma coletividade de informações relevantes sobre determinado assunto - neste caso, o MVC e a evolução de software. A fim de executar essa revisão tomou-se como referência o processo executado no artigo de Breivold *et al.,* "A systematic review of software architecture evolution research" [1], por ser uma publicação recente e de uma estrutura focada e compreensível, porém esse trabalho não teve o padrão MVC como foco.

A revisão foi realizada para a disciplina de Metodologia Científica 2012/01 do curso de Ciência da Computação, com o objetivo de obter prática e experiência neste método. O tema escolhido se generalizou para todos, porém, houve a divisão das bases de dados científicas para a consulta e obtenção dos artigos primários. A revisão se estendeu por várias etapas de acordo com o artigo de referência: (i) estabelecer um protocolo para a revisão, (ii) definir os critérios de inclusão e exclusão, (iii) consultar estudos relevantes. (iv) avaliar a qualidade, (v) extrair e sintetizar as informações.

2.1 Estabelecer um protocolo para a revisão

O início deste estudo acontece com a criação de um protocolo de revisão, onde se discute e estabelece um fundamento para a revisão sistemática. O protocolo determina os termos de pesquisa nas bases de dados científicas, os critérios de inclusão e exclusão, os métodos para seleção dos estudos, extração e síntese dos dados. Esta etapa inicial foi discutida e planejada em sala de aula com a participação dos alunos e auxílio do professor da disciplina.

2.2 Definir os critérios de inclusão e exclusão

Os critérios de inclusão e exclusão separam os estudos relevantes daqueles que não preenchem corretamente os requisitos escolhidos. Excluímos os estudos que não são diretamente relacionados com a arquitetura MVC e sua evolução na criação de softwares, além de artigos duplicados, incluindo apenas a publicação mais completa. Para a inclusão de um artigo, ele deve satisfazer todos os critérios de inclusão e não pode satisfazer nenhum dos critérios de exclusão. Assim sendo, consideramos apenas publicações em inglês, que correspondam à exigência dos termos de busca e cujo conteúdo é relevante ao tema. Não foi estipulado qualquer critério referente à data de publicação, pois o nosso objetivo é obter todos os estudos primários na esfera da engenharia de software, especificamente sobre o padrão MVC de uma determinada base de dados, independentemente de quando foram divulgados.

2.3 Consultar estudos relevantes

Para facilitar o processo de consulta dos artigos científicos foram realizadas as pesquisas em bases de dados pela Internet designando para cada aluno uma base de dados diferente. As bases foram: IEEE Xplore; Compendex; Science Direct; Wiley InterScience; Springer Link; ISI Web of Science; ACM Digital Library; e Google Scholar. No nosso caso, a *ACM Digital Library* foi escolhida para a consulta e seleção dos estudos primários deste artigo.

As possíveis características advindas da evolução da arquitetura Modelo-Visão-Controle formaram os argumentos de consulta, que são: (MVC and evolvability) OR (MVC and maintainability) OR (MVC and extensibility) OR (MVC and adaptability) OR (MVC and flexibility) OR (MVC and changeability) OR (MVC and modifiability) OR (MVC and analyzability). Estes argumentos retornaram um total de 481 artigos.

Há uma série de fases cujo objetivo é apurar e selecionar os artigos relevantes. O processo foi feito nesta ordem: consulta em base de dados para obter artigos utilizando os termos de busca; excluir estudos irrisórios com base nos critérios de inclusão e exclusão; excluir estudos após análise de texto e resumo; definir estudos primários com a leitura de texto completo.

2.4 Avaliar a qualidade

Alguns critérios também foram estipulados a fim de deduzir o potencial dos estudos em relação ao seu conteúdo e credibilidade: análise precisa e embasada, desconsiderando explanações incompletas ou hipóteses ad hoc; o artigo deve ter fundamento no assunto da revisão sistemática; compatibilidade entre os objetivos dos estudos com a revisão e a pesquisa; o estudo faz uma declaração devida do método de pesquisa, posteriormente utilizado na extração dos dados.

Para assegurar a escolha dos artigos confiando na sua credibilidade e relevância para este projeto, é preciso que cada estudo satisfaça os requisitos de cada critério proposto.

2.5 Extrair e Sintetizar as informações

A síntese dos dados extraídos foi realizada com a leitura completa de cada um dos estudos, após serem apurados e selecionados. A extração dos dados relevantes foi administrada com o auxílio de uma planilha eletrônica Excel, com o objetivo de separar e organizar as informações de modo que forneçam uma visão geral dos artigos e facilite a manipulação dos dados para esta revisão.

