Uma Abordagem Colaborativa de Modelagem Conceitual de Informação utilizando *Mind Maps*

Rafael Duarte[1], José Júnior[1], Ruben Araújo[1], Fernando Wanderley[2], Maria Lencastre[1]

[1] Departamento de Engenharia da Computação
Universidade de Pernambuco
Pernambuco, Brasil

[2] CITI, FCT
Universidade Nova de Lisboa
Caparica, Portugal
{rbd,jmsj2,rna,mlpm}@ecomp.poli.br, fernando.wanderley@gmail.com

Abstract. Recent studies have shown that the existence of a poor requirements elicitation or the lack of this activity, during the information system development, have generated a series of system failures and resulted in unsatisfied customers. Among several reasons, the no user participation in the conceptual modelling process stands out. Another important issue is considering the need for creating a conceptual model with the integration of various viewpoints, from different stakeholders. In this context, this paper has two main goals: the first one is trying to make the modeling process more inclusive for the customers/stakeholders, during the conceptual modeling, through the adoption of cognitive models such as Mind Maps; the second one, is trying to consolidate the multiple perspectives of the resulting conceptual models (each from a different stakeholder) using merge techniques. The expected result is to obtain ubiquitous conceptual models with higher quality.

Keywords. Requirements Engineering, Collaborative Elicitation, Conceptual Modeling, Mind Maps.

1 Introdução

Segundo Pfleeger [1], a Engenharia de Requisitos (ER) estabelece a definição de requisitos como um processo que engloba: elicitar, especificar, modelar, documentar e validar os desejos do cliente para o futuro sistema. Dentre as várias atividades do processo da ER, a elicitação é uma das mais importantes. Segundo Pressman [2] e Sommerville [3], a importância desse passo se deve aos custos de manutenção por erros cometidos nessa etapa, os quais são considerados os mais altos de todo o projeto, principalmente se forem descobertos nas fases finais do desenvolvimento. É nesta fase onde os requisitos são definidos de forma a representar as expectativas e necessidades dos usuários. Freitas [4] ressalta ainda que a elicitação é o processo que deve ser continuo em todo o desenvolvimento do *software*, requerendo em todas as iterações (especialmente as iniciais) a ativa participação de todos *stakeholders* envolvidos (clientes e engenheiros de requisitos). Desta forma, ainda de acordo com o autor [4], é nesta atividade que se produz um conhecimento à cerca do domínio do *software* co-

mum a todos, melhorando a comunicação e consecutivamente a consistência e entendimento dos requisitos. Lopes [5] evidencia que, requisitos que não representam a real necessidade dos usuários, incompletos e/ou inconsistentes, adicionalmente com a dificuldade para um bom entendimento das expectativas do usuário, provocam retrabalho, atraso no cronograma e elevação nos custos do projeto, por fim ocasionando a insatisfação do cliente.

O recente estudo realizado pelo *Standish Group* [6] mostra que 63% dos projetos não atendem às expectativas dos clientes quanto aos comportamentos e funcionalidades do sistema. De acordo com a pesquisa, há falta de uma documentação clara (de comum entendimento entre as partes interessadas) e precisa dos requisitos (no sentindo de representar realmente os interesses e objetivos do usuário). Assim, esses são os possíveis motivos para o fracasso de um projeto. Segundo Pfleeger e Sommerville [1, 3], os requisitos elicitados devem ser claros, de fácil compreensão, completos e consistentes. Ainda segundo os autores, na prática, este é um objetivo não trivial de se alcançar, pois os *stakeholders* interpretam os requisitos de maneiras diferentes (devido ao seu conhecimento prévio e específico), e em muitos casos, acaba-se em conflitos e inconsistências inerentes aos mesmos.

De acordo com Olivé [7], é na fase de elicitação dos requisitos que é produzido o modelo conceitual, artefato responsável por representar, abstrair e delimitar o domínio do *software* a ser desenvolvido. A qualidade e clareza da informação desse modelo está vinculada à percepção de todos os envolvidos no domínio em questão. Segundo Wanderley et. al. [8], quando ocorre uma falha na comunicação, geralmente informações são perdidas. Essa perda produz um modelo com pouca qualidade de representação do domínio. Porém, ainda segundo esses autores, modelos que possibilitam ser entendidos por diferentes especialistas permitem um melhor entendimento e, consequentemente, uma melhor comunicação entre esses especialistas (os que entendem do negócio) e a equipe de desenvolvimento (profissionais da área de tecnologia da informação).

Os modelos apresentados neste artigo são voltados para a representação conceitual do domínio, sendo representados através de entidades e suas relações. Outras abordagens para elicitação de requisitos, como por exemplo, modelos orientados a metas ou requisitos não funcionais, não são abordados aqui.

Com o objetivo de facilitar a elicitação de requisitos, este artigo propõe a criação de uma abordagem colaborativa para modelagem conceitual por meio da utilização de mapas mentais. Assim, tem-se como principais contribuições da proposta: *(i)* a participação inclusiva do cliente no processo de modelagem conceitual a partir da adoção de mapas mentais; *(ii)* a definição de um algoritmo para a verificação de similaridades dos modelos conceituais; e por fim *(iii)* a produção de um modelo conceitual de informação ubíquo, ou seja, de comum entendimento a todos os envolvidos.

