Mineração de Repositórios de Software para Internet das Coisas e Robótica (IoRT): Análise Inicial dos Forks

Mining Software Repositories for Internet of Things and Robotics (IoRT): Initial Analysis of Forks

Alexandre Juarez Corneau[1], Michel Albonico [2]

RESUMO

Vivemos em um mundo onde os sistemas ubíquos, também chamados de pervasivos, são cada vez mais comuns e dependem fortemente de software altamente distribuído, desafiadores de projetar e manter. Por outro lado, temos acesso a uma abundância de repositórios de código aberto no GitHub, que podem servir como base para simplificar o desenvolvimento de novos projetos. Este artigo descreve a metodologia utilizada para coletar dados de repositórios desse tipo de sistema, destacando as ferramentas usadas e apresenta resultados quantitativos preliminares. O processo de coleta envolveu quatro etapas: identificação dos desenvolvedores que fizeram *fork* dos repositórios selecionados, listagem dos repositórios de cada desenvolvedor, extração de informações de cada repositório e tabulação dos dados para análises futuras.
PALAVRAS-CHAVE: forks; repositórios de código aberto; sistemas ubíquos.

ABSTRACT

We live in a world where ubiquitous systems, also referred to as pervasive systems, are becoming increasingly common and heavily rely on highly distributed software, which is challenging to design and maintain. On the other hand, we have access to an abundance of open-source repositories on GitHub, which can serve as a foundation to streamline the development of new projects. This article describes the methodology employed to collect data from repositories of this kind of systems, highlighting the tools used, and presents preliminary quantitative results. The data collection process involved four stages: identifying developers who forked the selected repositories, listing the repositories of each developer, extracting information from each repository, and tabulating the data for future analysis.
KEYWORDS: forks; open-source repositories; ubiquitous systems.

INTRODUÇÃO

Vivemos em mundo onde sistemas ubíquos, também chamadas de pervasivos, têm se tornado comuns. Esses sistemas são largamente dependentes de software (ALBONICO; MALAVOLTA et al., 2021), que por sua vez, tendem a ser heterogêneos e altamente distribuídos (FERNÁNDEZ-MADRIGAL et al., 2008), tornado-os complexos de projetar e manter (ELKADY; SOBH et al., 2012). Por outro lado, há uma grande quantidade de repositórios de software de código aberto disponíveis em plataformas como o GitHub[1], os quais podem ser utilizados como base para novos projetos, diminuindo a complexidade de desenvolvimento.

[1] Bolsista PIBIC/Fundação Araucária. Universidade Tecnológica Federal do Paraná (UTFPR), Francisco Beltrão, Paraná, Brasil. E-mail: alexandrecorneau@alunos.utfpr.edu.br . ID Lattes: http://lattes.cnpq.br/7895796847820664.
[2] Docente no Departamento de Informática. Universidade Tecnológica Federal do Paraná (UTFPR), Francisco Beltrão, Paraná, Brasil. E-mail: michelalbonico@utfpr.edu.br. ID Lattes: http://lattes.cnpq.br/2972309763879154.

[1] https://github.com/

Não há estudo atual que analise a extensão que repositórios de software aberto ajudam no desenvolvimento de sistemas ubíquos. Em um trabalho anterior (ALBONICO; ROHLING et al., 2021), parte dos autores selecionou repositórios de software[2] no GitHub que podem ser utilizados como base para um estudo desse tipo. Aqueles repositórios são bem documentados e referem-se a sistemas que integram Internet das Coisas e Robótica, duas áreas representativas dos sistemas ubíquos.

Em um projeto de código aberto no GitHub, o processo de colaboração inicia com um *fork*, que consiste na cópia do repositório público para a conta do novo colaborador. Então, o colaborador procede com as alterações no *fork*/cópia do repositório público, efetivando-as por meio de *commits*. Assim que todas as alterações necessárias são efetivadas, o colaborador faz uma requisição de envio para o repositório público original por meio de um *pull request*. Todas as etapas do processo mantém registros que podem ser auditados posteriormente, o que pode ser útil na análise sobre o uso dos repositórios de software públicos.