3 Resultados

Todos os 481 artigos foram apurados seguindo as exigências de cada fase. A tabela 1 mostra com detalhes o número de artigos selecionados no decorrer dos passos,

organizados de acordo com o respectivo termo de busca. Inicialmente, houve uma queda considerável de estudos na fase de análise dos critérios de inclusão e exclusão. As causas giram em torno do alto descarte de artigos duplicados ou relacionados à biologia marinha, medicina e outros assuntos irrelevantes, descobertos com o acesso imediato às palavras-chave disponibilizadas pela ACM Digital Library. Posteriormente, outros artigos foram removidos após apuração de textos e resumos, restando apenas nove artigos primários definidos com base na leitura do texto completo.

Revisão sistemática da evolução MVC				
ACM Digital Library				
Termos de busca	Consulta pelos Termos de busca	Após Excluir com base nos critérios de inclusão e exclusao	Após Excluir com base no texto e resumo	Após Excluir com base no texto completo
MVC and EVOLVABILITY	10	8	0	0
MVC and MAINTAINABILITY	73	30	5	4
MVC and EXTENSIBILITY	98	26	4	2
MVC and ADAPTABILITY	41	7	1	1
MVC and FLEXIBILITY	243	16	5	2
MVC and CHANGEABILITY	5	0	0	0
MVC and MODIFIABILITY	11	2	0	0
MVC and ANALYZABILITY	0	0	0	0
Total	**481**	**89**	**15**	**9**

Tabela 1. Busca dos dados

Após a leitura completa, houve a extração de dados dos artigos selecionados, sintetizados abaixo, na tabela 2, de acordo com os itens: Identificação: identifica o artigo analisado através do seu título; Aplicação: Faz referência ao nome da arquitetura resultante da evolução MVC; Objetivo: Este campo descreve as idéias e necessidades que impulsionaram as inovações; Breve descrição: descreve de forma resumida a implementação e aplicabilidade da arquitetura; Resultados: destaque dos principais resultados obtidos, positivos ou negativos, em cada situação.

Analisando os artigos a partir de seus respectivos termos de busca, verifica-se que a expressão manutenibilidade está intimamente ligada à arquitetura MVC e suas inovações. A facilidade de manutenção acontece devido ao isolamento das camadas, facilitando a adição de novas funcionalidades.

O percentual de artigos selecionados após a leitura completa corresponde menos de 2% do total obtido na consulta inicial. Esse desfalque limitou a disposição de

informações para esta revisão, impossibilitando uma análise mais profunda do MVC e sua evolução na criação de softwares na base de dados da ACM.

Tabela 2. Extração e síntese dos dados

Identificação	Aplicação	Objetivo	Breve descrição	Resultados
A Functional Model-View-Controller Software Architecture for Command-oriented Programs	Functional MVC	Corrigir a dificuldade e tendência a erros no desenvolvimento de múltiplas interfaces de usuário.	No MVC funcional, um programa está estruturado com um modelo (domínio de aspectos específicos), visão (usuário abstrato) e controlador (loops de comando). Ao contrário de orientado a objetos MVC, um controlador é ativo, consistindo de um número de funções recursivas. Para aumentar a adaptabilidade, o controlador deve ser parametrizado pelo seu modelo e visão, utilizando uma função ou um functor ML-style.	Foi ilustrado e testado através da implementação de um programa completo que foi escolhido para ser complexo o suficiente para levantar as principais questões envolvidas no comando de programas orientados.
Re-Engineering the AlgorithmA Project for Long-Term Maintenance	MVC architecture	Re-engenharia do projeto AlgorithmA inteiro, que é uma situação que realmente acontece em uma empresa de software real. Um grande desafio para este projeto tem sido a dificuldade em manter o controle de qualidade durante muitos anos.	A reengenharia foi baseada em uma estrita aderência à arquitetura MVC utilizando padrões de projeto conhecidos. Isto tem a função de garantir que o novo sistema permaneça sustentável e escalável.	AlgorithmA 2007 foi re-projetado usando o modelo MVC. Foi mostrado que a manutenção era mais fácil devido à aderência ao design MVC e mantendo todos os componentes neste padrão de design.
Using XForms to Simplify Web Programming	HopiXForms (HX)	A dificuldade de desenvolver e implantar aplicações web comerciais aumenta à medida que o número de tecnologias utilizadas aumenta e, como as interações entre essas tecnologias se tornam mais complexas. Este artigo descreve uma maneira de evitar a complexidade crescente de voltar a analisar os requisitos básicos de aplicações web.	Esta abordagem diz respeito do cliente separado de preocupações do servidor, e, em seguida, a redução da interação entre cliente e servidor para sua forma mais elementar: passagem de parâmetro. Houve a descrição da implementação de um construtor de aplicação *MVC-based* para este modelo, que gera automaticamente o código necessário para empacotar entrada e saída de dados entre clientes e servidores.	Mostraram como implementar de forma eficiente o modelo de programação usando um construtor de aplicação *MVC-base* e automaticamente empacotamento de dados entre XML e Java. Fizeram a decomposição de aplicações web em suas partes constituintes, colocando a função de exibição no cliente, a função de controlador no servidor, e função de modelo no cliente e servidor. O construtor de aplicativo fornece editores visuais que refletem a estrutura MVC das aplicações web geradas.