O presente artigo encontra-se estruturado nas seguintes seções: a Seção 2 aborda o referencial teórico necessário para o entendimento do problema; a Seção 3 apresenta a solução proposta; a Seção 4 apresenta um estudo de caso; a Seção 5 apresenta trabalhos relacionados; por fim, a Seção 6 apresenta conclusões e trabalhos futuros.

2 Fundamentação Teórica

2.1 Modelagem Conceitual da Informação

De acordo com Larman [9] e Cougo [10], a modelagem conceitual de informação consiste na representação dos objetos, suas características e seus relacionamentos, realizada de uma forma fiel ao ambiente observado. Essa representação independe do uso de técnicas de implementação, tecnologias ou dispositivos físicos. Nesse modelo, devem-se representar os conceitos e características observados em um dado ambiente, voltando-se simplesmente ao aspecto conceitual. A representação do ambiente observado, usando-se a modelagem conceitual, ocorre por meio de entidades ou classes, suas características e seus relacionamentos. Uma entidade/classe é uma categoria atribuída ao conjunto de objetos existentes neste ambiente que estão agrupados em função de suas semelhanças. Os objetos representados passam a ser denominadas instâncias destas entidades [11].

Segundo Fowler [12], para se construir um produto ou sistema com propósito específico, é necessário o desenvolvimento de um modelo conceitual que atenda às necessidades do mesmo. Para que o sistema execute funções, é preciso possuir algum conhecimento sobre o domínio e sobre as funções que o mesmo deverá executar. Esse conhecimento é chamado de esquema conceitual. De acordo com Olivé [7], para desenvolver um sistema de informação é necessário e suficiente definir seu esquema conceitual.

Neste artigo, o esquema conceitual será representado por diagramas de mapas mentais, com o objetivo de melhorar seu entendimento e uso. Neste trabalho pretende-se manter um modelo ubíquo, independente de plataforma, além de melhorar a compreensão do projeto com a finalidade de facilitar a adaptação de novos integrantes da equipe e a adaptação do projeto a novos ambientes de implementação.

2.2 Mapa Mental

O Mapa Mental (do inglês *Mind map*) é um modelo de *brainstorming*, idealizada por Buzan [13], voltado para a gestão de informação. Segundo o autor, os modelos mentais são utilizados na estruturação de ideias para compreensão e soluções de problemas além de auxiliar a memorização e aprendizado.

A técnica para construir modelos mentais permite representar raciocínios, ideias e informações por meio de imagens, palavras-chave, cores e associações. Sua estrutura, que se assemelha à forma como as informações são armazenadas na mente humana, facilita o desenvolvimento e a geração de ideias de forma imediata, integrando a capacidade imaginativa com a lógico-racional, facilitando a tomada de decisão por viabilizar uma visão global do assunto [14].

Nos mapas mentais, os elementos são organizados de forma intuitiva, de acordo com a relevância dos conceitos relacionados a um assunto central. Por sua vez, esses conceitos relacionados são organizados em agrupamentos, ramificações ou áreas, como se pode ver na Fig. 1, que representa o domínio de aluguel de carros, onde são modelados os seus principais elementos e respectivos atributos. A partir do nó central, *Aluguel de Carro*, surge nós de maior relevância, como o nó *Cliente*, e este dá origem

a nós que são seus atributos que por sua vez podem gerar novos nós atributos, como o caso, por exemplo, de *Telefone*.

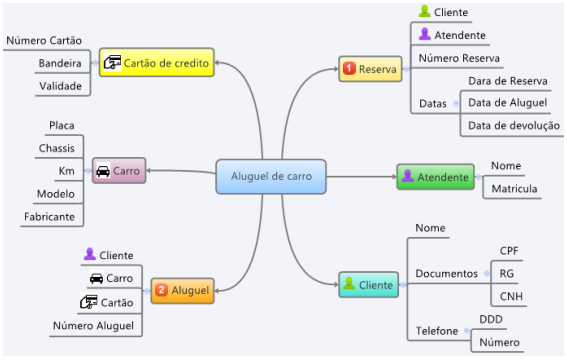

Fig. 1. Exemplo de um mapa mental representativo do domínio de aluguel de carro

Em outras palavras, um mapa mental é um diagrama radial que por meio de vocabulário (palavras-chave), distribuído em certa ordem e correlacionado, pode representar e modelar cognitivamente um conceito ou um domínio específico. Segundo Buzan [13], quando um mapa mental é elaborado, cada nó está associado com o restante do mapa mental; os principais benefícios de um mapa mental são: organização, utilização de palavras-chave, associação, agrupamento de ideias, memória visual, criatividade e inovação com simplicidade.