Neste artigo, descrevemos como foram extraídos os dados dos repositórios de software estudados, as ferramentas utilizadas, e ilustramos os resultados atuais. Para a extração das informações dos *forks* dos repositórios listados no trabalho anterior (ALBONICO; ROHLING et al., 2021), seguimos uma abordagem em 4 fases: i) buscamos todos os desenvolvedores que fizeram *fork* dos repositórios de software pré-selecionados; ii) listamos os repositórios de cada desenvolvedor; iii) extraímos informações de cada repositório; iv) tabulamos os dados extraídos para uma análise qualitativa futura. Neste etapa da pesquisa, somente fazemos uma análise quantitativa preliminar dos dados extraídos.

REVISÃO DA LITERATURA

Na literatura encontramos diversos trabalhos que se baseiam na mineração de repositórios de software (MSR).

Em Farias et al. (2016), os autores fazem um mapeamento sistemático sobre a atividade MSR em artigos das principais conferências na área de Engenharia de Software. Nenhum dos artigos estudados usa *forks* como fonte de dados. O que chama a atenção é que nem mesmo *commits* e *pull requests* são estudados pelos artigos analisados. O mapeamento contrasta com o de outros dois mapeamentos feitos 6 anos mais tarde (VIDONI, 2022; TUTKO; HENLEY; MOCKUS, 2022), os quais evidenciam que as fontes de dados mudaram consideravelmente com o passar dos anos. Nos mapeamentos mais recentes é possível ver dados extraídos dos repositórios do GitHub como fontes de dados, o que inclui, *forks*, *issues*, *commits* e *pull requests*. Em Tukto et al.(2022), por exemplo, *forks* foram usados 27 vezes, em 9.4% dos trabalhos analisados. Nenhum dos trabalhos analisados nos dois últimos mapeamentos tem o mesmo propósito que este.

Brisson et. al (2020) utilizam os *forks* de repositórios do GitHub como uma das fontes de dados no seu estudo para agrupar repositórios de software em famílias. Além dos forks, os autores também consideram atributos adicionais, como como *pull requests* e *estrelas*. O objetivo daquele estudo é diferente do deste trabalho, umas vez que os autores não caracterizam os *forks*.

Pietri et al. (2020) e de Biazzini e Baudry (2014) também utilizam *forks* como fonte de dados de sua pesquisa. No entanto, os autores buscam abordagens independente das plataformas de

[2] https://docs.google.com/spreadsheets/d/1zZBJivV7sx2SO8pTwbgYK3lmMYP2hXs7VvVjS3vLFdY/edit?usp=sharing

repositórios de código, como GitHub e GitLab. As abordagens buscam por *forks* em fluxos de trabalho, como dados do controle de versões e histórico de desenvolvimento, que não são registrados nas plataformas de compartilhamento. Ambos estudos focam mais em como identificar *forks* que não são explícitos do que necessariamente utilizar os *forks* para inferência.

METODOLOGIA

A Figura 1 ilustra a abordagem em três fases que utilizamos neste trabalho para extração de informações dos repositórios do GitHub. Para todas as fases, utilizamos a API REST do GitHub[3].

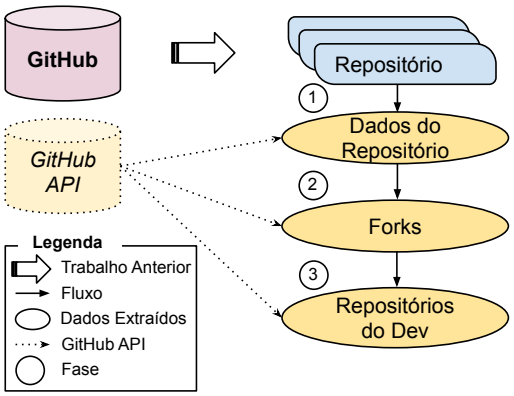

Figura 1 – Abordagem para extração de informações do GitHub.

Fase 1 - Extração dos dados dos repositórios

Inicialmente, buscamos as informações relevantes a respeito dos repositórios listados em um trabalho anterior (ALBONICO; ROHLING et al., 2021), disponíveis publicamente[4]. Para isso, utilizamos a URL da API REST do GitHub como a seguir, informando o repositório:

*https://api.github.com/repos/**[repositório]***

Fase 2 - Extração dos *forks*

Para cada repositório, extraímos a lista de *forks*, contabilizando a quantidade. Para isso, utilizamos a URL no atributo *Forks URL* retornado na fase anterior, conforme o exemplo:

*https://api.github.com/repos/**rwaldron/johnny-five/forks***

Como resultado, obtemos um novo arquivo JSON que, entre outros, mostra o nome do desenvolvedor que fez o *fork*, a URL utilizada para a próxima busca (repositórios do desenvolvedor), e a URL do *fork*.