Community portals for architecture-based middleware P2P	4-layer MVC model	As cidades digitais podem ter muitas definições. São as pessoas, sistemas e instituições que permanecem conectados através de uma infra-estrutura de comunicação digital. O objetivo é apresentar o desenvolvimento de um portal comunitário com base no modelo MVC 4 camadas, impulsionado pelo crescimento das cidades digitais.	Em busca de maior segurança e flexibilidade, a camada do cliente de comunicação foi separada da lógica de negócios, criando uma camada de aplicação responsável por despachar os pedidos e controlar seus fluxos com a arquitetura de 4 camadas (Model-View-Controller-Data).	A arquitetura de 4 camadas (Model-View-Controller-Data) permitiu alcançar alguns objetivos: Facilita o desenvolvimento de um sistema distribuído com segurança e com alta disponibilidade; aumenta a capacidade de automação da oferta de cidadãos e exerce serviços online;, acelera a sua execução e cumprimento maior de qualidade global.
Synchronous Online Help Support with Visual Instruction Aids for Workflow-based MVC Web Applications	W-MVC	A falta de ações coordenadas entre o usuário final e o agente de apoio durante uma sessão de ajuda, baseado na web, relacionado com a tarefa era visto como uma limitação importante. Para lidar com as limitações, foi concebido e implementado um sistema de suporte online de ajuda, não-intrusivo, interativo, baseado na web, que usa as anotações da Web para fornecer coordenadas e tarefas orientadas a instruções visuais. Nomeou-se o AnnSOS sistema de ajuda para anotação baseada no Suporte Synchronous Online.	Para manter e apoiar o crescimento de aplicações web que oferecem serviços on-line para uma ampla gama de usuários finais, os desenvolvedores web têm utilizado, entre outras abordagens de design, tradicionais padrões baseados em metodologias de desenvolvimento de software, como o Model View Controller (MVC) e paradigma do fluxo de trabalho (workflow-based Model View Controller (MVC)) combinados numa abordagem para desenvolver esses aplicativos web mais sofisticados.	Referimo-nos ao MVC baseado em frameworks web de desenvolvimento que incorporem o conceito de fluxos de trabalho como W-MVC. AnnSoS podem ser integrados, ainda que com pequenas modificações, em qualquer aplicação web em conformidade com um quadro W-MVC. Esta técnica alivia os esforços exigidos pelos usuários finais e atribui mais responsabilidades para os agentes de apoio. Além disso, a técnica de implementação é independente de plataforma que utiliza tecnologias web padrão para apoiar as questões de portabilidade e interoperabilidade. Durante o processo de desenvolvimento de AnnSoS, chegou-se a uma convicção de que sua realização é possível com as atuais tecnologias existentes e melhor ainda com emergentes tecnologias padrão.
Towards A Minimalist Multimodal Dialogue Framework Using Recursive MVC Pattern	Recursive application of Model-View-Controller (MVC)	Há uma tendência crescente para incorporar ricas interações multimodais em diversas interfaces de usuário, incluindo navegador, dispositivo móvel, etc. Estudos mostraram que a interação multimodal é a interface mais eficaz entre homem e máquina. No entanto, a eficácia vem com um custo, porque os sistemas multimodais de diálogo são muito mais complexos do que os unimodais. Este artigo apresenta um quadro formal de sistemas de diálogo multimodais através da aplicação de um conjunto de padrões de redução de complexidade.	O conceito de visão neste caso não se limita à interface gráfica do usuário. Em vez disso, significa uma visão composta que coordena as atividades entre um conjunto de visões primitivas de acordo com as restrições espaciais, temporais e semânticas. Sistemas multimodais de diálogo exibem recursivos padrões MVC em todas as camadas. Uma visão composta pode ser decomposta em componentes MVC, que cria essa relação abstrata recursiva: MVC = M(MVC)C. Esta recursão eventualmente termina quando as visões primitivas não podem ser decompostostas adicionalmente. MVC recursiva, portanto, impõe uma estrutura uniforme hierárquica para representar arbitrariamente complexos sistemas de diálogo multimodais usando o padrão Composite.	Foi apresentado um quadro minimalista para os sistemas multimodais de diálogo através da aplicação de um conjunto de padrões de redução de complexidade no MMI projeto da arquitetura do sistema, ou seja, a estratificação, recursão, MVC, e interpretação. Estas técnicas são combinadas para criar uma estrutura formal que é modular, concisa, flexível e dinâmica. Fez-se um protótipo de sistema baseado em XML multimodal do diálogo baseado na estrutura proposta. Estudos experimentais indicam que o método proposto é eficaz. Acredita-se que a abordagem estruturada ajudará a ganhar mais introspecções em interações multimodais e promover sistemas de diálogo multimodais de uma maneira extensível e consistente. Outros estudos estão em andamento para investigar problemas na composição dinâmica e sincronização para sistemas distribuídos de diálogo multimodais.

