A ENGENHARIA DE REQUISITOS E O DESENVOLVIMENTO DIRIGIDO A MODELOS: UMA REVISÃO SISTEMÁTICA

THE REQUIREMENTS ENGENEERING AND THE MODEL DRIVEN DEVELOPMENT: A SYSTEMATIC REVISION

Izabella Cristine Oliveira Rezende (Universidade Federal de Sergipe, Sergipe, Brasil) – izaa09@gmail.com

Adicinéia Aparecida de Oliveira (Universidade Federal de Sergipe, Sergipe, Brasil) – adicineia@ufs.com.br

Resumo *– O Desenvolvimento de Software Dirigido a modelos visa minimizar problemas relacionados a produtividade, portabilidade, custo e tempo encontrado nas formas tradicionais de desenvolvimento através do uso de abstrações. Esse artigo tem como objetivo apresentar os resultados de uma revisão bibliográfica que busca analisar as metodologias de desenvolvimento de software orientada a modelos para a Engenharia de Requisitos. Seu objetivo é apresentar as técnicas encontradas e fazer uma avaliação sobre o estado da arte da Engenharia de Requisitos e do Desenvolvimento Dirigido a Modelos.*

Abstract – *The Model Software Driven Development aims to minimize problems related to productivity, portability, cost and time found in traditional forms of development through the use of abstractions. This article aims to present the results of a literature review that seeks to analyze the software development methodologies oriented models for requirements engineering. Your goal is to present the techniques found and make an assessment on the state of the art Requirements Engineering and Model Driven Development.*

Palavras-chave *– Desenvolvimento de Software Dirigido a Modelos, DDM, Engenharia de Requisitos, ER, Requisitos.*

1 INTRODUÇÃO

Os atuais sistemas de informação precisam satisfazer as expectativas de seus usuários, as quais se encontram cada vez mais ambiciosas. Por esse motivo, é primordial reduzir o tempo e o custo de desenvolvimento de *software*. Além disso, a criação de sistemas complexos envolve pessoas de diferentes disciplinas, cada uma se comunicando com sua própria maneira. Esse fator pode ocasionar problemas de comunicação entre os interessados, afetando diretamente nos resultados (AIGUIER *et al.*, 2012). Dessa maneira, torna-se essencial que os envolvidos estejam integrados de forma consistente.

Uma das propostas para minimizar esses problemas é a aplicação de diferentes níveis de modelagem e modelos de transformação, como prescrito pelas Abordagens Orientadas a Modelos (ZOHREVAND *et al*., 2011). Essas abordagens fornecem meios promissores para automatizar o processo de software e, por esse motivo, são mais flexíveis para lidar com os desafios.

Nesse contexto, Zohrevand *et al*. (2011) afirmam que o Desenvolvimento Dirigido a Modelos (DDM) permite aos desenvolvedores foco na geração do *software* sem o envolvimento de conceitos específicos de plataforma. Isso pode acarretar a redução dos custos de desenvolvimento, melhoria da consistência de *software*, manutenção e qualidade, conforme colocam Chitforoush *et al.* (2007). Lazarte *et al.* (2010) esperam que DDM seja uma maneira eficiente de lidar e resolver problemas de interoperabilidade, heterogeneidade e alinhamento entre o negócio e soluções tecnológicas.

Antes da codificação do *software*, as melhores práticas da Engenharia de *Software* indicam a necessidade da definição o planejamento do que será construído. A Engenharia de Requisitos (ER) abrange tarefas que levam à concepção de qual será o impacto do *software* sobre o negócio, das reais necessidades do cliente e de como serão as interações dos usuários o *software* (PRESSMAN, 2010) Em ER, os projetos de *software* necessitam ter seus requisitos devidamente definidos, uma vez que eles apresentam uma visão central do projeto (FALBO, 2012). A a utilização de modelos leva a problemas relacionados à fase de levantamento de requisitos da ER. Dessa forma, surgem preocupações como:

- A necessidade de um processo bem definido para identificar e especificar o escopo de requisitos;
- Os mecanismos adequados para apoiar a comunicação entre os diferentes atores e equipes de desenvolvimento envolvidos no processo de ER;
- Os mecanismos para lidar com a volatilidade dos requisitos inerentes; e,
- A necessidade de um regime de rastreabilidade para ajudar a controlar os requisitos nas fases do processo de desenvolvimento.

É fundamental entender e alinhas as técnicas da ER com o MDD, uma vez que a necessidade de alterar requisitos persiste ao longo do desenvolvimento e entrega do sistema e um dos problemas em se manter requisitos em MDD está relacionado ao controle e rastreabilidade durante esse processo.

Para propor soluções para esses problemas, é necessário conhecer o estado da arte da ER e do DDM. Com esse objetivo, foi realizada uma revisão sistemática que abrange o período de 2004 a 2014. Esse artigo apresenta os resultados obtidos e uma breve análise.

O trabalho está organizado da seguinte maneira: A seção 2 descreve o protocolo de revisão sistemática, a seção 3 faz uma breve descrição sobre DDM, a seção 4 descreve a Engenharia de Requisitos, o item 5 representa o resultado da revisão sistemática e a seção 6 faz um resumo dos trabalhos selecionados. Por fim, a seção 7 apresenta a análise dos resultados e a seção 8 as considerações finais.

2 PROTOCOLO DE REVISÃO SISTEMÁTICA

Esta seção apresenta o protocolo utilizado para a aplicação da revisão sistemática deste artigo, que tem como objetivo localizar estudos de Engenharia de Requisitos no Desenvolvimento Dirigido a Modelos.
As perguntas utilizadas para realizar as buscas deste artigo foram:
Pergunta 1: Existem propostas de técnicas para a Engenharia de Requisitos alinhada ao DDM?
Pergunta 2: Quais são as propostas existentes?
Os critérios de seleção de fontes são:

- Consultas de artigos através da *web*;
- Utilização de mecanismos de busca através de palavras-chave; e,
- Artigos publicados entre 2004 e 2014.

