Minerando Conhecimentos de Projetos de *Software* a partir dos Registros de Comunicação de Desenvolvedores

Márcia Lima[1], **Tayana Conte**[1], **Igor Steinmacher**[2], **Bruno Gadelha**[1]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Manaus – AM – Brasil

[2]Northern Arizona University (NAU)
Arizona – USA

{marcia.lima,tayana}@icomp.ufam.edu.br, igor.steinmacher@nau.edu

bruno@icomp.ufam.edu.br

Resumo. *Desenvolvedores de software utilizam diversos canais de comunicação para apoiar o desenvolvimento e gestão de projetos. No entanto, discussões relevantes podem se perder, ser esquecidas ou duplicadas em meio ao grande volume de mensagens, comprometendo o compartilhamento e reuso de conhecimentos. Esta pesquisa investiga como usar métodos automáticos para identificar discussões relevantes nos registros de conversa de desenvolvedores e apoiar a identificação de conhecimentos de projetos de software. Com base na metodologia Design Science Research, foi criado o framework experimental Miner4DevTeam, que viabiliza a identificação de conhecimentos úteis para a evolução do produto e tomada de decisões. Futuramente, serão desenvolvidas estratégias para a gestão de conhecimento de projetos.*

1. Introdução

Equipes de desenvolvimento de software recorrem a diferentes canais de comunicação para dar suporte às tarefas de desenvolvimento e gerenciamento de projetos [Storey et al. 2016]. Dentre tais canais destacam-se *e-mail*, *chat* e fóruns [Storey et al. 2016]. Pesquisas relatam que as mensagens de *chats* substituíram os *e-mails* em muitas equipes [Alkadhi et al. 2018]. Em comunidades de software livre também observa-se uma mudança de comportamento com respeito a utilização das ferramentas de comunicação. Nos últimos anos, fóruns do tipo Q&A, voltados para desenvolvedores de software (*Programming Community-based Question Answering (PCQA)*) têm atraído a atenção dos membros das equipes de software [Pei et al. 2021].

Tanto os *chats* como os fóruns de comunicação constituem valiosas fontes de informação, pois podem conter registros de discussões relacionadas ao processo de desenvolvimento dos projetos de software. Contudo, quando equipes usam tais canais, discussões relevantes podem tornar-se "perdidas", não implementadas, esquecidas, duplicadas ou difíceis de serem encontradas em meio ao excesso de mensagens trocadas. Em todos os cenários destacados, a perda e a duplicação de informações podem comprometer o compartilhamento e o reuso de conhecimentos dos projetos de software.

Motivada pelo contexto mencionado, o objetivo desta pesquisa é **definir mecanismos para apoiar a mineração de conhecimentos de projetos de software a partir dos**

registros não estruturados de comunicação de desenvolvedores, gerados através do uso de ferramentas de comunicação síncronas (*chats*) e assíncronas (fórum do tipo PCQA). A seguinte questão de pesquisa (QP) guiou o desenvolvimento desta pesquisa:

QP:Como utilizar métodos automáticos para identificar discussões relevantes de projetos de software a partir dos registros de comunicação de desenvolvedores?

Baseando-se na metodologia *Design Science Research* (DSR) [Hevner 2007] foi desenvolvido e avaliado o *framework* experimental *Miner4DevTeam*. O desenvolvimento do *Miner4DevTeam* baseou-se em técnicas e algoritmos de Recuperação de Informação (RI), Processamento de Linguagem Natural (PLN) e *Deep Learning* (DL). O *framework* dá suporte a determinação do:

- ***Project Lost Knowledge* (PLK):** refere-se a decisões e discussões de projetos que não foram registradas formalmente na documentação oficial ou que foram esquecidas pelas equipes, tornando-se "perdidas". O PLK é importante para gerentes e líderes de equipe, pois pode promover a qualidade do software, apoiar a evolução dos produtos e auxiliar na tomada de decisões estratégicas em empresas de software [Lima et al. 2019a].
- ***Project Frequent Knowledge* (PFK):** abrange decisões de desenvolvimento e gestão que já foram amplamente discutidas pela equipe e se tornaram claras para o time, mas que podem não estar formalmente registradas na documentação oficial. O PFK é importante para facilitar a integração de novos membros, pois esclarece o escopo de desenvolvimento do software [Lima et al. 2020].
- ***Project Related Knowledge* (PRK):** refere-se a postagens de discussão duplicadas ou quase duplicadas, que abordam tópicos semelhantes. Detectar essas postagens é importante para reduzir a duplicação de informações e disseminar o conhecimento do projeto. Na tese, essas postagens são chamadas de postagens relacionadas, e seu público-alvo são mantenedores e líderes de projetos [Lima et al. 2023].

Os resultados obtidos mostram que os conhecimentos identificados (PLK, PFK e PRK) podem apoiar a evolução dos produtos desenvolvidos, auxiliar o processo de tomada de decisões estratégicas das empresas e promover o compartilhamento e a reutilização do conhecimento dos projetos.

2. Referencial Teórico

No contexto da mineração de repositórios que registram discussões de equipes de desenvolvimento, pesquisadores têm abordado o problema da duplicação de informação em sistemas de rastreamento de *bugs*, fóruns do tipo PCQA e plataformas colaborativas como o GitHub [Pei et al. 2021]. Pesquisadores usaram DL para identificar postagens duplicadas no *Stack Overflow* [Wang et al. 2020]. Também foram propostas técnicas tradicionais de RI e PLN para detectar duplicadas em *Pull Requests* no GitHub [Ren et al. 2019]. Estratégias foram desenvolvidas para detectar *issues* relacionadas no GitHub [Zhang et al. 2020]. Na mineração de conhecimento a partir de *e-mails* e ferramentas de bate-papo, pesquisadores propuseram o sistema KTR para recuperar conhecimentos sobre soluções de problemas de software [Francois et al. 2015]. Alkadhi et al (2018) examinaram a extração automática de *design rationale* a partir de mensagens de bate-papo em projetos *open source*.

3. *Framework* Miner4DevTeam

A mineração de decisões e discussões relevantes de projetos de software viabiliza a disseminação e o reuso de conhecimentos acerca dos projetos. Gerentes ou líderes de equipes podem se beneficiar com tais informações para (1) apoiar a gestão dos projetos, (2) recuperar informações técnicas já discutidas, (3) controlar a qualidade do produto, (4) delegar tarefas e (5) acompanhar o desenvolvimento dos projetos. Membros das equipes podem se beneficiar com tais informações para (1) conhecer o projeto e (2) compreender o raciocínio acerca de decisões já tomadas. Para tanto, foi desenvolvido o *framework Miner4DevTeam* (Figura 1). Os dados gerados pelos *framework* são apresentados aos membros das equipes que transformam tais dados em conhecimentos sobre os projetos.