A Novel Web Application Frame Developed by MVC	Web application frame based on MVC	Os padrões de desenvolvimento de aplicativos da Web e ferramentas influenciam diretamente a vários fatores importantes, tais como manutenção, estabilidade, escalabilidade e segurança. Neste trabalho, o objetivo do MVC baseado no padrão Web de design do aplicativo é resolver os dois fatores - a manutenção e escalabilidade, que são os problemas mais difíceis no desenvolvimento de aplicações web.	Mesmo que o padrão de projeto MVC já foi apresentado em smalltalk-80 e amplamente utilizado em design de software, é difícil introduzi-lo no desenvolvimento de aplicações web. Os fatores que dificultavam a utilização do MVC foram sanados até certo ponto com o lançamento do modelo JSP 2. Dependendo das funções potentes da linguagem Java, o modelo JSP 2 implementa a idéia de projeto MVC até certo ponto: • Modelo (a camada de dados): ela é encapsulada na Empresa JavaBean. • Visao (a camada de apresentação): JSP é responsável pela apresentação de páginas, tais como o formato de apresentação,paging, etc • Controlador (a camada de controle): Servlet recebe a entrada do usuário em páginas e executa a operação em causa (chamando os módulos correspondentes EJB), então dá os resultados para a Apresentação relevante, de modo que JSP possa apresentar estes resultados. Alguns limites ainda existem, por exemplo, embora JSP Model 2 implementou a idéia básica do padrão de projeto MVC, a relação destas três camadas (Servlet, JSP e Java-Bean/EJB) ainda está próxi	A melhoria da estrutura de sistemas Web com base no padrão MVC tem estrutura mais clara do que as tradicionais. Os acoplamentos entre cada módulo são mais flexíveis. Especialmente, a presente Estrutura resolve um grande problema, que confundiu programadores Web por um longo tempo. Ele separa a lógica de operação da apresentação, e implementa o processo paralelo no desenvolvimento. Além disso, usando o banco de dados modelo e banco de dados LPM, a flexibilidade de manutenção e escalabilidade do sistema como um todo foram melhorados significativamente.
Domain Driven Web Development With WebJinn	DDD model	Propor um novo modelo como o próximo passo lógico no caminho evolutivo de desenvolvimento web, que resolva os problemas de embaraço e espalhamento inerentes em páginas dinâmicas, causados respectivamente pelo *Intra-crosscutting e Inter-crosscutting.*	A evolução dos modelos de desenvolvimento de aplicações web é impulsionado pelo desejo de modularizar características transversais e diminuir a dependência entre os web designers e programadores web. Mais recentemente, um modelo Model-View-Controller está sendo empregado no desenvolvimento web. Este trabalho relata um novo modelo, o DDD, como o próximo passo lógico no caminho evolutivo de desenvolvimento, que incorpora XP no modelo MVC. O modelo de DDD combina os modelos MVC e XP para fornecer uma solução unificada para ambas preocupações, *intra and inter-crosscutting*. O WebJinn / DDD é um modelo que implementa a estrutura DDD.	O desenvolvimento Web implementado em DDD gera a reutilização no código da aplicação e facilidade de adaptação. A estrutura DDD suporta dois mecanismos de reutilização: reaproveitamento através da composição e reutilização por meio da especialização. Com WebJinn / DDD, é possível criar componentes de aplicação web reutilizáveis e criar uma instância de um aplicativo executável a partir de modelos de componentes montados. Aplicações desenvolvidas com WebJinn / DDD aumentam significativamente a produtividade de desenvolvimento web e sua reutilização.
openMVC: A Non-proprietary Component-based Framework for Web Applications	openMVC	Propor um framework de desenvolvimento híbrido - uma evolução do MVC - que resolva problemas de abordagens não padronizadas no desenvolvimento web. Este artigo apresenta openMVC, um framework baseado em MVC que irá promover uma abordagem de desenvolvimento padronizado baseado em componentes para sistemas baseados na web e utilizar tecnologias não proprietárias interoperáveis para evitar aprisionamento tecnológico.	O openMVC é implementado usando uma arquitetura de cinco camadas. A primeira camada é a camada Client. A segunda, a camada de lógica de apresentação (PLL) é o servidor web. Nesta camada de dados, é dada uma estrutura de apresentação que o navegador será capaz de exibir. Detalhes sobre como os elementos de layout devem olhar são definidos nas informações de estilo. O PLL se comunica com o terceiro nível através de um mecanismo de chamada remota de procedimento (RPC). O terceiro nível, o Business Logic Layer (BLL), normalmente localizados em um servidor de aplicação implementa o domínio de processos de negócios e regras específicas, bem como definindo todas as restrições de validação de dados. A camada de abstração - quarta camada - ou de dados (DAL) abstrai o Sistema de Gerenciamento de Banco de Dados (SGBD) a partir do código que acessa-lo. A BLL usa essa camada para manipular o banco de dados. A camada final é o banco de dados onde os dados persistentes para a solução existem.	A estrutura openMVC baseado no padrão MVC permitiu que os componentes da Informação estilo, layout e restrições de validação sejam atualizados sem qualquer alteração de código ou necessidade de recompilar. A Redundância é reduzida, como restrições de validação, e informações de layout nunca são repetidamente definidas. Além de facilitar a manutenção mais rápida, o que promove uma gestão mais fácil de configuração e reduz os problemas de controle de versão em reconstruções de aplicação.