Os critérios de inclusão são:

- Devem apresentar textos completos dos estudos em formato eletrônico;
- Os arquivos devem estar descritos em inglês;
- Os artigos devem apresentar estudos sobre a Engenharia de Requisitos alinhada ao Desenvolvimento Dirigido a Modelos; e,
- O título deve possuir referência a Engenharia de Requisitos e ao Desenvolvimento Dirigido a Modelos.

O critério de exclusão escolhido para esse trabalho é:

- Não devem existir artigos que não levantem os problemas da Engenharia de Requisitos para o DDM.

As consultas ocorreram na base da IEEExplorer, ACM e nos Periódicos CAPES. A *String* de busca utilizada foi*:*

- *((("Model-Driven Development") or ("Model Driven Development") or ("MDD")) and ("Requirements Engineering")).*

3 DESENVOLVIMENTO DE *SOFTWARE* DIRIGIDO A MODELOS

Novos conceitos e abordagens vêm sendo propostos para facilitar o desenvolvimento de *software*, como a aplicação de diferentes níveis de modelagem e modelos de transformação referentes às Abordagens Dirigidas a Modelos (ZOHREVAND *et al*., 2011). Essa seção apresenta alguns desses conceitos. Essas abordagens podem oferecer benefícios como: custo reduzido de desenvolvimento, melhor qualidade do produto, maior retorno em relação aos investimentos em tecnologia, entre outros (ALMEIDA, 2014).
A disciplina da Engenharia de *Software* em que os modelos são os artifícios fundamentais para o desenvolvimento é chamada de Engenharia Dirigida a Modelos, em inglês, *Model Driven Engineering* (MDE). A MDE é utilizada para descrever as abordagens de desenvolvimento em que modelos abstratos de sistemas de *software* são transformados em implementação (DEMIR *et al.,* 2006). A Figura 1 apresenta uma representação das iniciativas dirigidas a modelos.

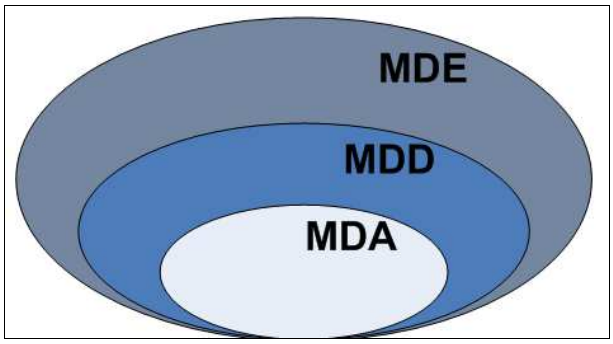

Figura 1 – Abordagens Dirigidas a Modelos.
Fonte: (DEMIR, 2006).

Observa-se na Figura 1 que a MDE pode ser considerada um superconjunto do DDM, uma vez que ela engloba outras atividades dirigidas por modelos do processo da Engenharia de *Software*. Enquanto que a Arquitetura Dirigida a Modelos, em inglês, *Model Driven Arquitecture* (MDA), que depende da utilização dos padrões da *Object Management Group* (OMG), pode ser considerada um subconjunto do DDM, sendo que essa última coopera com a flexibilidade para definição dos processos de desenvolvimento de *software* (ALMEIDA, 2014).

3.1 Modelos

Para Swithinbank *et al.* (2005), um modelo é a representação de um sistema a partir de uma perspectiva própria, com a omissão de detalhes com pouca relevância, de modo que as características sejam percebidas com clareza, podendo ser considerados como o esboço de um projeto. Para Almeida (2014) o modelo é um conceito simples do mundo real, que foi estabelecido para proporcionar um melhor entendimento de um *software* a ser desenvolvido.
Os modelos de *software* auxiliam as atividades de desenvolvimento e são construídos através da *Unified Modeling Language* (UML), que oferece uma notação visual e semântica da representação do *software* (SWITHINBANK *et al.*,2005).
No âmbito de Abordagens Dirigidas a Modelos, os modelos não são utilizados somente como esboço de um projeto, mas como componente principal a partir do qual as implementações são geradas (SWITHINBANK *et al.*, 2005). Dessa forma, Chitforoush *et al.* (2007) afirmam que o modelo é um meio importante para a especificação de soluções de grande escala, o qual deve ser expresso por meio notações de bem definidas, como a UML. Porém, Almeida (2014) cita que isso só irá ocorrer se o benefício obtido com a produção de modelos for maior e o esforço para o para mantê-los de acordo com o código for menor que a prática já existente.

3.2 Desenvolvimento de *Software* Dirigido a Modelos

O DDM baseia-se na ideia de geração de código através de modelos criados por meio das linguagens de modelagem como UML. Esse fator permite ao desenvolvedor foco na geração do *software* sem o envolvimento de conceitos específicos de plataforma (ZOHREVAND *et al*., 2011).

MacDonald *et al.* (2005) definem DDM como uma forma de separar a lógica do negócio de sua implementação. De acordo com Zohrevand *et al.* (2011), isso permite que o programador concentre-se em resolver o problema, ao invés de preocupar-se com os detalhes da tecnologia.
Na visão de Lazarte *et al.* (2010), o DDM utiliza os modelos de construção de software para simular, estimar, compreender, comunicar e produzir código. Já Franch *et al.* (2013) afirmam que os modelos são usados para especificar, simular, verificar, testar e gerar o sistema a ser construído. Segundo Azoff (2008), isso ocorre porque o usuário gera modelos através de uma plataforma independente e seleciona a plataforma de destino, a qual gera o código-fonte do programa. Nesse contexto, a MDA é um padrão arquitetural proposto pela OMG que tem sido utilizado como base para muitos outros modelos. A MDA incentiva o uso de modelos e suas transformações em diferentes níveis (FRANCH *et al.*, 2013), que são representados na Figura 2.

