Um modelo para tratamento de ambiguidades em requisitos de evolução de sistemas jurídicos baseado em mapeamento conceitual

Luiz Gustavo Ferreira Aguiar, Rebeca Teodoro da Silva, Elias Canhadas Genvigir, José Augusto Fabri, Alexandre L´Erario

Universidade Tecnológica Federal do Paraná – Campus Cornélio Procópio

lgfaguiar@hotmail.com, rebeca.teodoro@gmail.com, elias@utfpr.edu.br, fabri@utfpr.edu.br, alerario@utfpr.edu.br

Abstract. O Domínio Judicial Brasileiro possui sistemas em processo de evolução, principalmente os sistemas de processo judicial eletrônico. A participação de diferentes usuários, concomitante às complexidades do direito, geram requisitos de evolução de software contendo ambiguidades. Este trabalho apresenta um modelo para captação e tratamento desses requisitos, através de *feedback* dos usuários, utilizando mapeamento conceitual. O modelo proporciona a delimitação do escopo do requisito e sua conceituação, sendo importantes contribuições para a compreensão do requisito, reduzindo a incidência de ambiguidades. Devido à complexidade do direito, a linguagem natural ainda é predominante para os requisitos, contudo, o mapeamento conceitual demonstra-se eficiente para a representação dos conceitos da área, bem como de fácil aprendizado.

Keywords: Software *Evolution*, Engenharia de Requisitos, Mapas Conceituais

1 Introdução

Um software bem sucedido é um produto de mudanças provindas de requisitos de desenvolvimento, tecnologias e conhecimento dos *stakeholders*, sendo estas responsáveis pela evolução do software [1]. Originalmente, a evolução de software surgiu como um fenômeno inesperado [2], mas que, com o decorrer dos anos, ganhou a atenção dos engenheiros de software, tornando-se o maior estágio do desenvolvimento de software, incluindo novas perspectivas como o desenvolvimento ágil, *open-source* e outros. Pesquisas tem sido realizadas, principalmente com foco na tarefa fundamental da evolução de software que é a mudança do software [1].

Ao tratar da mudança do software o *feedback* do usuário é fundamental para a melhoria da qualidade pois constitui-se de uma rica fonte de informações para a melhora de futuras versões do software, sendo a força motriz para a sua evolução [3]. No entanto, a principal fonte de defeitos em um software provém de uma análise deficiente dos requisitos [4–6].

O domínio judicial, amplia a perspectiva da questão da análise deficiente de requisitos dentro da evolução do software. Isso se deve à complexidade do direito e a ocorrência de ambiguidades, visto a existência e concorrência de múltiplas leis, estatutos e normativas [7], necessitando de métodos e ferramentas que auxiliem na análise dos requisitos de usuários e qualidade da evolução de seus softwares.

Propõe-se neste artigo um modelo de coleta e tratamento de requisitos para a evolução de software do domínio judicial, por meio de *feedback* dos usuários em mapas conceituais para contextos de sistemas judiciais, que permita a redução de ambiguidades e a melhoria na compreensão dos requisitos pelos engenheiros de software.

Atualmente, dentro do domínio judicial, devido a ocorrência de ambiguidades, esses requisitos são efetuados em linguagem natural [8], dentro de um processo *ad hoc* e propenso a erros [9]. Um modelo conceitual das leis e textos legais pode atuar como um orientador para a representação dos requisitos [10]. O mapeamento conceitual é uma técnica conhecida pela sua facilidade de uso e rápida implementação. Os Mapas Conceituais [11–13] estabelecem uma estrutura hierárquica baseada em conceitos, que permite ganho cognitivo e facilita a compreensão.

O presente artigo está organizado da seguinte forma: na sessão 2, apresenta-se a importância do *feedback* do usuário para a evolução de um sistema de software; na sessão 3, analisa-se a estrutura do Domínio Judicial Brasileiro, o processo judicial eletrônico e sua iminente necessidade de evolução; na sessão 4, a complexidade dos requisitos evolutivos do domínio judicial, no que tange a características de ambiguidade, é trabalhada; na sessão 5, apresenta-se e discute um modelo para captação dos requisitos evolutivos do domínio judicial utilizando mapeamento conceitual; por fim, na sessão 6, são realizadas considerações sobre o trabalho apresentado.

2 O papel do usuário na evolução do software

Para a Engenharia de Software, a evolução do software é a fase central do ciclo de vida do software, de forma que para um software ser bem sucedido, uma grande parcela de tempo e recursos são gastos na evolução, merecendo assim uma atenção especial dos pesquisadores [1].

A mudança é o princípio básico da evolução de um software, de modo que a evolução do software é o estágio do ciclo de vida onde são realizadas alterações substanciais. Tais mudanças podem ser abordadas de duas maneiras: de forma preventiva, onde é possível antecipar-se e preparar-se para o obvio, baseado em experiências passadas; ou de forma reativa, onde enfrenta-se mudanças inesperadas. Esta última, é essencial para a prática da evolução do software, necessitando de ferramentas e metodologias [14].

O envolvimento do usuário no processo de software mudou significativamente nas últimas décadas, passando de programadores e pessoal treinado para praticamente qualquer pessoa, implicando em mudança de atitude dos desenvolvedores em relação aos usuários [15]. Há uma distância maior entre os usuários e desenvolvedores, no entanto, é necessário manter o foco nos usuários para atender suas crescentes demandas, tornando o *feedback* do usuário cada vez mais importante para os desenvolvedores [16, 17].