Normalmente, as pessoas têm diferentes preferências de linguagens de modelagem. De acordo com Siau e Tan [15], a forma diferenciada de como acontece o processo cognitivo em cada pessoa reflete em suas escolhas. Enquanto que os *stakeholders* se identificam mais com representações visuais (característica presente nos mapas mentais), analistas e desenvolvedores tendem a ser aproximar de linguagens formais. Assim, o uso da modelagem conceitual produz benefícios diretos no processo de elicitação de requisitos, como pode ser observado a partir das vantagens proporcionadas por tipos de estruturas como os próprios mapas mentais.

3 Proposta Colaborativa para Modelagem Conceitual

Esta seção tem por objetivo detalhar a proposta de solução para o problema de elicitação de requisitos na fase de projeto do modelo de domínio. Para tal propósito, é necessário consolidar essas diferentes visões, que podem ser conflitantes como também complementares entre si, em um único modelo conceitual. Pretende-se, como já mencionado, produzir um modelo ubíquo que espelhe o domínio, facilitando a comunicação entre os *stakeholders*.

Com esse objetivo, foram adotados modelos como os mapas mentais, para representação dos esquemas conceituais (modelos conceituais). Para cada *stakeholder*, considera-se que é gerado um mapa mental, que representa o seu ponto de vista do domínio. O conjunto de mapas mentais de um domínio especifico será o artefato de entrada da ferramenta, que então consolidará esse conjunto em um único mapa mental. O mapa mental consolidado, possivelmente, representará um mapeamento mais completo do domínio.

3.1 Ferramenta de *Merge*

O diagrama de componentes da ferramenta, proposta neste artigo, é apresentado na Fig. 2.

Fig. 2. Diagrama de componentes da ferramenta proposta

Nessa figura, é possível notar os quatro componentes principais, detalhados a seguir:

- **Leitor de mapas mentais:** esse módulo é responsável pela leitura dos mapas mentais no formato XML (*eXtensible Markup Language*), e, consequentemente, transpor para um estrutura interna em árvore contendo todos os nós (pais e filhos) dos mapas mentais;
- **Verificador de similaridade:** esse módulo recebe como entrada o mapa mental importado pelo componente anterior. Nesse momento, o usuário deve selecionar em qual domínio ocorrerá o *merge*, para então ser solicitado ao *Gerenciador de banco de dados* o mapa mental relativo ao domínio solicitado. A partir desses mapas mentais, são realizadas as comparações (apresentadas na seção 3.4) e, por fim, gerado um modelo mental consolidado;
- **Gerenciador do banco de dados:** esse módulo é responsável por buscar e armazenar os mapas mentais requisitados pelo componente anterior;
- **Escritor de modelos mentais:** esse módulo é responsável por converter a estrutura interna utilizada (em árvore) para representar os modelos mentais na ferramenta em questão para o mesmo formato em XML que possibilitará a importação para os editores de mapas mentais, que foram inicialmente utilizados.

A Fig. 3 ilustra a interação desses componentes por meio de uma execução.

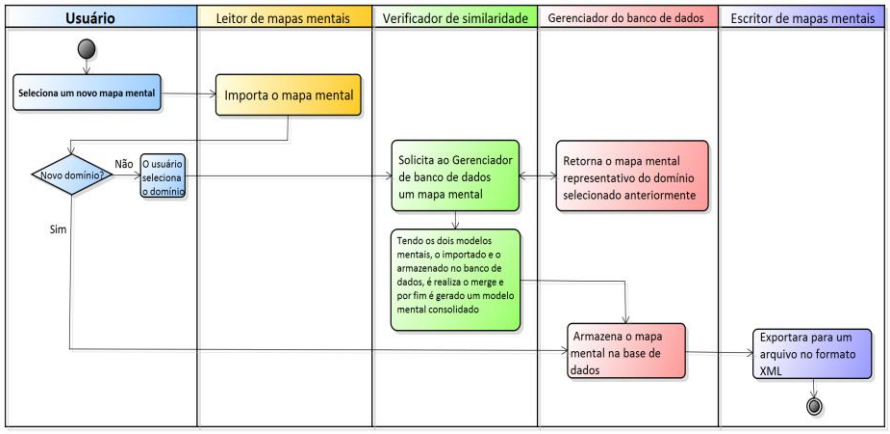

Fig. 3. Interação dos componentes

A execução é dita semiautomática, por não ser possível garantir todas as similaridades automaticamente; sendo assim, a ferramenta agrupa as possíveis similaridades pertencentes a um mesmo nível e auxilia o usuário a decidir se deve ou não ser realizado o *merge*. A Fig. 4 apresenta a interface da ferramenta onde é possível visualizar o passo semiautomático, no qual, o usuário deve selecionar quais nós serão utilizados durante o processo de *merge*. Caso a reposta seja negativa, ou seja, o usuário não considere uma palavra similar a outra, as duas palavras são armazenadas na base de dados.