4 Discussão

A revisão sistemática se mostrou eficiente na análise de literatura, neste caso na área de engenharia de software. Este trabalho serviu como fonte de conhecimento e experiência na aplicação deste método e confirmou a existência de inovações baseadas na arquitetura MVC, que atendem às diversas carências no desenvolvimento e manutenção de um software.

Este estudo levantou uma coletividade de artigos com a finalidade de observar os impactos na utilização da arquitetura MVC, buscando a aplicação de possíveis mudanças neste padrão devido às peculiaridades e requisitos distintos de cada sistema.

De acordo com os dados sintetizados, pôde-se notar que as camadas de Modelo, Visão e Controle nem sempre satisfazem as necessidades dos desenvolvedores, tendo que gerar modificações ou mesmo adicionar camadas para um melhor desempenho e eficiência do software. Dessa forma, a realização desta revisão foi satisfatória diante das expectativas, e revelou a importância de reunir estudos com assunto em comum para que se faça a integração das evidências relevantes.

Uma sugestão para trabalhos futuros é a análise da literatura sobre o MVC e sua evolução, utilizando todas as bases de dados aqui mencionadas para uma análise mais ampla e aprofundada do assunto.

5 Agradecimentos

Este trabalho foi orientando pelo professor da disciplina de Metodologia Científica do Curso de Ciência da Computação da Universidade Federal do Tocantins, Patrick Letouze. Agradeço a atenção e acompanhamento durante todo o processo.

6 Referências

1. H. P. Breivold, I. Crnkovic, M. Larsson - A systematic review of software architecture evolution research 2011.
2. R. Cardone, D. Soroker, A. Tiwari - Using XForms to Simplify Web Programming 2005.
3. A. Stoughton - A Functional Model-View-Controller Software Architecture for Command-oriented Programs 2008.
4. W. James, P. Lucas, J. O'Connor, A. I Concepcion - Re-Engineering the AlgorithmA Project for Long-Term Maintenance 2008.
5. M. Karam, M. A. Ibrahim -Synchronous Online Help Support with Visual Instruction Aids for Workflow-based MVC Web Applications 2009.
6. Li Li, Wu Chou -Towards A Minimalist Multimodal Dialogue Framework Using Recursive MVC Pattern 2008.
7. A. M. Panhan, E. Ignatowicz, L. S. Mendes - Community portals for architecture-based middleware P2P. 2009.
8. R. Barrett, S. J. Delany - openMVC: A Non-proprietary Component-based Framework for Web Applications. 2004.
9. S. Kojarski, D. H. Lorenz - Domain Driven Web Development With WebJinn 2003

10. L. GuangChun, WangYanhua, Lu X. Hanhong - A Novel Web Application Frame Developed by MVC. 2003.