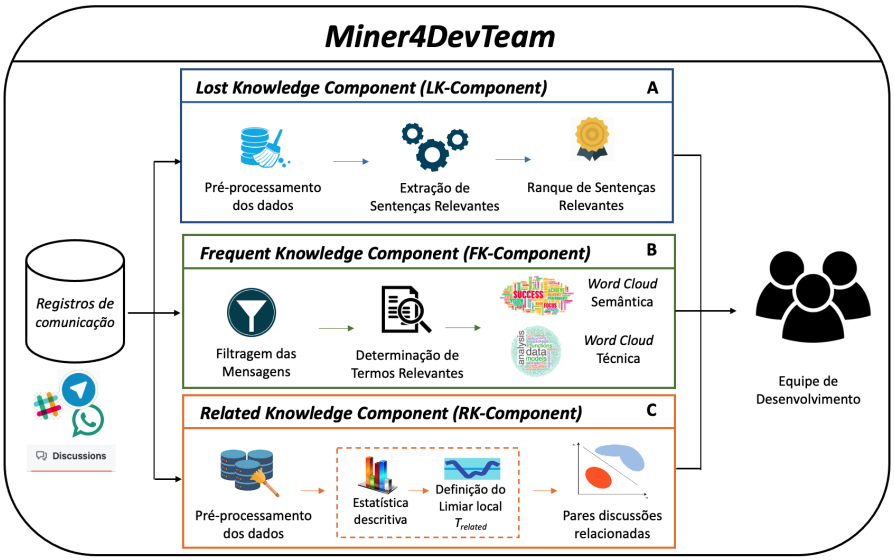

Figura 1. *Framework Miner4DevTeam*

A geração de cada componente corresponde a um ciclo de *design*, da metodologia DSR adotada. Os componentes são descritos a seguir:

LK-Component: visa identificar e extrair mensagens relevantes dos registros de comunicação de equipes de desenvolvimento, que usam *chats*, para determinar o PLK. Para tanto são executas três etapas principais: *1. Pré-processamento de dados*: onde são feitos o tratamento e filtragem dos dados. *2. Extração de sentenças relevantes:* etapa em que é feita a identificação de sentenças relevantes para determinar o PLK usando técnicas de sumarização de texto e métrica TF-IDF.*3. Ranque de sentenças relevantes:* determinação das N sentenças mais relevantes, sendo N configurável.

Para avaliar o *LK-Component* foram realizados estudos em dois contextos distintos: *1. Indústria:* foram analisados os registros de comunicação de duas equipes de desenvolvimento atuantes em duas *startups* de Manaus [Lima et al. 2019a]. *2. Academia:* foram analisados os registros de comunicação de três equipes de projetos de um curso interdisciplinar para avaliar a utilidade dos *logs* no acompanhamento do desenvolvimento de projetos de software e progresso dos alunos [Lima et al. 2019b].

Os estudos realizados para validar o *LK-Component* mostram que ele é eficaz no apoio a recuperação do conhecimentos de projetos perdidos (PLK), facilitando a gestão e a continuidade dos projetos.

FK-Component: desenvolvido para capturar o PFK dos *logs* de comunicação das equipes de desenvolvimento que usam *chats*. O *FK-Component* é composto por três etapas principais: *1. Filtragem de mensagens:* técnicas de PLN são usadas para dividir as mensagens em frases e algoritmos de distância de edição são utilizados para eliminar frases irrelevantes. *2. Determinação de termos relevantes:* nesta etapa as mensagens são transformadas em palavras, eliminando as *stopwords*. Técnicas de PLN, como *part-of-speech*, são utilizadas para classificar e filtrar as palavras. *3. Sumarização do PFK:* o PFK é representado por duas *word clouds*: (1) **semântica**, com termos sobre especificações, desenvolvimento e gerenciamento do projeto e (2) **técnica**, que inclui termos relacionados ao uso de tecnologias e decisões técnicas.

Os estudos realizados mostram que o *FK-Component* é útil para a integração de novos membros às equipes de desenvolvimento, bem como para auxiliar membros antigos a recordar decisões anteriores [Lima et al. 2020].

RK-Component: desenvolvido para identificar discussões relacionadas (PRK) em fóruns de desenvolvedores. O *RK-Component* executa três etapas principais: *1. Pré-processamento de dados:* nesta etapa os dados foram preparados para otimizar a identificação de discussões relacionadas. *2. Verificador de similaridade:* nesta etapa ocorre a identificação de pares de postagens candidatas a relacionadas. Foi utilizado o modelo de aprendizagem de máquina profunda `all-mpnet-base-v2`[1] para gerar *embeddings* das postagens e calcular a similaridade semântica entre elas usando a métrica de similaridade de cosseno. *3. Seleção dos pares de discussões relacionadas:* um limiar local é determinado para identificar discussões relacionadas com base na estatística descritiva dos valores de similaridade. Os pares com valor igual ou superior ao limiar são considerados relacionados.