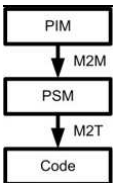

Figura 2 – Modelos e Transformações.
Fonte: (FRANCH *et al.*, 2013).

Os modelos e transformações apresentados na Figura 2 podem ser descritos da seguinte maneira:

- Modelos:
  - Modelos Independentes de Plataforma (PIM): Especifica o sistema de *software* de uma maneira independente da plataforma da tecnologia escolhida para implementá-lo.
  - Modelos Específicos de Plataforma (PSM): Refina a PIM com a especificação da plataforma de desenvolvimento do *software*. Sendo assim, duas implementações diferentes do mesmo sistema compartilham o mesmo PIM, mas têm dois PSMs diferentes, cada uma adaptada para as capacidades tecnológicas de cada plataforma.
- Transformações:
  - Modelo para Modelo (M2M): momento em que transformações evoluem um PIM em um PSM.
  - Modelo para Texto (M2T): Utilizados para gerar o sistema de executável a partir do PSM. Essa etapa inclui a geração de vários artefatos de código, como por exemplo, classes de negócio Java, esquemas banco de dados Oracle, etc.

Além dos Modelos citados, existe a Computação Independente de Modelos (CIM), que representa a fase que antecede o PIM. O CIM apresenta apenas os requisitos de sistema e não mostra detalhes de sua estrutura, este pode ser apresentado por meio de diagramas em UML (BRAGA, 2011).
Segundo Chitforoush *et al.*(2007), o uso do DDM acarreta na redução dos custos de desenvolvimento, melhoria da consistência de *software*, manutenção e qualidade. Devido a esses fatores, Lazarte *et al.* (2010) coloca o DDM como uma maneira eficiente de lidar e resolver problemas de interoperabilidade, heterogeneidade e alinhamento entre o negócio e soluções tecnológicas.

Logo, como DDM possui um potencial em automatização e geração de código, pode ocorrer à eliminação de tarefas repetitivas. Com isso, os custos de desenvolvimento são reduzidos, há melhoria na consistência do *software*, manutenção e qualidade, além do aumento significativo da produtividade (CHITFOROUSH *et al.*, 2007). Nesse contexto, os requisitos tornam-se fundamentais no processo de DDM. No entanto, uma das preocupações do DDM é identificar como seguir dos requisitos de usuários até a solução, projeto ou implementação.

4 ENGENHARIA DE REQUISITOS

Os requisitos apresentam uma visão central do projeto de *software* (FALBO, 2012). As fases de levantamento, análise, documentação, controle e gestão de qualidade dos requisitos compõem a Engenharia de Requisitos, etapa que serve como base para todas as outras fases do processo de desenvolvimento. Dessa forma, a ER torna-se decisiva no sucesso do projeto de *software*. A presente seção trata da ER abordando os conceitos de requisitos e seus processos e fases.

4.1 Engenharia de Requisitos

A Engenharia de Requisitos é uma das principais áreas em Engenharia de *Software*, uma vez que relaciona elementos de engenharia social e comportamental com o objetivo de fornecer uma interface entre os *stakeholders*, os projetistas e desenvolvedores (KILICAY e LAPLANTE, 2008). A ER engloba diferentes atividades no projeto de desenvolvimento de *software*. É por meio dela que é possível fazer uma interface entre o desejo do cliente e a implementação do sistema. Dessa forma, segundo Silva *et al.* (2004), a ER realiza o mapeamento informal ou levantamento de requisitos e o mapeamento formal ou geração de modelos e código. Segundo os autores, os requisitos mapeados na primeira etapa serão utilizados durante todo o processo de desenvolvimento de *software* com o intuito de garantir que o *software* construído seja o desejado.

Sommerville (2004) afirma que a ER auxilia no alcance de requisitos claros e consistentes. Para o autor, a ER é a maneira de descrever as tarefas envolvidas no ciclo de vida do *software* referente à acepção dos requisitos do sistema. Na visão de Zave (1997), a ER é uma área multidisciplinar que trata da identificação dos objetivos do sistema e com a forma em que serão atingidos, além de preocupar-se com processos de obtenção, aprimoramento e verificação das necessidades do cliente para que haja uma especificação completa e correta dos requisitos. A fase referente a ER antecede o processo de desenvolvimento de *software*.

4.2 Requisitos

A norma 830-1993 do *Institute of Electrical and Electronics Engineers* (IEEE) define requisitos como uma condição necessária para resolver um problema ou alcançar um objetivo. Segundo Leite (1999), “requisitos são objetivos ou restrições estabelecidas por clientes e usuários de sistemas que definem as propriedades do sistema”. Na visão de Mello e Meyer (2010), os requisitos são como uma coleção de sentenças que descrevem de maneira coesa todos os aspectos do sistema proposto.

Os requisitos devem ser determinados por clientes, usuários e envolvidos no produto de *software* que será construído (WESTFALL, 2006). O Quadro 1 apresenta alguns termos mais comumente encontrados no domínio dos requisitos.

Quadro 1 – Conceitos na área de ER.

Termo	Definição
Requisitos de Negócio	O objetivo dos requisitos de negócio de alto nível da organização é construir um produto ou adquirir de um cliente.
Regras de Negócio	A política, diretriz, norma ou regulamento que define ou restringe algum aspecto do negócio. Não é um requisito de *software* em si, mas a origem de vários tipos de requisitos de *software*.
Restrições	A restrição que se impõe sobre as opções disponíveis para o desenvolvedor para a concepção e construção de um produto.
Interface externa de requisitos	Descrição de uma conexão entre um sistema de *software* e um utilizador, outro sistema disponível ou um dispositivo de *hardware*.
Característica	Um ou mais recursos do sistema relacionados logicamente que agregam valor a um usuário e são descritos por um conjunto de requisitos funcionais.
Requisitos Funcionais	Uma descrição de um comportamento que o sistema irá apresentar sob condições específicas.
Requisitos não funcionais	Descrição de uma propriedade ou característica que o sistema deve possuir ou restrições que o sistema deve respeitar.
Atributos de qualidade	Tipo de requisito não funcional que descreve um serviço ou características de performance de um produto.
Requisitos de sistema	Requisitos de nível superior para um produto que contém vários subsistemas, que pode ser tudo do *software* ou *software* e *hardware*.
Requisitos de usuário	A meta ou tarefa que classes específicas de usuários devem ser capazes de realizar com um sistema, ou um atributo do produto desejado.