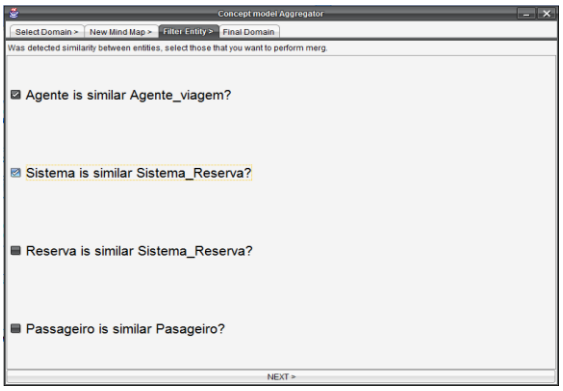

Fig. 4. Tela da ferramenta que auxilia no processo de *merge*

3.2 Visão Geral do Processo

Esta subseção detalha o fluxo de execução proposto para a ferramenta, de modo a proporcionar uma visão geral do seu funcionamento. A Fig. 5 mostra um infográfico descritivo do sistema. Nele, é possível ver as principais atividades que denotam o processo necessário para construção do mapa mental que agrega as diferentes visões dos *stakeholders*.

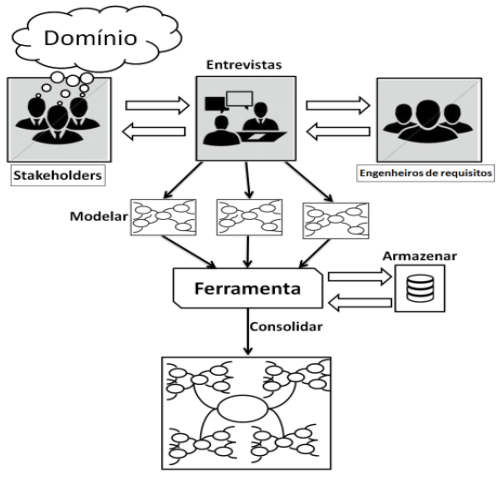

Fig. 5. Infográfico representativo da ferramenta

O fluxo de execução do processo segue o seguinte conjunto de atividades:

- **Entrevistar:** inicialmente, têm-se entrevistas entre os especialistas de domínio e o engenheiro de requisitos. Para servir como mentor desta atividade, além do *template* definido, existe o *guideline* contendo as diretrizes e dicas para a construção "correta" dos mapas mentais (detalhados na subseção seguinte);
- **Modelar:** como cada especialista possui entendimentos e visões diferentes do domínio, ao fim da atividade *Entrevistar* são produzidos vários mapas mentais para um mesmo domínio do problema;
- **Armazenar:** o primeiro mapa mental, para um dado domínio, é inserido completamente na base de dados, ao passo que a partir do segundo existirá um processo de *merge* com o que já foi armazenado, realizando a consolidação dos modelos (mapas mentais). Essa etapa de *merge* é feita em duas: (*i*) a primeira, onde os artefatos considerados novos são armazenados no banco de dados com os demais existentes; e (*ii*) na segunda, os artefatos considerados já existentes na base de dados são agrupados e é requerido do engenheiro de requisitos que faça o passo semiautomático, a fim de evitar integrações indevidas. Esse processo é feito de modo que as estruturas que são semelhantes não são armazenadas mais de uma vez no repositório, e, finalmente;
- **Consolidar**: a aplicação permite exportar um único mapa mental que representa um domínio. Este mapa mental agrega todos os mapas mentais anteriores, ou seja, o mesmo apresenta as diversas visões dos vários especialistas dos domínios. Esse mapa mental é uma representação do domínio, que segundo Wanderley *et al.* [8] pode ser transformado em um modelo conceitual de informação.

A princípio os modelos mentais podem se construídos de forma livre e cada usuário pode representar o mesmo domínio de forma diferente usando mapas mentais. Diante disto, fez-se necessário a elaboração de uma estrutura genérica para o processo de elicitação de requisitos por meio da construção dos modelos mentais, para que em trabalhos futuros possam ser definidos verificadores semânticos e sintáticos, com o auxílio, por exemplo, da definição de *domain-specific language* (DSL) para modelos mentais, promovendo maior consistência ao modelo utilizado.

A Fig. 6 ilustra o template definido. Nele é possível identificar os artefatos básicos que o engenheiro de requisitos precisará descrever durante a entrevista feita com os especialistas de negócio. A estrutura radial do template é apropriada para o metamodelo do mapa mental.

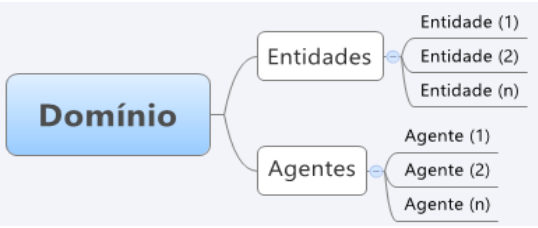

Fig. 6. *Template* para modelagem dos modelos mentais

Na Fig. 6, o *Domínio* corresponde à natureza do espaço do problema a ser resolvido. Uma *Entidade* está relacionada a um objeto do domínio, conforme mencionado anteriormente, e é um artefato do sistema que tem a dependência de um *Agente* para que mude seu respectivo estado. Ou seja, uma *Entidade* é um artefato passivo. Para cada *Entidade* é possível especificar seus atributos internos. Os *Agentes* são artefatos ativos no sistema. Logo, eles podem promover mudanças de estado do sistema, sendo possível também especificar seus atributos e propriedades internas. A seguir, um breve detalhamento acerca do *guideline* proposto é utilizado como suporte na construção deste *template* apresentado.