O *feedback* do usuário contém informações importantes para os desenvolvedores, ajudando a melhorar a qualidade do software, bem como identificar recursos ausentes [15].

3 O sistema judiciário brasileiro e perspectivas de evolução dos sistemas de processo eletrônico

O sistema judiciário brasileiro é organizado em unidades autônomas, respeitando cada unidade federativa do Brasil [18]. Observando uma combinação de natureza judicial, localização física e grau de jurisdição, a estrutura do sistema judiciário brasileiro possui um número superior à 100 unidades [19].

Essa estrutura não possui uma coordenação central, tendo em vista que suas instancias são autônomas, inclusive quanto ao uso de recurso de tecnologia da informação [18]. O CNJ (Conselho Nacional de Justiça), visando auxiliar principalmente os tribunais que não possuem estrutura e recursos para desenvolver ou adquirir uma solução tecnológica de porte, desenvolveu sistemas de forma gratuita aos tribunais. Num primeiro momento disponibilizou o sistema Projudi[1], e atualmente, o PJe[2]. O sistema de processo eletrônico deve ser abrangente suficientemente para atender as várias instâncias do judiciário brasileiro, com seus milhares de usuários, com perspectivas e anseios de diferentes culturas e necessidades.

O PJe é um sistema em fase de evolução, e como já não bastasse os esforços pertinentes a tal fase, o sistema foi desenvolvido observando os ditames da Lei 5.869, porém, será substituída pela Lei 13.105, a partir de 16 de março de 2016, implementando mudanças substanciais quanto ao processo judicial.

Os sistemas de processo judicial eletrônico são sistemas complexos em plena evolução, inseridos em um contexto peculiar. Há perspectivas que apontam tais como ecossistemas de software (ECOS). Um ECOS é composto por sistemas e subsistemas de negócio que se interagem em um nicho específico de mercado. Com relação aos sistemas de processo judicial eletrônico, em especial o sistema PROJUDI, possuem uma complexa rede de interligações entre sistemas, atores, elementos técnicos, transacionais e sociais [20].

ECOS tem se tornado um tema de pesquisa relevante para a engenharia de software, principalmente no aspecto de estudo da evolução desses softwares. Assim, há grandes desafios também para a Engenharia de Requisitos, buscando alcançar concordância e compreensão acerca dos requisitos para sua evolução. Pesquisas tem sido realizadas no sentido de encontrar uma abordagem mais holística no tratamento dos requisitos dos usuários [21].

---

[1] Projudi – um sistema de informática que reproduz todo o procedimento judicial em meio eletrônico, substituindo o registro dos atos processuais realizados no papel por armazenamento e manipulação dos autos em meio digital [40].

[2] PJe - um sistema de processo judicial eletrônico capaz de permitir a prática de atos processuais, assim como o acompanhamento desse processo judicial, independentemente de o processo tramitar na Justiça Federal, na Justiça dos Estados, na Justiça Militar dos Estados e na Justiça do Trabalho [41].

4 Ambiguidades em requisitos evolutivos do Domínio Judicial

A forma como o direito é escrito pode criar uma barreira impenetrável para sua compreensão, até mesmo para profissionais do direito. Para cidadãos comuns, os textos legais até parecem escritos em língua estrangeira [22]. Pesquisas tem sido realizadas no sentido de nivelar o nível de linguagem entre as diversas camadas de pessoas envolvidas na dinâmica do direito [23].

A Hermenêutica Jurídica é responsável pelo estudo, definição e sistematização de métodos para determinar o sentido das expressões contidas nas normas jurídicas. A interpretação jurídica é a aplicação prática da hermenêutica, sendo gerida pelas suas descobertas e princípios. A interpretação de leis se torna mais completa quando coloca em crise o modelo representacional da linguagem, passando a considerar outros elementos além dos linguísticos em seus textos [24].

A principal mudança recente no direito é a perspectiva do chamado "destinatário-interprete", onde o receptor da mensagem passa a ter participação ativa no processo comunicativo [25]. Essa "viragem ontológica" tem sido a responsável pela construção de um novo paradigma filosófico no direito, sendo consequência da inevitável participação de elementos alheios a ciência pura do direito na interpretação de textos legais [26]. Esse novo paradigma coloca em confronto a até então consolidada, Teoria Pura do Direito[3], proposta por Hans Kelsen, responsável por grande parte do que hoje se entende por Direito Contemporâneo.

Neste novo paradigma, não é possível desvincular o direito de outros elementos em seu entorno. Concomitante a isso, a sociolinguística interacional, estabelece que quando o contexto não é observado, pode ser criada uma indeterminação. Apesar da existência de significados sociais pré-estabelecidos, a situação também cria conhecimento. As ambivalências e paradoxos são inerentes a língua como instrumento simbólico, de forma que o contexto é utilizado para solucionar as indeterminações [27].

As ambiguidades são um fenômeno cotidiano no direito. Entende-se como ambíguo um texto que comporta muitos sentidos, produzindo dúvida a respeito de uma norma ou texto legal. A ambiguidade é inerente a linguagem, porém, é possível identificar textos legais com graves ambiguidades, promovendo uma acentuada instabilidade e imprevisibilidade [25]. Em consequência disso, as ambiguidades são refletidas em todos os processos de comunicação do direito.