Fonte: (WIEGERES e BEATTY, 2013).

Wiegeres e Beatty (2013) fazem a distinção dos requisitos de *software* em três níveis: Requisitos de negócio, requisitos de usuário e requisitos funcionais. Além disso, todo sistema necessita de requisitos não funcionais. A Figura 3 representa os níveis e tipos de requisitos definidos pelo autor como uma forma de organizar o conhecimento a respeito dos requisitos.

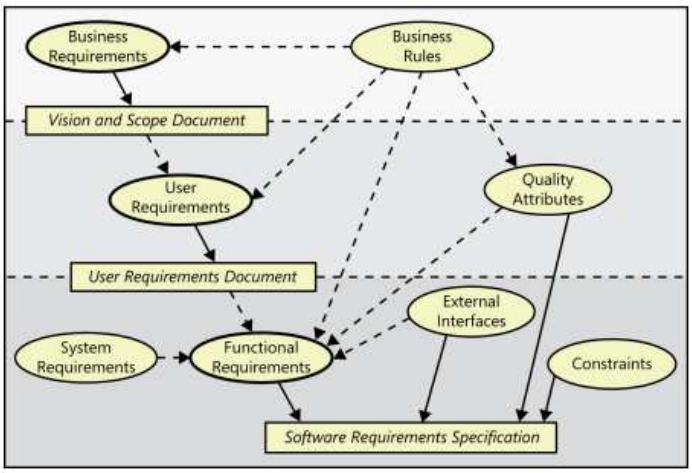

Figura 3 – Relação entre os vários tipos de necessidades de informação.
Fonte: (WIEGERES e BEATTY, 2013).

As representações ovais da Figura 3 apresentam os tipos de requisitos de informação e os retângulos os documentos necessários para armazenar essas informações. As setas preenchidas indicam que um tipo de informação está sendo armazenado em um documento enquanto que as tracejadas indicam que um tipo de informação foi originado de algum tipo

de exigência. Os requisitos de dados podem aparecer em qualquer nível de requisito. Os de negócio descrevem porque a organização está implementando o sistema. Os de usuário descrevem objetivos ou tarefas que um usuário deve executar no *software* com o objetivo de oferecer valor a algo ou alguém. Requisitos funcionais apresentam o comportamento do produto em situações específicas, ou seja, descrevem o que o sistema deve realizar para atender aos usuários (requisitos de usuário) satisfazendo assim os requisitos de negócio. Os requisitos de sistema descrevem os requisitos de um produto que é composto de outros subsistemas ou componentes.
Além dos requisitos, é importante conhecer os processos da ER. Segundo Falbo (2012), a ER deve ser descrita como um processo. Esses processos podem ser considerados um conjunto de atividades com o intuito de derivar, validar e manter um documento de requisitos (FRANCETO, 2005). Eles compreendem a sistematização dos procedimentos de definição dos requisitos. A próxima seção apresenta alguns processos da ER.

4.3 Processos da Engenharia de Requisitos

Existem diferentes propostas de modelos de processos em ER, os quais podem descrever de maneira simplificada o processo através de uma determinada visão. Cada atividade de um processo de ER pode ou não resultar em um produto no qual precisará ser mantido e estar apto a avaliação de qualidade. Sommerville (2007) propõe a divisão do processo de ER em 4 fases ou subprocessos: estudo da viabilidade, análise e elicitação dos requisitos, especificação dos requisitos e validação dos requisitos. A Figura 4 apresenta as entradas e saídas do processo da ER como proposto por Sommerville (2007).

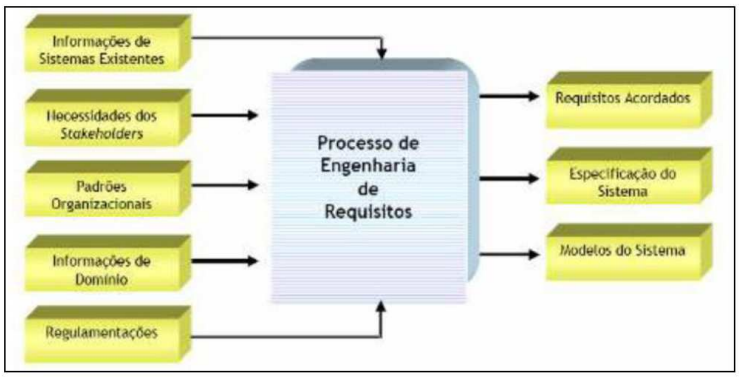

Figura 4 – Entradas e Saídas do Processo de ER.
Fonte: (SOMMERVILLE, 1997).

Na Figura 4 nota-se que os processos devem receber como entrada: informações sobre sistemas já existentes, necessidades dos *stakeholders*, padrões da organização, regulamentações e informações sobre o domínio da aplicação. Como saída, obtém-se os requisitos acordados, a especificação do sistema e os modelos do sistema. A Figura 5 apresenta o modelo espiral que foi proposto por Kotonya *et al.* (1997).