3.3 *Guideline* de Boas Práticas

Uma vez definido o *template* a ser utilizado como estrutura (metamodelo) dos modelos de mapas mentais, identificou-se a necessidade de se ter um guia com regras e boas práticas (conjunto de etapas) que auxiliasse a criação destes modelos. O mesmo é descrito a seguir:

- Definir o *Domínio* como nó central;
- Usar apenas uma palavra para representar *Entidades*, *Agentes*, (visando facilitar o processo de *merge* que acontece no momento de inserção de um artefato no banco de dados). Em caso de necessidade de palavras compostas para identificar *Entidades* ou *Agentes*, as mesmas devem ser concatenadas por *underline* ("_");
- Usar o identificador de dois pontos **(":"**) para referir-se à *Entidade* e arroba (**"@"**) para *Agente*, fora de suas respectivas subáreas;
- Não usar caracteres especiais, como por exemplo: ç, ~, ´, etc.

3.4 Discussão à Respeito de *Merge* dos Modelos

No momento em que os modelos são comparados, estas análises são feitas basicamente por meio de uma busca do texto existente em cada nó do modelo (como dito no *guideline*, geralmente os textos se resumem em palavras). No entanto, para as *Entidades* e os *Agentes*, como se trata apenas de palavras, a comparação é feita pelo próprio sistema. Existem situações distintas no processo de comparação que exigem soluções diferentes, sendo elas:

- **Mesma palavra, porém com caixa alta e caixa baixa diferente:** as palavras são as mesmas, mas, com caracteres maiúsculos e minúsculos diferentes. Por exemplo, considere a palavra "*Casa*" e "*casa*". Como a maioria das linguagens de programação imperativa são *case sensitive*, o mesmo caractere estando em maiúsculo é diferente de sua versão minúscula. Logo, estas palavras são consideradas diferentes. No contexto dessa abordagem, todas as palavras são internamente colocadas em maiúsculas;
- **Erro de acentuação:** é a mesma palavra, entretanto, uma está acentuada e a outra não. Neste caso, como exemplo, considere a palavra "*táxi*" e que a mesma será comparada com sua versão não acentuada "*taxi*". Sendo essa situação implementada como o auxílio da classe *java.text.Collator*, que realiza comparações ignorando os caracteres especiais;

- **Palavras compostas:** uma palavra é parte da outra; considere, por exemplo, que um modelo possua um nó "*sistema_reserva*" e em outro exista o nó "*sistema*". Estas duas palavras representam o mesmo conceito em um determinado domínio. Para tal situação, basta usar um método de comparação que considere parte de uma palavra em outra. É importante salientar que, uma vez identificado que uma palavra está contida em outra, o *merge* não pode ser realizado imediatamente, pois, diferente das outras situações, neste caso não se tem a certeza que as palavras representam o mesmo conceito. Considere agora que a palavra "*sistema_reserva*" foi comparada com a palavra "*reserva*". Considere ainda que estas palavras não representam o mesmo conceito no domínio. Nesta situação, será adotada que as palavras são consideradas iguais, todavia, não representam o mesmo conceito. Neste caso, as palavras são agrupadas para que o engenheiro de requisitos manualmente identifique que as palavras realmente representam o mesmo conceito, para estes tipos de situação;
- **Erro ortográfico:** é a mesma palavra, mas ocorreu um erro de ortografia. Neste caso, considere, por exemplo, que um modelo possua um nó "*viagem*" e em outro exista a mesma palavra, entretanto, com um erro ortográfico ("*vigem*"). Estas duas palavras representam o mesmo conceito em um determinado domínio, mas são palavras distintas. Para tais ocasiões, foi utilizado o algoritmo chamado Distância Levenshtein ou Distância de Edição. A Distância Levenshtein, desenvolvida pelo russo Vladimir Levenshtein, determina o grau de similaridade entre duas palavras. Em Manning *et. al.* [16], os autores afirmam que o algoritmo mede a distância de edição entre duas palavras, calculando quantas operações são necessárias para transformar uma palavra em outra não importando o tamanho. As operações podem ser entendidas como sendo: substituição, deleção e inserção de caracteres. Após a execução do algoritmo, se a distância foi igual ou inferior a três operações, as duas palavras podem ser consideradas iguais, mas podendo ter algum erro ortográfico. Neste caso, novamente não se tem certeza de que duas palavras similares representam o mesmo conceito, sendo novamente agrupadas para que o engenheiro de requisitos identifique quais palavras similares realmente representam o mesmo conceito e realize manualmente o *merge*.