Fase 3 - Extração dos repositórios dos desenvolvedores

Para cada um dos desenvolvedores com *fork* é recuperado um arquivo JSON com seus dados e a lista de repositórios, usando uma URL como a do exemplo:

*https://api.github.com/users/**Donwilliams501/repos***

Todos os dados extraídos são armazenados em uma base de dados SQLite[5]. O SQLite também permite que os dados sejam facilmente exportados para planilhas de cálculo, para posterior

[3] https://docs.github.com/pt/rest
[4] https://urlis.net/n83nzo3q
[5] https://www.sqlite.org/index.html

Tabela 1 – Projetos com seus *forks* e atualizações dos mesmos.

Repositório Original	Forks	Atualizados	Desatualizados	% Atualizados	% Desatualizados
rwaldron/johnny-five	998	16	982	1,60%	98,40%
hybridgroup/gobot	951	177	774	18,61%	81,39%
NVIDIA-AI-IOT/jetbot	908	53	855	5,84%	94,16%
hybridgroup/cylon	354	174	180	49,15%	50,85%
NVIDIA-AI-IOT/redtail	337	62	275	18,40%	81,60%
aws/aws-iot-device-sdk-cpp	106	9	97	8,49%	91,51%
Samsung/cotopaxi	75	20	55	26,67%	73,33%
NVIDIA-AI-IOT/jetson-cloudnative-demo	43	29	14	67,44%	32,56%
NVIDIA-AI-IOT/Gesture-Recognition	39	39	0	100,00%	0,00%
rdbox-intec/rdbox	36	3	33	8,33%	91,67%
ms-iot/virtual-shields-universal	29	2	27	6,90%	93,10%
iotJumpway/Intel-Examples	28	18	10	64,29%	35,71%
NVIDIA-AI-IOT/jetson-trashformers	23	19	4	82,61%	17,39%
NVIDIA-AI-IOT/Electron	20	20	0	100,00%	0,00%
NVIDIA-AI-IOT/Foresee-Navigation	18	18	0	100,00%	0,00%
Robo4J/robo4j	16	2	14	12,50%	87,50%
AIIAL/HIAS	11	3	8	27,27%	72,73%
pyaiot/pyaiot	11	2	9	18,18%	81,82%
iot-lab/cli-tools	11	0	11	0,00%	100,00%
hybridgroup/cylon-joystick	10	2	8	20,00%	80,00%
suculent/thinx-device-api	8	1	7	12,50%	87,50%
usnistgov/analise!analise!H2204	7	0	7	0,00%	100,00%
hybridgroup/cylon-mqtt	5	0	5	0,00%	100,00%
kaiwaehner/iiot-integration-[...]	3	3	0	100,00%	0,00%
vomyrak/BuddyHub	1	0	1	0,00%	100,00%
scramjs/johnny-five-elements	1	0	1	0,00%	100,00%
nsynapse/cossb	1	0	1	0,00%	100,00%
IoT-Lab-Minden/SwarmRob	1	0	1	0,00%	100,00%
0xJeremy/Dum-E-IOT	1	0	1	0,00%	100,00%
ZeroSum24/IoT-Room-Occupancy-Monitor	1	0	1	0,00%	100,00%
ANRGUSC/iris-riot	1	1	0	100,00%	0,00%
nebrius/johnny-five-iot-edge	1	0	1	0,00%	100,00%
Agile-and-Adaptive-Robotics/Bipedal_Robot	1	0	1	0,00%	100,00%
Pawel2357/IoT_home	0	0	0	0,00%	0,00%
PTiagorio/fikalab	0	0	0	0,00%	0,00%

análise. Todos os artefatos de código e banco de dados estão publicamente disponíveis em um repositório do GitHub [6], que contém um tutorial de execução.

RESULTADOS E DISCUSSÃO

A partir dos **35** repositórios de software originais, identificamos **4056** *forks* de **3812** desenvolvedores distintos (média de 1.06 *forks* por desenvolvedor), e esses desenvolvedores contabilizam **759642** repositórios públicos. Aqui nos atemos a listar os dados quantitativos sobre os *forks* extraídos.