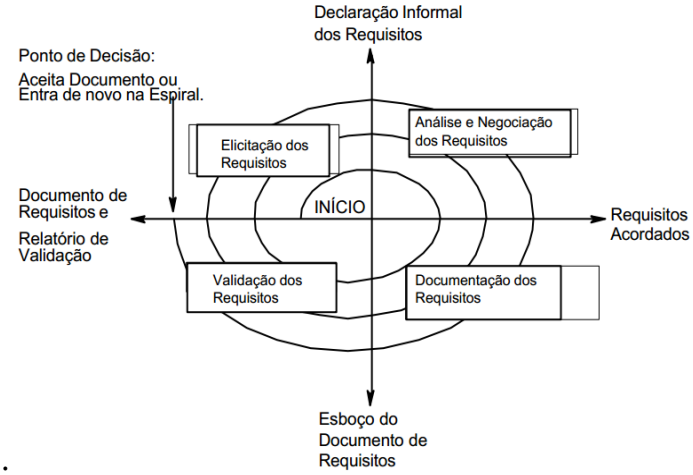

Figura 5 – Modelo Espiral de ER.
Fonte: (KOTONYA *et al.*, 1997).

Observa-se por meio da Figura 5 que o modelo sugerido é composto por quatro etapas. Cada uma dessas etapas é repetida até que o documento de requisitos atenda todos os objetivos propostos do sistema. Esse modelo permite iterações de acordo com o esforço e o tipo de desenvolvimento do *software*. Segundo Franceto (2005), essas etapas compreendem:

- Elicitação dos Requisitos: Primeira atividade da ER refere-se ao levantamento do conhecimento relacionado ao problema a ser resolvido. É nesse momento que o cliente e o desenvolvedor analisam as necessidades do usuário e as restrições ou validações que o sistema deverá ter. Além disso, há uma análise da organização, do domínio da aplicação e dos processos organizacionais. O principal objetivo dessa fase é o correto entendimento do que é esperado na solução a ser desenvolvida.
- Análise e Negociação dos Requisitos: Após coletar todos os requisitos, é necessário realizar a análise e entrar em acordo com os *stakeholders*. O objetivo dessa fase é gerar descrições dos requisitos de maneira em que eles possam ser facilmente compreendidos. Esses requisitos devem ser discutidos, negociados e priorizados juntamente com os interessados do sistema. O objeto de retorno dessa fase são os requisitos completos e organizados em ordem de prioridade.
- Documentação/Especificação dos Requisitos: Criação de um documento que apresente todos os requisitos do sistema de *software* de maneira coerente. Todos os envolvidos devem entender esse documento, uma vez que deverá servir como contrato entre o cliente e o desenvolvedor.
- Validação dos Requisitos: Processo que verifica se os requisitos definidos nas fases anteriores atendem às necessidades dos *stakeholders*.
- Gerência de Requisitos: Engloba todas as fases anteriores da ER, equivale ao planejamento do controle da elicitação, análise e negociação, documentação e validação dos requisitos.

Diante desse contexto torna-se necessário entender como os processos da ER influenciam no DDM, uma vez que existem problemas de integração entre essas duas áreas. Por esse motivo, na seção 5 é apresentado o resultado de uma revisão sistemática que teve como objetivo o aprofundamento e familiaridade com o problema, que engloba propostas e técnicas para lidar com a ER em DDM.

5 RESULTADO DA REVISÃO SISTEMÁTICA

Os resultados obtidos pela aplicação dos métodos de seleção dos trabalhos são mostrados de forma quantitativa na Tabela 1, sendo que a primeira coluna apresenta os métodos utilizados do protocolo, enquanto que as demais apresentam as fontes de pesquisa utilizadas para a investigação e a quantidade de artigos encontrada para cada método e fonte.

Tabela 1 – Resultado quantitativo dos artigos encontrados.

Métodos do Protocolo	**IEEE**	**CAPES**	**ACM**
Consulta das *strings* de busca	25	17	53
Período de 2004 a 2014	25	16	53
Critérios de inclusão	8	3	8
Leituras dos resumos	4	3	3
Critérios de Exclusão	2	1	1

Os trabalhos selecionados após a aplicação dos critérios de inclusão e exclusão são: (VALDERAS e PELECHANDO, 2009), (FRANCH *et al.*, 2010), (LONIEWSKI *et al.*, 2011) e (WINKLER e PILGRIM, 2010). O Gráfico 1 apresenta a quantidade de trabalhos encontrados no período pesquisado por fonte de pesquisa.

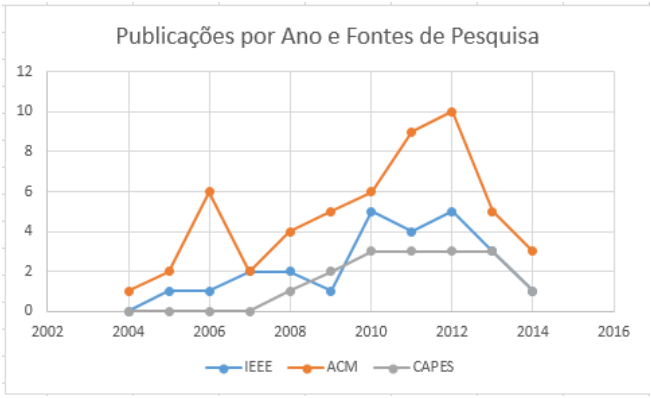

Gráfico 1 – Resultado Quantitativo de Publicações por Ano e Fontes de Pesquisa.

É possível notar que esse é um tema recente e que vem ganhando destaque a partir de 2010, quando houve uma maior quantidade de publicações sobre o assunto. A seção 6 apresenta o resumo dos trabalhos selecionados.

6 RESUMO DOS TRABALHOS SELECIONADOS

Valderas e Pelechando (2009) apresentam uma abordagem com recursos que suportam a rastreabilidade de requisitos em DDM no contexto de aplicações *Web*. A abordagem proposta fornece um retorno sobre como as transformações de modelos são realizadas, possibilitando a identificação de erros. Foi apresentado um guia metodológico que permite derivar modelos de navegação de requisitos baseados em modelos de tarefas, o qual tem sido definido como uma coleção de mapeamentos que tomam as abstrações de requisitos baseados em modelos de tarefas como base e indicam a forma que eles podem ser apoiados no nível conceitual.