Com o objetivo de ilustrar a aplicabilidade do processo definido (com um conjunto de atividades, *template* e *guideline*), a seção a seguir apresenta um estudo de caso relacionado ao domínio de agências de viagens e suas reservas de passagem área.

4 Estudo de Caso

O estudo de caso refere-se ao domínio de uma agência de passagem aérea, mais especificamente à tarefa de reservar uma passagem aérea. Para validar o *merge* entre modelos mentais, foram selecionados três profissionais da área de TI que possuem diferentes níveis (básico, intermediário e avançado) de conhecimento do domínio.

Para classificar o nível de conhecimento desse domínio, foram escolhidos um profissional que trabalhou em sistema de reserva aérea (*Usuário1*), um profissional que viaja constantemente (*Usuário2*) e um usuário leigo que não possui tanto contato com o domínio (*Usuário3*). Este estudo de caso não se trata de um experimento

controlado, a ideia é apenas utilizar um conjunto de usuários com diferentes níveis de conhecimento do domínio para testar a efetividade das técnicas de *merge* dos modelos mentais.

4.1 Descrição do estudo de caso

No domínio escolhido, algumas regras básicas foram estabelecidas, conforme detalhado a seguir:

- Antes do agente de viagem solicitar uma reserva aérea, é necessário coletar alguns dados do passageiro; estes incluem: para onde o mesmo deseja ir, período da viagem, se possui acompanhante, preferências de horário, valores e companhia aérea. Em seguida, o agente de viagem ingressa no sistema de reserva de passagens áereas e informa os dados levantados na entrevista com o cliente;
- O agente de viagem realiza uma busca em seu sistema tentando encontrar voos com as características solicitadas pelo passageiro;
- Caso o resultado da busca seja negativo (ou seja, se nenhum voo com as características solicitadas for encontrado), as características devem ser revistas para encontrar inconsistência. Caso o resultado seja positivo, a reserva da passagem em questão é efetuada no sistema;
- Para concluir a compra da passagem aérea, o pagamento da mesma deve ser feita em até 48 horas, caso contrário ocorrerá o cancelamento da reserva.

Sabendo destas regras, os três usuários foram colocados em ambientes distintos, sem comunicação uns com os outros, a fim de que seus comentários, dúvidas ou sugestões não influenciassem o processo de construção dos modelos mentais. Decorridos aproximadamente vinte minutos, os modelos mentais foram criados.

As Fig. 7, 8 e 9 são os mapas mentais modelados pelos respectivos usuários *Usuário1*, *Usuário2*, *Usuário3*. Apesar dos diferentes níveis de conhecimento, os usuários modelaram de forma semelhante o domínio. De acordo com o procedimento de *merge*, os nós destacados em vermelho foram considerados com os mesmos conceitos semelhantes e, portanto, sofreram o *merge* de forma automática. Por exemplo, o nó *Viagem* existe nos três modelos e por isso seus nós descendentes podem ser comparados e inseridos sem redundancia na base de dados. Porém, com os nós destacados em verde não se tem essa certeza e, por isso, eles necessitaram da intervenção de um usuário para decidir quais os nós deveriam sofrer o *merge*, como exemplo os nós *agente_viagem* e *agente*. Neste caso, como uma palavra está contida na outra, o algoritmo entende que são similares, porem não tem total certeza, portanto, nestes casos, os nós são agrupados e apresentados no passo semiautomatico da ferramenta para que o usuario decida se deve ou não ocorrer o *merge*. Por fim, os nós em azul que não obtiveram nenhuma similaridade e, por isso, não foram inseridos em nenhum *merge*. Considerando o nó *Pagamento* como exemplo, não foi detectado similaridade com nenhum nó existente, sendo inserido diretamente (sem sofrer *merge*) na base de dados.