Os estudos realizados para validar o *RK-Component* mostram que abordagem apoia a gestão do conhecimento ao identificar discussões relacionadas, promovendo a eficiência na colaboração e na disseminação de informações importantes em projetos de software. Além disso, os resultados obtidos apoiam a minimização dos problemas gerados pela duplicação de informação em projetos de software [Lima et al. 2023].

4. Conclusões

Respondendo a QP norteadora desta pesquisa tem-se que *os estudos realizados mostraram resultados positivos ao utilizar técnicas de mineração de dados, RI, PLN e DL para identificar e extrair discussões relevantes de projetos de software a partir de registros de comunicação. Essas discussões são vitais para a disseminação e reuso de conhecimentos entre equipes, abrangendo temas como análise, planejamento, desenvolvimento, testes, lições aprendidas e gerenciamento. O Miner4DevTeam foi criado para apoiar a gestão do conhecimento, com componentes que ajudam a identificar conhecimentos perdidos (LK-Component), facilitar a integração de novos membros (FK-Component) e reduzir inconsistências de informações duplicadas (RK-Component).*

Futuras pesquisas podem focar no aprimoramento dos componentes propostos, explorar diferentes repositórios de dados, e desenvolver estratégias para a gestão do conhecimento em projetos de software.

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Agradecimentos

CAPES - Financiamento 001. FAPEAM – POSGRAD e 062.00150/2020. CNPq nº 314797/2023-8 e 443934/2023-1. Samsung-UFAM de Ensino e Pesquisa (SUPER). Samsung Eletrônica da Amazônia Ltda., nos termos da Lei Federal n. 8.387/1991, através do convênio n 003/2019, firmado com o ICOMP/UFAM.

Referências

Alkadhi, R., Nonnenmacher, M., Guzman, E., and Bruegge, B. (2018). How do developers discuss rationale? In *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 357–369. IEEE.

Francois, R., Nada, M., and Hassan, A. (2015). How to extract knowledge from professional e-mails. In *2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 687–692. IEEE.

Hevner, A. R. (2007). A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4.

Lima, M., Ahmed, I., Conte, T., Nascimento, E., Oliveira, E., and Gadelha, B. (2019a). Land of lost knowledge: An initial investigation into projects lost knowledge. In *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–6. IEEE.

Lima, M., Fontão, A., Fernandes, D., Conte, T., and Gadelha, B. (2019b). How are my students going? a tool to analyse students' interactions on capstone courses. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 1611.

Lima, M., Oliveira, E., Conte, T., and Gadelha, B. (2020). Clouds are heavy! a storm of relevant project-related terms to support newcomers' onboarding. In *Proceedings of the 34th Brazilian Symposium on Software Engineering, SBES 2020, Natal, Brazil, October 19-23, 2020*, pages 319–324.

Lima, M., Steinmacher, I., Ford, D., Liu, E., Vorreuter, G., Conte, T., and Gadelha, B. (2023). Looking for related discussions on github discussions. *PeerJ Computer Science*.

Pei, J., Wu, Y., Qin, Z., Cong, Y., and Guan, J. (2021). Attention-based model for predicting question relatedness on stack overflow. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*, pages 97–107. IEEE.

Ren, L., Zhou, S., Kastner, C., and Wasowski, A. (2019). Identifying redundancies in fork-based development. In *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 230–241. IEEE.

Storey, M.-A., Zagalsky, A., Figueira Filho, F., Singer, L., and German, D. M. (2016). How social and communication channels shape and challenge a participatory culture in software development. *IEEE Transactions on Software Engineering*, 43(2):185–204.

Wang, L., Zhang, L., and Jiang, J. (2020). Duplicate question detection with deep learning in stack overflow. *IEEE Access*, 8:25964–25975.

Zhang, Y., Wu, Y., Wang, T., and Wang, H. (2020). ilinker: a novel approach for issue knowledge acquisition in github projects. *World Wide Web*, 23(3):1589–1619.