Diante de toda a complexidade inerente a linguagem do direito, de uma maneira especial pela ocorrências de ambiguidades, o processo de requisitos tem sido realizado de forma *ad hoc* [9], sendo os requisitos tratados e realizados em linguagem natural [8]. Há abordagens de métodos e ferramentas, que foram introduzidas nos últimos anos, contudo, tais se concentram em aspectos específicos, resolvendo apenas parcialmente os problemas, não se demonstrando adequados para a elicitação de requisitos legais [28].

---

[3] Para a Teoria Pura do Direito [42] os conceitos devem estar baseados exclusivamente no conteúdo das normas jurídicas positivas, devendo estar isentas de motivações, intensões, desejos e interesses das autoridades legisladoras.

Um sistema de software de sucesso deve ser capaz de evoluir com os requisitos dos *stakeholders* e ambientes onde o sistema opera [29]. O sistema de software deve se adaptar ao ambiente social onde os usuários formam suas opiniões e experiências [30]. Desta forma, uma sistemática eficaz para captar requisitos evolutivos é fundamental para o software se adaptar e melhorar [31].

Segundo a Norma ISO/IEC/IEEE 29148:2011, uma especificação de requisitos deve ser completa e inequívoca [32], auxiliando no diálogo e acordo entre os *stakeholders* acerca do que o software deve fazer. Entre várias recomendações, a norma defini que a especificação dos requisitos deve ser "não ambígua", permitindo apenas uma única interpretação. Assim, não se deve utilizar uma linguagem e/ou recursos que sejam técnicos demais que prejudiquem a compreensão dos *stakeholders*, nem muito simples e não estruturados que prejudiquem uma correta modelagem do sistema a ser desenvolvido.

A maioria dos defeitos de software são originados num processo de requisitos deficiente, e mais difíceis de reparar [4–6]. Estima-se que o processo de requisitos utiliza de 5 a 16% dos esforços do processo de desenvolvimento de software, porém o retrabalho para correção de erros utiliza cerca de 40 a 50%, o que indica se concentrados maiores esforços durante o processo de requisitos, levará a um menor retrabalho [33].

Os sistemas de processo judicial eletrônico encontram-se em processo de evolução, recebendo requisitos de diferentes usuários, com diferentes expectativas e objetivos, podendo ocorrer requisitos com conflitos de interpretação (ambiguidades). Se tais não forem tratados adequadamente, poderão comprometer a qualidade da evolução do sistema.

5 Tratando ambiguidades do Domínio Judicial usando Mapeamento Conceitual

Um modelo conceitual das leis e textos legais pode atuar como um orientador para a representação dos requisitos [10]. Assim, considera-se o uso de mapeamento conceitual para delimitar o contexto no qual o requisito está inserido, permitindo identificar os conceitos relacionados com o mesmo, representando adequadamente as necessidades dos usuários. Os mapas conceituais propostos por Novak e Gowin [11, 13] estabelecem uma estruturação hierárquica dos conceitos que são apresentados numa diferenciação progressiva e reconciliação integrativa, estruturados conforme a Teoria da Aprendizagem Significativa proposta por David Ausubel [34, 35], contribuindo de maneira eficiente para a construção do conhecimento [36].

Tal proposta é uma importante ferramenta disponibilizada pela Engenharia e Gestão do Conhecimento [37], sendo aplicada em áreas diversas, como melhoria de processos de software [38], e também em para o tratamento de requisitos de sistema do domínio jurídico [39].

Tendo em vista que os Mapas Conceituais facilitam a aprendizagem criando quadros de conhecimento, que permitem não somente que o conhecimento seja utilizado em novos contextos, mas também permite sua retenção [12]. Tal característica levou a considerar a hipótese de seu uso como ferramenta de apoio a um modelo de comunicação de requisitos para sistemas do domínio jurídico existentes.

Assim, propõe-se um modelo de captação de requisitos, tendo como meta o auxílio na interpretação das solicitações/requisitos para sistemas existentes do domínio jurídico. Adota-se o mapeamento conceitual como auxílio no nível de comunicação, produzindo um modelo de comunicação onde os requisitos produzidos por usuários e especialistas do domínio jurídico dentro de um contexto conceitual. Ao incluir o mapeamento conceitual, espera-se reduzir o número de ambiguidades e falhas na interpretação e compreensão dos requisitos, quando se comparado aos requisitos produzidos de maneira textual.

O modelo proposto oferece ao usuário uma opção para que o mesmo informe uma nova solicitação durante o uso do sistema. Essa nova solicitação será realizada dentro de um mapeamento conceitual apresentado pelo sistema, relacionado a funcionalidade do sistema que está sendo utilizada no momento. Essa requisição deve ser realizada de modo textual, sem a necessidade de o requisitante "construir" o mapeamento conceitual. Ao terminar de descrever sua solicitação, a mesma será encaminhada para a equipe de desenvolvimento para análise. Esse processo é realizado conforme disposto pelo diagrama apresentado na figura 1.