Além disso, foi desenvolvida uma ferramenta de rastreabilidade para aplicar os mapeamentos de forma automática. Essa ferramenta gera relatórios após as transformações

que permitem que analistas estudem como os requisitos são suportados pelo modelo de navegação. Essa abordagem tem sido utilizada em aplicativos *Web* de médio e pequeno porte, através dele é possível colocar que os mapeamentos são corretamente aplicados para determinar elementos conceituais.

Winkler e Pilgrim (2009) buscam, por meio de um referencial teórico, encontrar as diferenças entre a rastreabilidade dos requisitos na ER e em DDM. Diante desse contexto foi descrita de forma teórica uma perspectiva da rastreabilidade com a definição desse termo, regimes e metamodelos utilizados, além das diferentes atividades realizadas nesse âmbito. Foi observado que o DDM concentra-se mais na rastreabilidade dos elementos do modelo por meio de passos de transformação consecutivos. Segundo os autores, em DDM os links de rastreabilidade podem ser produzidos de forma fácil, bastando reforçar transformações ou usando ferramentas de apoio rastreabilidade *built-in*. No entanto, notou-se que o foco em relação a rastreabilidade de requisitos é pouco em DDM, isso leva a problemas. Se por um lado, links de rastreabilidade fazem a ponte entre os não-modelos, e nesse caso, os modelos têm de ser registrados e expressos de alguma forma. Por outro lado, vários aspectos de rastreabilidade dependem de fatores externos e isso encontra-se fora do escopo DDM. Esse problema leva a uma motivação limitada para melhoria das abordagens mais globais para rastreabilidade e pesquisa em DDM.

Notou-se também que em DDM e em ER existem formas de rastreabilidade que podem ser utilizadas para chegar a um objetivo, porém elas são tratadas de forma isoladas. Falta uma estrutura para determinar o tipo e a granularidade dos links de rastreabilidade necessários para atingir um conjunto de determinadas metas. Além disso, falta uma boa ferramenta de suporte e isso faz com que os métodos de rastreabilidade não sejam utilizados da forma correta.

Loniewski *et al.* (2011) possuem uma proposta de abordagem metodológica ágil para o processo DDM que inclui as atividades da ER e que integra a arquitetura como o principal artefato da construção processo de DDM. Para os autores, a inclusão das atividades de ER para o DDM seria um meio para criar uma ponte entre o cliente e os profissionais de TI, como também um vínculo de rastreabilidade dos requisitos para a análise, concepção e implementação. O método proposto é orientado a arquitetura e uma metodologia dirigida a modelos que aplica abordagens interativas e incrementais em um ciclo de vida estruturado. Os autores apresentaram a estrutura padrão o ciclo MDA, porém incluíram foco particular na geração do PIM para os requisitos. Isso ocorre com uma estrutura bem definida orientada a arquitetura, na qual a transição dos requisitos para o CIM e depois do CIM para o PIM é feita no contexto de uma arquitetura específica.

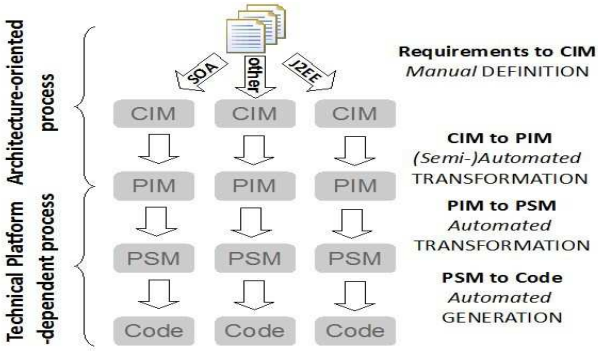

Figura 6 – Metodologia Proposta.

Fonte: (LONIEWSKI *et al.*, 2011).

O esquema desse ciclo é mostrado na Figura 6, em que os requisitos de alto nível são capturados, é identificado o candidato a arquitetura e a definição dos principais modelos, então é enviado um manual de definição ao CIM, e a partir desse momento ocorrem as transformações de CIM para PIM, PIM para PSM e PSM para código.
Através de um estudo de caso os autores puderam concluir que o método proposto melhora o DDM pois envolve a ER como parte do processo, isso significa que os modelos são usados como os artefatos primários na criação de outros modelos em diferentes níveis de abstração. Logo, pelo fato dos modelos de análise no nível PIM não serem criados manualmente, mas agora em um processo orientado a arquitetura, são gerados diversos benefícios aos programadores, como a melhoria dos requisitos, rastreabilidade, maior consistência entre os modelos, etc.
O artigo de Franch *et al.* (2010) aponta que o DDM preocupa-se somente com os Requisitos Funcionais (RF) do projeto, havendo uma falta de integração dos Requisitos Não-Funcionais (NFR) em seus processos atuais. Esse fator faz com que o sistema não satisfaça completamente as expectativas dos *steakholders,* já que a falta de preocupação com os NFR resulta na limitação do sucesso do DDM, da sua aplicabilidade e adoção pela indústria.
Dessa maneira, os autores preocupam-se com o impacto da definição dos NFRs causados com o DDM. No trabalho, são identificados os desafios a superar a fim de integrar definitivamente os NFRs ao DDM. Além disso, é traçado um quadro geral que integra os NFRs para o centro do processo de DDM e fornece uma
Na proposta dos autores, os requisitos funcionais e os não funcionais tornar-se-iam de primeira ordem, sendo assim, o *framework* proposto envolveria as seguintes fases:

- O analista especifica o PIM com os requisitos funcionais e não funcionais;
- O engenheiro verifica as informações sobre não-funcionalidade, arquiteturas e tecnologias e realiza uma transformação que leva o PIM como entrada e produz um PIM/PSM;
- É aplicada uma segunda transformação M2M, que gera o PSM com as diretrizes arquitetônicas forçando a adoção de uma tecnologia específica ou produto; e,
- Por fim, é aplicada a transformação final para o código, resultando em uma aplicação especifica atendendo a todos os requisitos funcionais e não funcionais

Devido à pequena quantidade de publicações, conforme mencionado na seção 5, apenas quatro artigos foram selecionados para essa revisão sistemática, os quais foram apresentados nessa seção. A próxima seção apresenta uma análise de resultados desses dos artigos.