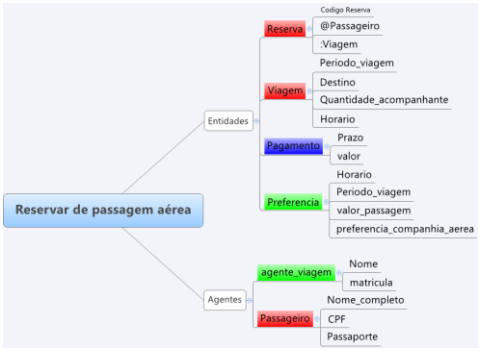

Fig. 7. Mapa mental feito pelo *Usuário1*

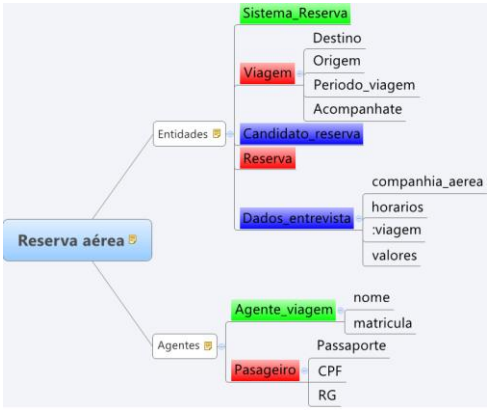

Fig. 8. Mapa mental feito pelo *Usuário2*

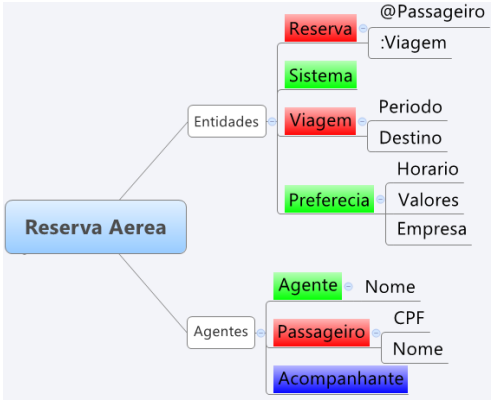

Fig. 9. Mapa mental feito pelo *Usuário3*

Dentro desse contexto, os três modelos mentais (Fig. 7, 8 e 9) são considerados como artefatos de entrada para ferramenta, que após o processo de *merge* gera, de

forma semiautomática, um novo modelo consolidado (Fig. 10), que deve representar a visão dos três usuários.

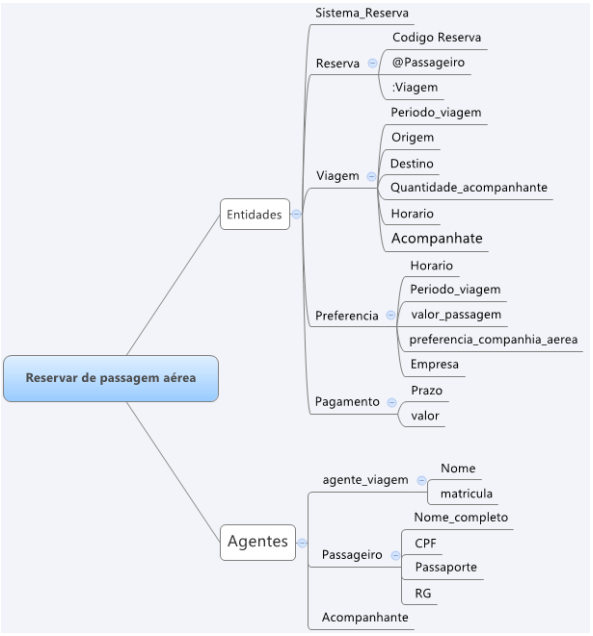

Fig. 10. Mapa Mental consolidado do domínio de Reserva aérea

5 Trabalhos Relacionados

Promover melhorias no processo de modelagem conceitual de um domínio tem sido objeto de estudo de pesquisa há algum tempo. Pastor e Molina [17] definiram o método Oasis que combina técnicas formais de especificação para modelagem conceitual com metodologias de desenvolvimento orientada a objetos convencionais. Ainda existe a ontologia que, segundo Guizzardi [18], é efetiva em diferentes aplicações ao solucionar algumas das deficiências encontradas na representação do conhecimento de um domínio. Tanto o método Oasis quanto a ontologia são abordagens excelentes por prezarem pela corretude, toda via por conta de seu formalismo, não é atrativa para o usuário.

Existem outras formas de representar o domínio, similares aos mapas mentais, como exemplo os mapas de tópicos. De acordo com Park [19], os mapas de tópicos baseiam-se na representação de conceitos (tópicos) através de relações (associações); a partir dessas ligações de conceitos e associações são construídas as "trilhas" que apontam para as informações relevantes para um domínio. Librelotto [20], afirma que os principais benefícios dos mapas de tópicos são: estruturar recursos de informação através de mecanismos externos a estes recursos, permitir buscas que recuperem a informação desejada e criar visões diferentes para usuários ou aplicações específicas, através de filtros sobre as informações. Ainda existe o mapeamento cognitivo; Marx e Dagan [21] afirmam que mapas conceituais visam organizar e representar o conheci-

mento, assim os mapas são estruturados a partir de conceitos fundamentais e suas relações, tendo por objetivo reduzir de forma analítica, a estrutura cognitiva subjacente a um dado conhecimento, aos seus elementos básicos. Siau e Tan [15] descrevem um estudo de caso que mostra as vantagens de se usar mapeamento cognitivo em sistemas de modo a promover melhorias na qualidade de modelagem conceitual dos mesmos. Essas técnicas são similares aos modelos mentais, entretanto são mais rígidas quanto à semântica de seus elementos e a abordagem deste artigo propõe uma abordagem mais simples, facilitando o entendimento pelo usuário.