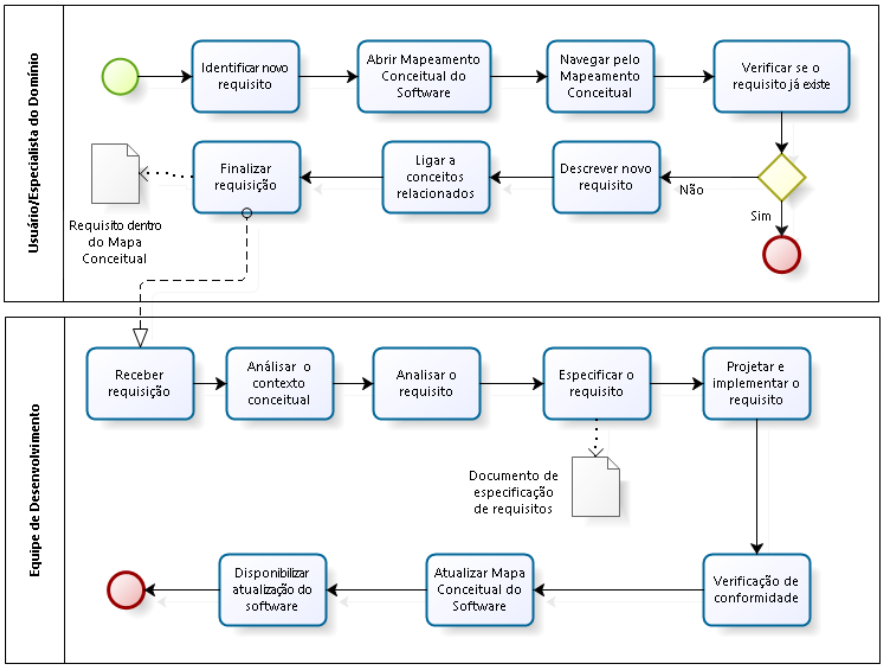

Fig. 1. Processo de criação de requisitos com mapeamento conceitual (Fonte: autoria própria)

Com esse procedimento, a equipe de desenvolvimento receberá o requisito dentro de um contexto conceitual. O artefato produzido é ilustrado pela figura 2. O mapeamento conceitual permitirá uma correta contextualização na qual o requisito foi inserido, permitindo uma melhor compreensão do mesmo.

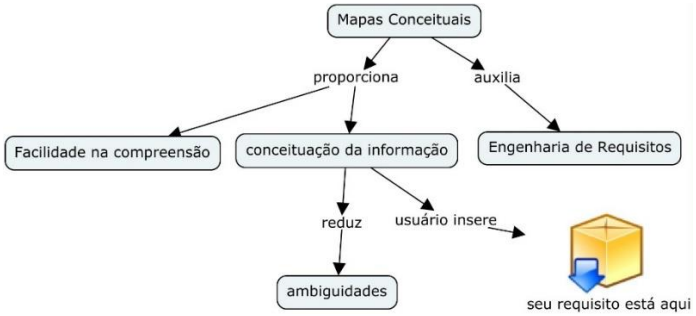

Fig. 2. Ilustração de requisito inserido dentro de mapeamento conceitual (fonte: autoria própria)

5.1 Proposta de Arquitetura de Sistema

Propõe-se um modelo de arquitetura a ser utilizada nos sistemas a seguirem a metodologia, que seja de fácil integração, tanto para desenvolvimento de novos sistemas, como para seu uso em sistemas existentes. Esse modelo de arquitetura baseia-se em um conceito modular, de forma que haverá um módulo responsável pela geração e gerenciamento dos mapas conceituais do sistema. Essa estrutura pode ser melhor visualizada pela figura 3.

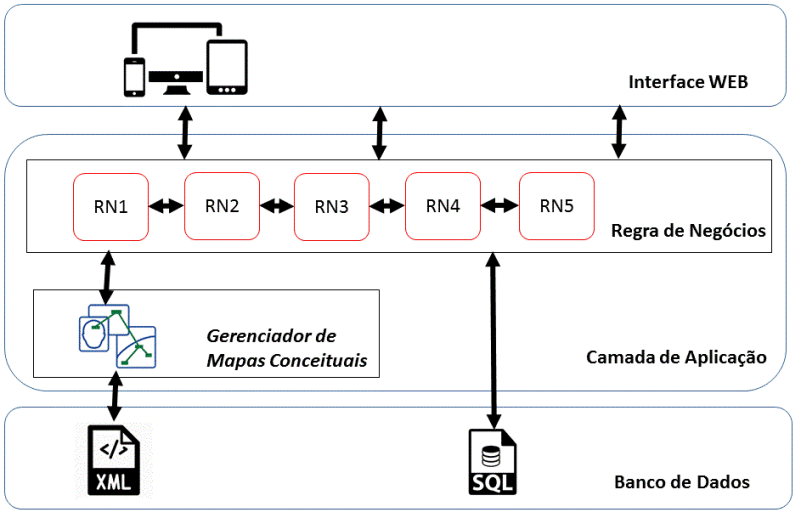

Fig. 3. Arquitetura de sistema em 3 camadas com gerenciador de mapas conceituais

A arquitetura apresentada pela figura acima apresenta a disposição de um sistema organizado em 3 camadas modulares. A primeira camada “Interface WEB” consiste na interface gráfica do usuário, disponível em HTML ou XHTML, portável para dispositivos *desktop*, bem como para dispositivos móveis como *tablets* e *smartphones*. Essa camada possui uma comunicação bilateral com uma segunda camada, a qual representa

o sistema propriamente dito. Essa "Camada de Aplicação" possui um conjunto de módulos que implementam as regras de negócio do sistema, que por sua vez possui uma comunicação bilateral com uma terceira camada, contendo um banco de dados SQL. Ainda dentro de "Camada de Aplicação", possuímos um *framework* identificado como "Gerenciador de Mapas Conceituais", que possui uma comunicação bilateral com o conjunto de módulos de regras de negócio, bem como uma comunicação bilateral com uma base de dados XML.