7 ANÁLISE DOS RESULTADOS

Observa-se por meio desta revisão sistemática que as publicações em DDM são recentes, principalmente a partir de 2010. É notável a preocupação na área de Engenharia de Requisitos, visto que a modelagem correta dos requisitos pode garantir o sucesso do projeto. Os artigos abordados abrangem diferentes propostas sobre como lidar com a Engenharia de Requisitos em DDM.
O quadro dois demonstra os assuntos abordados nos artigos quando analisado sobre a ótica de alguns conceitos da ER.

Quadro 2 – Análise dos Resultados – Conceitos da ER.

Conceitos da ER / Autores	Valderas e Pelechando (2009)	Winkler e Pilgrim (2009)	Loniewski *et al.* (2011)	Franch e Carbot (2010)
Requisitos de Negócio				
Requisitos Funcionais				X
Requisitos Não-Funcionais				X
Atributos de Qualidade				
Requisitos de Sistema	X			
Rastreabilidade dos Requisitos	X	X	X	

O Quadro 3 apresenta os resultados quando os artigos foram analisados do ponto de vista das fases da ER.

Quadro 3 – Análise dos Resultados – Fases da ER.

Processos da ER / Autores	Valderas e Pelechando (2009)	Winkler e Pilgrim (2009)	Loniewski *et al.* (2011)	Franch e Carbot (2010)
Elicitação de Requisitos			X	X
Análise e Negociação de Requisitos			X	
Documentação de Requisitos			X	
Validação de Requisitos				

O Quadro 4 apresenta os artefatos que devem ser produzidos que foram mencionados nos artigos analisados.

Quadro 4 – Análise dos Resultados – Artefatos da ER.

Artefatos / Autores	Valderas e Pelechando (2009)	Winkler e Pilgrim (2009)	Loniewski *et al.* (2011)	Franch e Carbot (2010)
Declaração Informal dos Requisitos				
Requisitos Acordados				
Esboço do Documento de Requisitos				
Documento de Requisitos			X	
Relatório de Validação	X			

Através do Quadro 2 observa-se que não são mencionados nos artigos os requisitos de negócio e os atributos de qualidade. Entretanto, pelo menos 3 dos 4 autores preocuparam-se com a rastreabilidade dos requisitos, esse fator é positivo, uma vez que é necessário manter o acompanhamento dos requisitos do início ao fim do projeto do *software*, além de ligação entre os requisitos dependentes, visto que a alteração de um requisito pode afetar em outro requisito existente.

Diante do ponto de vista das fases da ER, nota-se por meio do Quadro 3 que os autores fazem menção a somente uma ou duas fases da ER, não envolvendo o DDM a ER como um todo. Além disso, em nenhum dos artigos é tratada a fase validação de requisitos. Esse fator é preocupante, sendo que é na fase de validação que é analisado se os requisitos

atendem às necessidades dos *steakholders*. Por meio do Quadro 4, nota-se que há carência quanto a produção dos artefatos padrões da ER. A geração dos artefatos é importante pois eles servem de base para as demais atividades da ER e são fundamentais para o sucesso do projeto.
Apesar das abordagens dirigidas a modelos estarem em constante crescimento, o uso da ER ainda é limitado, sendo que essa é uma das fases fundamentais do desenvolvimento de *software*. Isso ocorre porque as principais abordagens da MDA focam nas estratégias de transformação de PIM para PSM e do PSM para o código, porém onde a ER possui mais ênfase, que é na transformação de CIM para PIM é dada pouca atenção. Consequentemente, há a grande possibilidade do produto não atender às necessidades do cliente uma vez que o uso das práticas de levantamento de requisitos é pouco abordado no contexto de DDM. Esse fator pode acarretar em uma manutenção custosa e pouca rastreabilidade. Diante desse contexto, nota-se a importância de uma proposta que envolvam todas as fases da ER para o DDM, pelas quais o DDM seja inteiramente aproveitado, havendo um controle e rastreabilidade dos requisitos, evitando retrabalho, insatisfação e atendendo aos requisitos dos clientes.

8 CONSIDERAÇÕES FINAIS

O objetivo desse trabalho foi descrever uma revisão sistemática com o intuito de conhecer o estado da arte da Engenharia de Requisitos e do DDM. Foram selecionados 4 artigos e por meio de análise notou-se que a utilização de modelos leva a problemas relacionados à ER, sobretudo ao acompanhamento de requisitos.
Tais problemas precisam ser solucionados, visto que a ER delimita o escopo do projeto e abrange todas as fases do processo de desenvolvimento de *software*. Por esse motivo há a importância de pesquisar e estudar sobre a ER para o DDM. Isso se deve ao fato de que com o DDM o tempo e o custo de desenvolvimento são reduzidos, já que uma solução é gerada automaticamente a partir do modelo conceitual definido no domínio da solução. Mas para alcançar resultados satisfatórios, o DDM deve estar em alinhado a Engenharia de Requisitos.
Os 4 artigos selecionados apresentam pontos de vista diferentes. Todos fizeram propostas para melhorar a ER na geração de *software* por meio de modelos, porém, conforme visto no item 7, os recursos da ER para o DDM ainda são pouco utilizados, nem todas as fases da ER são aproveitadas e a gerência dos requisitos torna-se um fator preocupante.
Para trabalhos futuros, é possível sugerir um estudo mais amplo da aplicação da ER para o DDM, avaliar benefícios e identificar limitações e problemas a serem solucionados. Além disso, as propostas encontradas podem ser refinadas a fim da obtenção de sucesso na separação da regra de negócio da tecnologia com o aumento do nível de abstração dos projetos de sistemas.