A atualização do *fork* futuramente nos ajudará a entender o papel do desenvolvedor que o possui. Por exemplo, se ele continua contribuindo com o projeto original, ele tende a manter o seu *fork* atualizado. Para saber quais *forks* estavam atualizados e quais não, utilizamos o Selenium[7] para buscar essa informação diretamente na página do projeto no GitHub. A Tabela 1 mostra todos os repositórios originais com a sua quantidade de *forks* e a porcentagem de *forks* atualizados e desatualizados por projeto. Uma análise superficial dos dados da tabela revela projetos com diferentes níveis de atividade e manutenção no GitHub. Enquanto alguns projetos apresentam altas

[6] https://anonymous.4open.science/r/iort-forks-replication-7474/
[7] https://www.selenium.dev/

taxas de atualização e baixa desatualização, indicando um estado saudável de desenvolvimento e interesse contínuo da comunidade, outros projetos exibem baixas taxas de atualização e elevada desatualização, o que pode sugerir um risco de estagnação ou abandono.

Curiosamente, os três projetos 100% atualizados com maior número de *forks* são todos mantidos pela `NVIDIA-AI-IOT`, que tem 5 dentre seus 7 projetos com *forks* bastante atualizados. Todos esses projetos contam com no máximo 3 *commits*, sendo que o projeto `NVIDIA-AI-IOT/Gesture-Recognition` tem somente 1 *commit*. Nesses projetos, todos os *forks* foram criados depois do último *commit*. Portanto, o caso de 100% de atualização dos *forks* acontece devido a projetos com poucos *commits*, que foram adicionados ao GitHub e não foram mais atualizados e um projeto com somente um *fork*, a partir do qual não foram feitas atualizações no repositório original.

DEMOGRAFIA DOS *FORKS*

A Figura 2 ilustra a demografia de cada projeto por ano, que ilustra de forma mais clara o que é mostrado na Tabela 1. É possível observar que a maioria dos projetos têm *forks* sendo feitos durante vários anos subsequentes à sua criação, com destaque para os projetos `retail`, `johnny-five`, `jetbot`, `gobot` e `cylon`. Isso pode estar diretamente relacionado à manutenção do software nesses repositórios originais.