O *framework* "Gerenciador de Mapas Conceituais" será acionado pelos módulos de regras de negócio no momento em que o usuário decidir fazer uma solicitação de sistema. Ao ser acionado, o *framework* irá identificar quais são os conceitos pertinentes as funcionalidades do sistema que estão sendo utilizadas no momento da solicitação, e irá realizar uma consulta a base de dados em XML acerca de tais conceitos, produzindo dinamicamente na sequência, um mapa conceitual para ser apresentado ao usuário, o qual irá realizar sua solicitação na forma descrita pela subseção anterior. Tais passos são apresentados no diagrama de atividades a seguir.

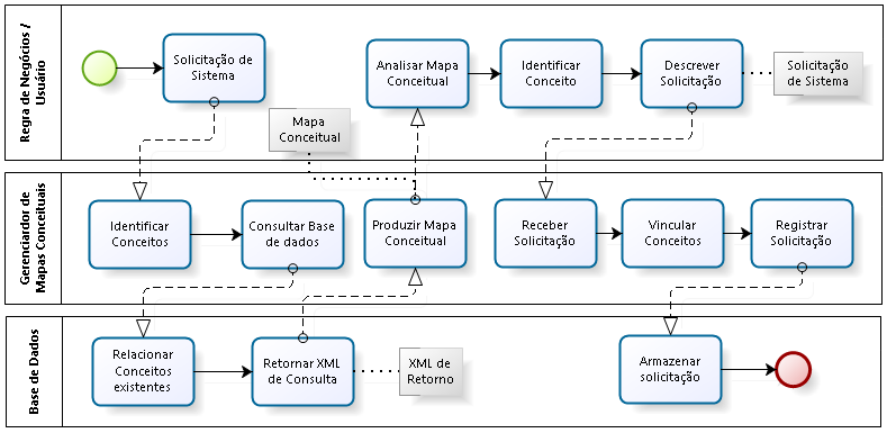

Fig. 4. Processamento de uma solicitação de sistema usando o "Gerenciador de Mapas Conceituais"

A base de dados em XML, com os conceitos do sistema, será alimentada sempre por uma analista de negócios e/ou analista de requisitos em momentos oportunos. Quando da criação de um novo sistema ou módulo, poderá ser alimentada a base de mapas conceituais durante o desenvolvimento do mesmo ou conforme o cronograma de implantação. Quando da integração em um sistema existente, essa alimentação poderá ser realizada gradualmente, sem prejuízos ao sistema, pois não é determinante que todos os conceitos estejam alimentados para que uma solicitação seja realizada.

Sempre que atendida uma solicitação, a base de dados em XML deverá ser examinada e atualizada, se for o caso, garantindo a consistência das funcionalidades do sistema com os conceitos inseridos na base de dados.

5.2 Exemplo prático do modelo proposto

Para uma melhor ilustração, considera-se o artigo 185[4] do CPC (Código de Processo Civil). Um advogado, utilizando o sistema de processo judicial eletrônico, possui uma relação de processos nos quais há intimações para serem verificadas, tal como apresentado pela figura 4. Cada intimação possui um prazo específico para cumprimento. As intimações após lidas, ficaram em uma lista de pendências para cumprimento. Porém, o advogado nota que uma determinada intimação recebida não possui prazo (ou realizada com prazo "zero"), e que assim, após abrir a intimação, a mesma não fica na lista de pendências do sistema, podendo assim cair em esquecimento. Considerando o disposto no artigo 185 do CPC, o advogado chega à conclusão que uma intimação deve necessariamente ter um prazo. Esse advogado decide realizar uma solicitação, na opção oferecida pelo sistema, conforme demonstrado pelo protótipo funcional apresentado na figura 5.

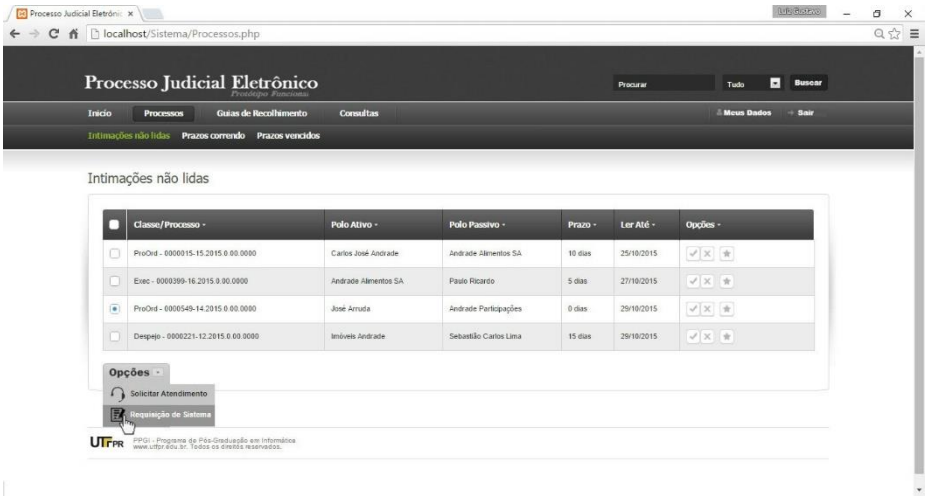

Fig. 5. Protótipo Funcional de um sistema de Processo Judicial Eletrônico – Realizar solicitação dentro da tela de Intimações não lidas

Ao advogado, quando requisitar uma solicitação de sistema, uma nova tela lhe é apresentada, contendo um mapa conceitual com os conceitos legais e normativos relacionados com a funcionalidade do sistema; conceitos de intimação no presente caso que foram observados quando do desenvolvimento do sistema até o presente momento. Assim, o advogado poderá examinar todos os conceitos, visualizando pelo mapa conceitual. Um exemplo de mapa conceitual para a funcionalidades de intimação é apresentado pela figura 6.