Existem ainda outros trabalhos relacionados focados em mapas mentais. Wanderley *et. al.* [8] sugere uma abordagem de melhoria no processo de análise de domínio por meio do uso de modelos mentais na elaboração de modelos de *software*. Hiranabe [22], por sua vez, propõe o uso de mapeamento mental relacionado com o conceito de "*User Wish*", uma forma diferenciada de usar requisitos antes dos mesmos serem formalizado; neste caso, o objetivo do autor é a transformação (não automática) de modelos mentais em mapas UML. Já Mahmud [23] relata resultados obtidos com experimentos envolvendo especialistas e não especialistas em modelos mentais para a modelagem de requisitos dentro do contexto de um projeto que adota a metodologia ágil. O diferencial da abordagem proposta neste artigo é facilitar o entendimento do cliente de como modelar o modelo mental criando uma forma de comunicação clara e de comum entendimento entre toda a equipe de desenvolvimento e os *stakeholders.*

6 Considerações Finais

Este artigo foi motivado pela necessidade de se facilitar a participação ativa do cliente no processo de geração de modelos conceituais, adotando mapas mentais como modelo conceitual. Para tal, foi definido um algoritmo para checagem de similaridade dos modelos conceituais.

Outra importante contribuição foi o desenvolvimento de uma ferramenta que recebe os vários modelos conceituais, representativo das várias perspectivas de um domínio e, utilizando-se das técnicas de similaridade, realizar o *merge* desses modelos. Como produto final tem-se um modelo conceitual de informação ubíquo, ou seja, de comum entendimento a todos os envolvidos.

Como trabalhos futuros, estão previstos mais estudos de caso em ambiente controlado, com um número maior de participantes, e com condição industrial, com foco em outros domínios, visando à melhoria do processo e seus artefatos internos, além de validar a eficácia do algoritmo de similaridade dos modelos. Estender o algoritmo de *merge* levando em consideração a hierarquia dos modelos e, por fim, desenvolver uma ferramenta no ambiente *Web* para prover a modelagem distribuída, favorecendo a participação ativa dos usuários.

Referências

1. Pfleeger, S. L.: Engenharia de Software: Teoria e Prática. 2ª Edição, São Paulo (2004)
2. Pressman, R.S., Software Engineering, McGraw-Hill, 6ª edição, (2006)
3. Sommerville, I.: Engenharia de Software. 9ª Edição. Pearson Education, São Paulo (2011)

4. Freitas, D. P., Borges, M. R. S., Araújo, R. M.: Colaboração e Negociação na Elicitação de Requisitos. Em: Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software. IDEAS 07, Ilha de Margarita (2007)
5. Lopes, P. S. N. D.: Uma Taxonomia da Pesquisa na Área de Engenharia de Requisitos. Instituto de Matemática e Estatística da Universidade de São Paulo, São Paulo (2002)
6. O CHAOS Manifesto do Standish Group International publicado em 2011, http://versionone.com/assets/img/files/ChaosManifest_2011.pdf
7. Olivé, A.: Conceptual Modeling of Information Systems. Springer, New York (2007)
8. Wanderley, F., Silveira, D. S., Araujo, J., Lencastre, M.: Generating feature model from creative requirements using model driven design. In: Proceedings of the 16th International Software Product Line Conference - volume 2 – pages 18-25. ACM, New York (2012)
9. Larman, C.: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3.ed. Prentice-Hall, New York (2004)
10. Cougo, P. S.: Modelagem conceitual e projeto BD. Campus, Rio de Janeiro (1997)
11. Schmitz, E. A., Silveira, D. S.: Desenv. de Software OO. Brasport, Rio de Janeiro (2000)
12. Fowler, M.: Analysis Patterns, Reusable Object Models. Addison Wesley, London (1997)
13. Buzan, T.: The Mind Map Book. BBC Active, London (2003)
14. Millen, D., Schriefer, A., Lehder, D., & Dray, S., Mind Maps and Causal Models: Using graphical Representations of Field Research Data. Extended Abstracts. CHI '97 - Human Factors in Computing Systems. March (1997)
15. Siau, K., Tan, X.: Improving the quality of conceptual modeling using cognitive mapping techniques In: The International Workshop on Conceptual Modeling Quality. Springer, Verlag Berlin Heidelberg (2005)
16. Manning, C. D., Raghvan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
17. Pastor, O.; Molina, J.C.: Model-Driven Architecture in Practice: A Software Production Environment Based on Conceptual Modeling. Springer Publisher (2007)
18. Guizzardi, G. Análise de Domínio e Ontologias. Vitória, 2000. 136 f. Universidade Federal do Espírito Santo, Vitória (2000)
19. Park, J.; Hunting, S. XML topic maps: creating end using topic maps for the web. Boston: Addison Wesley, p. 644 (2003)
20. Librelotto, G. R. Topic maps: da sintaxe à semântica. p. 333 (Tese Doutorado em Informática) – Escola de Engenharia da Universidade do Minho, Braga, Portugal, (2005)
21. Marx, Z.; Dagan. I; Conceptual mapping through keyword coupled clustering, DOI 10.1007/BF02512360 (2001)
22. Hiranabe, K.: Exploring User Requirements through Mind mapping, http://www.change-vision.com/en/ExploringUserRequirementsThroughMindMapping_Letter.pdf
23. Mahmud, I.: Mind-mapping: An Effective Technique to Facilitate Requirements Engineering in Agile Software Development. In: Proceedings of 14th International Conference on Computer and Information Technology. IEEE, Kansas (2011)