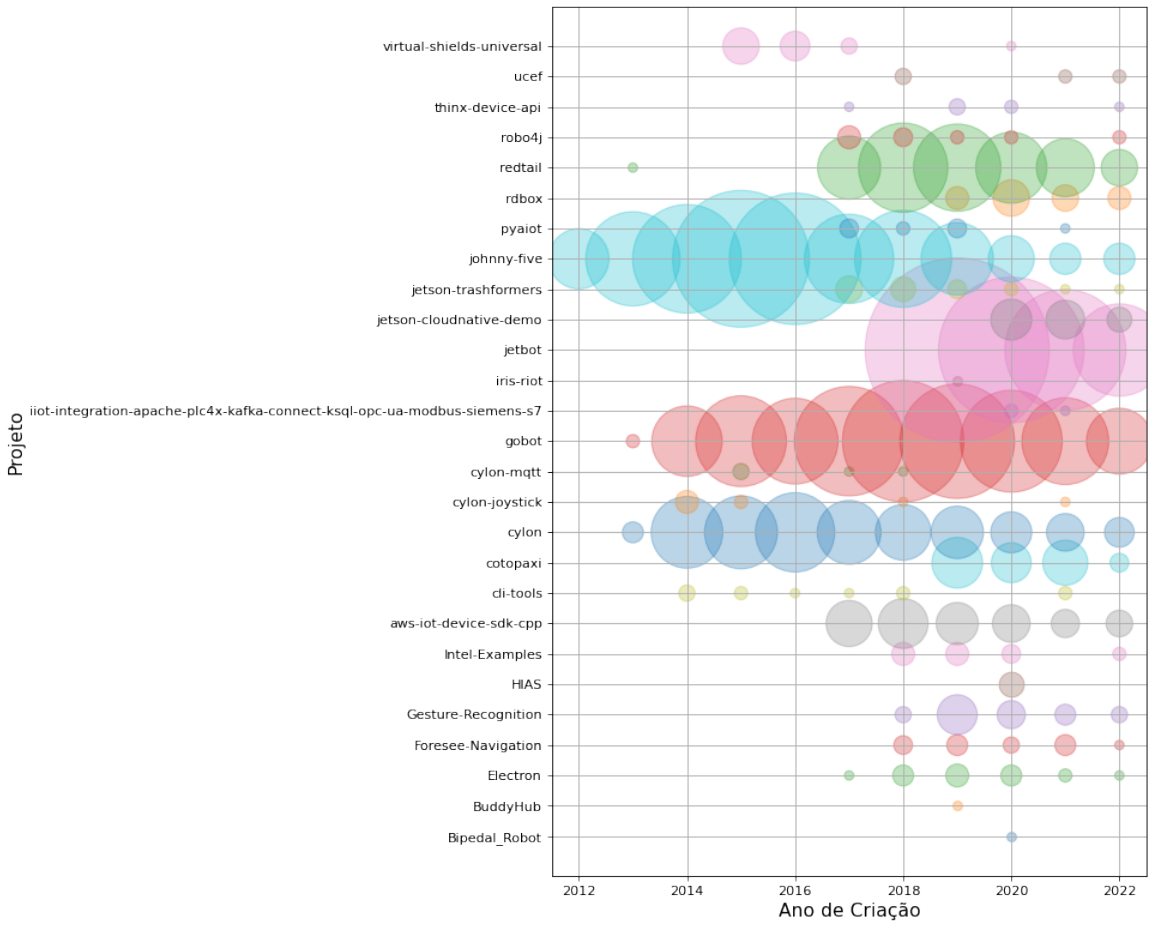

Figura 2 – Quantidade de *forks* de cada repositório criados por ano.

Conflito de interesse

Não há conflito de interesse.

REFERÊNCIAS

ALBONICO, Michel; MALAVOLTA, Ivano et al. Mining Energy-Related Practices in Robotics Software. In: 2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR). [S.l.: s.n.], 2021. P. 483–494. DOI: 10.1109/MSR52588.2021.00060.

ALBONICO, Michel; ROHLING, Adair Jose et al. Mining Evidences of Internet of Robotic Things (IoRT) Software from Open Source Projects. In: SBCARS. [S.l.: s.n.], set. 2021. to appear.

BIAZZINI, Marco; BAUDRY, Benoit. "May the fork be with you": novel metrics to analyze collaboration on GitHub. In: PROCEEDINGS of the 5th international workshop on emerging trends in software metrics. [S.l.: s.n.], 2014. P. 37–43.

BRISSON, Scott; NOEI, Ehsan; LYONS, Kelly. We are family: analyzing communication in GitHub software repositories and their forks. In: IEEE. 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER). [S.l.: s.n.], 2020. P. 59–69.

ELKADY, Ayssam; SOBH, Tarek et al. Robotics middleware: A comprehensive literature survey and attribute-based bibliography. **Journal of Robotics**, Hindawi, v. 2012, 2012.

F. FARIAS, Mário André de et al. A systematic mapping study on mining software repositories. In: PROCEEDINGS of the 31st Annual ACM Symposium on Applied Computing. [S.l.: s.n.], 2016. P. 1472–1479.

FERNÁNDEZ-MADRIGAL, Juan-Antonio et al. A software engineering approach for the development of heterogeneous robotic applications. **Robotics and Computer-Integrated Manufacturing**, v. 24, n. 1, p. 150–166, 2008. ISSN 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2006.10.002. Disponível em: .

PIETRI, Antoine; ROUSSEAU, Guillaume; ZACCHIROLI, Stefano. Forking without clicking: on how to identify software repository forks. In: PROCEEDINGS of the 17th International Conference on Mining Software Repositories. [S.l.: s.n.], 2020. P. 277–287.

TUTKO, Adam; HENLEY, Austin Z.; MOCKUS, Audris. How are Software Repositories Mined? A Systematic Literature Review of Workflows, Methodologies, Reproducibility, and Tools. **ArXiv**, abs/2204.08108, 2022.

VIDONI, M. A systematic process for Mining Software Repositories: Results from a systematic literature review. **Information and Software Technology**, v. 144, p. 106791, 2022. ISSN 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2021.106791. Disponível em: .