---

[4] Artigo 185 do CPC – "*Não havendo preceito legal nem assinação pelo juiz, será de 5 (cinco) dias o prazo para a prática de ato processual a cargo da parte".*

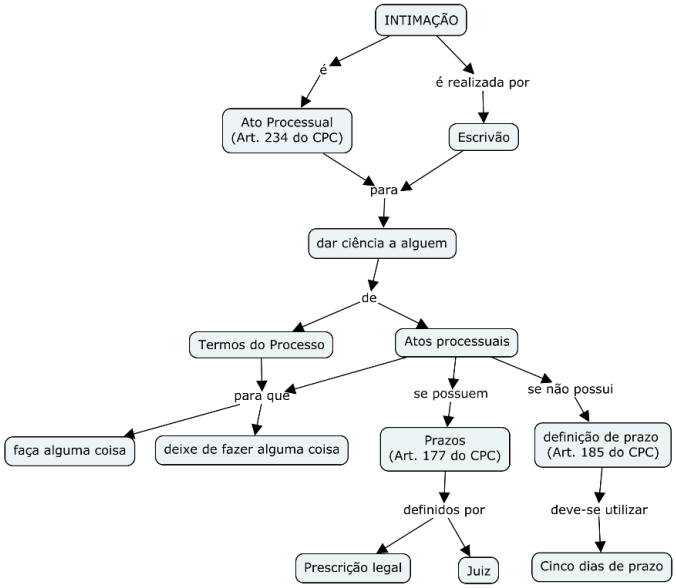

Fig. 6. Mapa Conceitual com definições e prescrições legais a respeito de "intimações"

Ao terminar de examinar o mapa conceitual, e identificado o conceito mais próximo da sua necessidade, o advogado seleciona o conceito. Neste momento o sistema abre uma caixa de texto para que o advogado apresente as informações que julgar necessárias. Ao final desse procedimento, um artefato é gerado pelo sistema e enviado aos analistas de requisitos do sistema. Tal artefato é ilustrado pela figura 7.

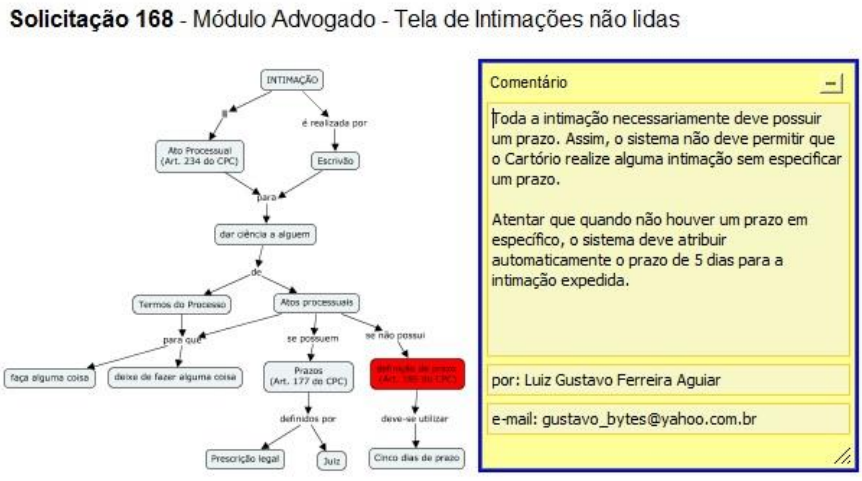

Fig. 7. Artefato de solicitação de sistema utilizando mapeamento conceitual

O artefato constante da figura 7 proporciona uma melhor compreensão conceitual da solicitação do usuário do sistema, permitindo ao analista de requisitos iniciar a leitura da solicitação contextualizado acerca do assunto referente a solicitação, reduzindo a margem de interpretações equivocadas, pois, mesmo que o usuário em sua descrição informações um tanto quanto desconexas, ou imprecisas, tendo em vista o mapeamento conceitual, as arestas são reduzidas pelo mapa, que reduz e especifica melhor o escopo da solicitação.

6 Considerações e trabalhos futuros

Os sistemas da área judicial encontram-se em um momento de evolução, em especial os sistemas de processo judicial eletrônico. A estrutura organizacional do Poder Judiciário Brasileiro apresenta diversas instituições autônomas com culturas e necessidades especificas. Para que os softwares possam atender bem tal estrutura, é fundamental a participação dos usuários, colaborando para a evolução dos softwares com seus conhecimentos e experiências. Processos de requisitos de forma *ad hoc* tem sido realizados, privilegiando o uso da linguagem natural para a especificação dos requisitos. Tal perspectiva, concomitante com as especificidades da linguagem do direito, que permitem uma variedade de interpretações, prejudica a especificação de requisitos, gerando ambiguidades.