REFERÊNCIAS

AIGUIER, M.; GOLDEN, B.; KROB, D. **Modeling of complex systems II: A minimalist and unified semantics for heterogeneous integrated systems.** Applied Mathematics and Computation, v. 218, p. 8039–8055, 2012.

ALMEIDA, C. ***Qualitas*: Um Modelo de Processo de Desenvolvimento de *Software* Orientado a Modelos.** 25/02/2014. 143. Dissertação – Universidade Federal de Sergipe.

AMELLER, D.; FRANCH, X.; CABOT, J. **Dealing with non-functional requirements in model-driven development**. Proceedings of the 2010 18th IEEE International Requirements Engineering Conference, RE2010, p. 189–198, 2010.

AZOFF, M. **The Benefits of Model Driven Development, MDD in Modern Web-based Systems.** 2008. Disponível em: < http://goo.gl/gvhCdf>. Acesso em 12/12/2014.

BRAGA, V. T. **Um Processo para Projeto Arquitetural de Software Dirigido a Modelos e Orientado a Serviços**. 2011.

CHITFOROUSH, F.; YAZDANDOOST, M.; RAMSIN, R. **Methodology Support for the Model Driven Architecture**. 14th Asia-Pacific Software Engineering Conference (APSEC'07), p. 454–461, 2007a.

DEMIR, A. **Comparison of model-driven architecture and software factories in the context of model-driven development**. Proc. - Joint Meeting of the 4th Workshop on Model-Based Dev. of Computer-Based Systems and the 3rd Int. Workshop on Model-Based Methodologies for Pervasive and Embedded Software, MBD/MOMPES 2006, p. 75–83, 2006.

FALBO, R. A. **Engenharia de Requisitos: Notas de Aula.** Universidade Federal do Espírito Santo, Vitória, 2012.

FRANCETO, S. **Especificação e Implementação de uma Ferramenta para Elicitação de Requisitos de Software baseada na Teoria da Atividade.** 2005. Disponível em: < http://goo.gl/4wj9EZ>. Acesso em 10/12/2014.

IEEE. **Transactions on Software Engineering**, v. 34, n. 3, p. 377–394, 2008.

KILICAY-ERGIN, N.; LAPLANTE, P. A. **An online graduate requirements engineering course.** IEEE Transactions on Education, v. 56, n. 2, p. 208–216, 2013.

KOTONYA, G. e SOMMERVILLE, I. **Requirements Engineering – Processes and Techniques.** John Willy & Sons. (1997)

LAZARTE, I. M. *et al.* **Model-driven development methodology for B2B collaborations.** Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOC, p. 69–78, 2010a.

LEITE, J. **Engenharia de *Software.*** Disponível em: < http://goo.gl/43GUdb>. Acesso em 10/12/2014.

LONIEWSKI, G.; ARMESTO, A.; INSFRAN, E. **An architecture-oriented model-driven requirements engineering approach.** 2011 Model-Driven Requirements Engineering Workshop, MoDRE 2011, n. Cim, p. 31–38, 2011.

MACDONALD, A.; RUSSELL, D.; ATCHISON, B. **Model-driven development within a legacy system: An industry experience report.** Proceedings of the Australian Software Engineering Conference, ASWEC, v. 2005, n. Mdd, p. 14–22, 2005a.

MELLO, L.; MEYER, J. **Levantamento de Requisitos.** Disponível em: < http://goo.gl/yDb2Tm>, Acesso em 10/12/2014.

O. SOUZA, R. X.; LIVEIRA, A. A; NASCIMENTO, R. P. C. ModelER: **Abordagem baseada em modelos aplicada ao processo de elicitação de requisitos**. Proceedings of the 7th Euro American Conference on Telematics and Information Systems, p. 11:1–11:7, 2014.

SOMMERVILLE, I. **Requisitos do usuário e do sistema e do software.** 2004. Addison-Wesley, 6a. edição

_____. **Engenharia de software**. 8ª ed. São Paulo: Pearson Addison-Wesley, 2007.

SOMMERVILLE, I.; SAWYER, **P. Requirements Engineering – A Good Practice Guide.**1997.

TEXEIRA, L. **Engenharia de Requisitos Orientada a Aspectos.** Universidade Federal de Pernambuco, 2008.

RIVERO, J. M. *et al.* **Improving user involvement through a model-driven requirements approach.** 2013 3rd International Workshop on Model-Driven Requirements Engineering,

SILVA, L. *et al.* **C&L: Uma Ferramenta de Apoio a Engenharia de Requisitos.** 2004. Disponível em: < http://goo.gl/gHKBY9>. Acesso em 20/12/2014.

SWITHINBANK, P. *et al*. **Front cover Patterns: Model-Driven Development Using IBM.** Contract, p. 252, 2005.

VALDERAS, P.; PELECHANO, V. **Introducing requirements traceability support in model-driven development of web applications.** Information and Software Technology, v. 51, p. 749–768, 2009.

WESTFALL, L. **Software Requirements Engineering: What, Why, Who, When, and How By Linda Westfall**. ASQs Software Quality Professional Journal, n. 2004, p. 9–15, 2006.

WIEGERS, K.; BEATTY, J. **Software Requirements.** Microsoft. 3ª Edição. 2013.

WINKLER, S.; VON PILGRIM, J. **A survey of traceability in requirements engineering and model-driven development.** Software and Systems Modeling, v. 9, p. 529–565, 2010.

ZAVE, P. **Classification of research efforts in requirements engineering.** Proceedings of 1995 IEEE International Symposium on Requirements Engineering (RE'95), v. 29, n. 4, 1995.

ZOHREVAND, Z.; BIBALAN, Y. M.; RAMSIN, R. **Towards a framework for the application of Model-Driven Development in Situational Method Engineering.** Proceedings - Asia-Pacific Software Engineering Conference, APSEC, p. 122–129, 2011a.