Este artigo apresentou um modelo para a captação e tratamento de ambiguidades para requisito de evolução de sistemas jurídicos, tendo como fonte o *feedback* dos usuários. Considerou-se, como solução, o uso de Mapeamento Conceitual tendo em vista sua característica cognitiva que permite transmitir, de modo simples, o escopo no qual uma solicitação é realizada. Uma vez que, delimitado o escopo de uma solicitação, e apresentado conceitos relacionados a tal solicitação, são reduzidas as possibilidades de variadas interpretações, reduzindo assim as ambiguidades.

Os Mapas Conceituais conseguem expressar adequadamente os conceitos da área jurídica para a equipe de desenvolvimento [39], viabilizando assim o desenvolvimento do modelo apresentado.

Como trabalhos futuros pretende-se desenvolver um estudo experimental utilizando um protótipo funcional de um processo judicial eletrônico que implemente o modelo apresentado neste artigo. O objetivo do estudo será observar o ciclo de solicitações utilizando o mapeamento conceitual comparando os resultados das solicitações realizadas em linguagem natural *versus* o modelo apresentado neste trabalho.

Por fim, um *framework*, baseado na metodologia do modelo, poderá ser desenvolvido e disponibilizado para a comunidade para a implementação do método em sistemas já existentes e em sistemas ainda a serem desenvolvidos.

Referências

1. Rajlich, Václav. 2014. Software evolution and maintenance. In *Proceedings of the on Future of Software Engineering - FOSE 2014*, 133–144. New York, New York, USA, New York, USA: ACM Press. doi:10.1145/2593882.2593893.

2. Belady, L. a., and M. M. Lehman. 1976. A model of large program development. *IBM Systems Journal* 15: 225–252. doi:10.1147/sj.153.0225.

3. Carreno, Laura V Galvis, and Kristina Winbladh. 2013. Analysis of user comments: An approach for software requirements evolution. *Proceedings - International Conference on Software Engineering*: 582–591. doi:10.1109/ICSE.2013.6606604.

4. Bourque, Pierre, and Richard E. (Dick) Fairley. 2014. *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)*. Edited by IEEE Computer Society. IEEE Computer Society.

5. Braude, E J, and M E Bernstein. 2010. *Software Engineering: Modern Approaches*. Wiley.

6. Brooks, Frederick P. 1995. No Silver Bullet — Essence and Accident in Software Engineering: 1–16.

7. Ghanavati, Sepideh, and Travis D Breaux. 2015. Comparing and Analyzing Definitions in Multi- jurisdictions. In *Requirements Engineering and Law (RELAW), 2015 IEEE Eighth International Workshop on*, 47–56. Ottawa. doi:10.1109/RELAW.2015.7330211.

8. Hassan, Wa ël, and Luigi Logrippo. 2008. Requirements and compliance in legal systems: a logic approach. *2008 Requirements Engineering and Law*: 40–44. doi:10.1109/RELAW.2008.8.

9. Breaux, Travis D., Matthew W. Vail, and Annie I. Antón. 2006. Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations. *Proceedings of the IEEE International Conference on Requirements Engineering*: 46–55. doi:10.1109/RE.2006.68.

10. Siena, Alberto, John Mylopoulos, Anna Perini, and Angelo Susi. 2008. From laws to requirements. *Requirements Engineering and Law, RELAW'08*: 6–10. doi:10.1109/RELAW.2008.6.

11. Novak, Joseph D, and Alberto J Cañas. 2006. The Origins of the Concept Mapping Tool and the Continuing Evolution of the Tool. *Information Visualization Journal* 5: 175–184.

12. Novak, Joseph D, and Alberto J Cañas. 2008. *The Theory Underlying Concept Maps and How to Construct and Use Them*. Flórida.

13. Novak, J D, and D B Gowin. 1999. *Aprender a aprender (2 ed.)*. Coleção Plátano Universitária. Plátano Edições Técnicas.

14. Rajlich, V.T. 2001. Software evolution: a road map. In *Proceedings IEEE International Conference on Software Maintenance. ICSM 2001*, 6. Florence: IEEE Comput. Soc. doi:10.1109/ICSM.2001.972705.

15. Pagano, Dennis, and Bernd Bruegge. 2013. User involvement in software evolution practice: A case study. In *2013 35th International Conference on Software Engineering*

(ICSE), 953–962. San Francisco, CA: IEEE. doi:10.1109/ICSE.2013.6606645.

16. Ko, Andrew J., Michael J. Lee, Valentina Ferrari, Steven Ip, and Charlie Tran. 2011. A case study of post-deployment user feedback triage. In *Proceeding of the 4th international workshop on Cooperative and human aspects of software engineering - CHASE '11*, 1. New York, New York, USA: ACM Press. doi:10.1145/1984642.1984644.

17. Maalej, Walid, and Dennis Pagano. 2011. On the Socialness of Software. In *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, 864–871. IEEE. doi:10.1109/DASC.2011.146.

18. Andrade, André, and Luiz Antonio Joia. 2012. Organizational structure and ICT strategies in the Brazilian Judiciary System. *Government Information Quarterly* 29, Supple: S32 – S42. doi:http://dx.doi.org/10.1016/j.giq.2011.08.003.

19. Andrade, Andre. 2008. Porque a Justiça não se comunica ? Um problema de estrutura organizacional. *Revista de Derecho nformático/Computer Law Magazine*.

20. Silva, Rebeca Teodoro da, Luiz Gustavo Ferreira Aguiar, and Elias Canhadas Genvigir. 2015. Ecossistema de Software no Contexto do Poder Judiciário - Apontamentos Sobre o ECOS Projudi no Estado do Paraná. In *9th WORKSHOP ON DISTRIBUTED SOFTWARE DEVELOPMENT, SOFTWARE ECOSYSTEMS AND SYSTEMS-OF-SYSTEMS*, 49 – 56. Belo Horizonte – MG, Brazil.

21. Valenca, George. 2013. Requirements negotiation model: A social oriented approach for software ecosystems evolution. In *2013 21st IEEE International Requirements Engineering Conference (RE)*, 393–396. IEEE. doi:10.1109/RE.2013.6636763.

22. Maley, Yon. 1987. The language of legislation. *Language in Society* 16: 25. doi:10.1017/S0047404500012112.

23. BAGNO, Marcos. 2006. *A norma oculta : língua e poder na sociedade brasileira*. 2ª Edição. São Paulo: Parábola.

24. Streck, Lenio Luiz. 1999. Hermeneutica Juridica e(m) crise.

25. SGARBI, ADRIAN. 2007. *TEORIA DO DIREITO PRIMEIRAS LIÇÕES*. Edited by LUMEN JURIS2. 1 Ed. Rio de Janeiro: LUMEN JURIS2.

26. SOUSA, Aline Delias. 2004. Direito e linguagem: a contribuição do neopositivismo lógico e da filosofia da linguagem ordinária para um direito transformador. *Revista Escrita Direito*.

27. MARTINS, Carla. 2002. A INDETERMINAÇÃO DO SIGNIFICADO NOS ESTUDOS SÓCIO-PRAGMÁTICOS: DIVERGÊNCIAS TEÓRICO-METODOLÓGICAS. *Delta*. doi:http://dx.doi.org/10.1590/S0102-44502002000100004.

28. Kiyavitskaya, Nadzeya, Alžběta Krausová, and Nicola Zannone. 2008. Why eliciting and managing legal requirements is hard. *Requirements Engineering and Law*: 26–30. doi:10.1109/RELAW.2008.10.

29. Nuseibeh, Bashar, and Steve Easterbrook. 2000. Requirements Engineering: A Roadmap. In *Proceedings of the conference on The future of Software engineering - ICSE '00*, 1:35–46. New York, New York, USA: ACM Press. doi:10.1145/336512.336523.

30. Godfrey, Michael W, and Daniel M German. 2008. The past, present, and future of software evolution. In *2008 Frontiers of Software Maintenance*, 129–138. IEEE. doi:10.1109/FOSM.2008.4659256.

31. Jiang, Wei, Haibin Ruan, Li Zhang, Philip Lew, and Jing Jiang. 2014. For User-Driven Software Evolution: Requirements Elicitation Derived from Mining Online Reviews. In , ed. Vincent S. Tseng, Tu Bao Ho, Zhi-Hua Zhou, Arbee L. P. Chen, and Hung-Yu Kao, 8444:584–595. Lecture Notes in Computer Science. Cham: Springer International Publishing. doi:10.1007/978-3-319-06605-9_48.

32. Standard, International. 2011. Systems and software engineering -- Life cycle processes --Requirements engineering. *ISO/IEC/IEEE 29148:2011(E)* 2011: 1–94. doi:10.1109/IEEESTD.2011.6146379.

33. Génova, Gonzalo, José M. Fuentes, Juan Llorens, Omar Hurtado, and Valentín Moreno. 2011. A framework to measure and improve the quality of textual requirements. *Requirements Engineering* 18: 25–41. doi:10.1007/s00766-011-0134-z.

34. Ausubel, David Paul, Joseph D Novak, and Helen Hanesian. 1980. *Psicologia educacional*. Interamericana.

35. Ausubel, David P. 2003. Aquisição e retenção de conhecimentos: uma perspectiva cognitiva. *Lisboa: Plátano* 1.

36. Tavares, Romero. 2007. Construindo mapas conceituais. *Ciências & Cognição*.

37. Fabri, José Augusto, Alessandro Silveira Duarte, Alexandre L'Erario, Rodrigo H. Cunha Palácios, Elias Canhadas Genvigir, and André Luís dos Santos Domingues. 2012. Aplicación de los mapas conceptuales en la definición y la institucionalización de los procesos para la producción de software. *Fifth Int. Conference on Concept Mapping*: 25 – 32.

38. Fabri, José Augusto, Alexandre L Erario, André Luis, and Marcelo S De Paula Pessôa. 2011. Knowledge Management and Concept Maps applied to Software Process Improvement. *6th Iberian Conference on Information Systems and Technologies (CISTI)*: 1 – 4.

39. Aguiar, Luiz Gustavo Ferreira, Elias Canhadas Genvigir, José Augusto Fabri, and Alexandre L´Erario. 2014. APLICAÇÃO DE MAPAS CONCEITUAIS PARA REGISTRO DE REQUISITOS DE SOFTWARE NO AMBIENTE JURÍDICO. In *Sixth International Conference on Concept Mapping*. Santos: Institute for Human & Machine Cognition.

40. Justiça, Conselho Nacional de. 2015. Sistema CNJ - Projudi.

41. Justiça, Conselho Nacinal de. 2015. Processo Judicial Eletrônico (PJe).

42. Kelsen, Hans. 2009. *Teoria Pura do Direito*. Edited by Wmf Martins Fontes. 8ª Edição. São Paulo: Wmf Martins Fontes.