Bagaimana Pengembang Perangkat Lunak Menggunakan Tindakan GitHub untuk Mengotomatiskan Alur Kerja Mereka?

Timothy Kinsmanÿ , Mairieli Wessel† , Marco A. Gerosa‡ dan Christoph Treudeÿ
ÿUniversitas Adelaide, Australia †Universitas
Sao Paulo, Brasil ‡Northern Arizona
University, AS
timothy.kinsman@student.adelaide.edu.au, mairieli@ime.usp.br,
marco.gerosa@nau.edu, christoph.treude@adelaide.edu.au

Abstrak—Alat otomatis sering digunakan dalam repositori pengkodean sosial untuk melakukan aktivitas berulang yang merupakan bagian dari proses pengembangan perangkat lunak terdistribusi. Baru-baru ini, GitHub memperkenalkan GitHub Actions, sebuah fitur yang menyediakan alur kerja otomatis untuk pengelola repositori. Meskipun beberapa Aksi telah dibuat dan digunakan oleh para praktisi, relatif sedikit yang dilakukan untuk mengevaluasinya. Memahami dan mengantisipasi dampak penerapan teknologi semacam itu penting untuk perencanaan dan pengelolaan. Penelitian kami adalah penelitian pertama yang menyelidiki bagaimana pengembang menggunakan Actions dan bagaimana beberapa indikator aktivitas berubah setelah penerapannya. Hasil kami menunjukkan bahwa, meskipun hanya sebagian kecil repositori yang mengadopsi GitHub Actions hingga saat ini, terdapat persepsi positif terhadap teknologi tersebut. Temuan kami juga menunjukkan bahwa penerapan Tindakan GitHub meningkatkan jumlah permintaan penarikan bulanan yang ditolak dan menurunkan jumlah komitmen bulanan pada permintaan penarikan gabungan. Hasil ini sangat relevan bagi para praktisi untuk memahami dan men

Ketentuan Indeks—Tindakan GitHub, Bot GitHub, Alur kerja otomatis, Desain Diskontinuitas Regresi

I. PENDAHULUAN

Platform pengkodean sosial, seperti GitHub, telah mengubah sifat kolaboratif pengembangan perangkat lunak sumber terbuka dengan mengintegrasikan mekanisme seperti pelaporan masalah dan permintaan tarik ke dalam alat kontrol versi terdistribusi [1, 2]. Alur kerja pengembangan berbasis tarikan ini menawarkan peluang baru untuk keterlibatan komunitas namun pada saat yang sama meningkatkan beban kerja pengelola repositori untuk berkomunikasi, meninjau kode, menangani masalah perjanjian lisensi kontributor, menjelaskan pedoman proyek, menjalankan pengujian, dan menggabungkan permintaan tarik [3] .

Untuk mengurangi beban kerja intensif ini, pengembang sering kali mengandalkan alat otomatisasi untuk melakukan tugas berulang guna memeriksa apakah kode dibuat, pengujian berhasil, dan kontribusi sesuai dengan panduan gaya yang ditentukan [4]. Proyek GitHub mengadopsi, misalnya, alat untuk mendukung Continuous Integration dan Continuous Delivery or Deployment (CI/CD) [5, 6] dan untuk tinjauan kode [4]. Dalam beberapa tahun terakhir, bot perangkat lunak telah banyak diadopsi untuk mengotomatisasi berbagai tugas yang telah ditentukan sebelumnya seputar permintaan tarik [7]. Dengan mengotomatiskan sebagian alur kerja, pengembang berharap dapat meningkatkan produktivitas dan kualitas [8].

Untuk lebih mendukung otomatisasi, GitHub baru-baru ini memperkenalkan GitHub Actions1 (fitur ini tersedia untuk umum pada bulan November 2019). Tindakan GitHub memungkinkan otomatisasi tugas berdasarkan berbagai pemicu (misalnya, penerapan, permintaan penarikan, masalah, komentar, dll.) dan dapat dengan mudah dibagikan dari satu repositori ke repositori lainnya, sehingga memudahkan untuk mengotomatiskan cara pengembang membangun, menguji, dan menerapkan proyek perangkat lunak.

Namun, masih sedikit yang diketahui mengenai dampak otomatisasi semacam itu dan tantangan yang mungkin timbul pada proses pengembangan proyek. Dalam makalah ini, kami bertujuan untuk memahami bagaimana pengembang perangkat lunak menggunakan GitHub Actions untuk mengotomatiskan alur kerja mereka dan bagaimana dinamika pull request proyek GitHub berubah setelah penerapan GitHub Actions. Untuk mencapai tujuan kami, kami menjawab pertanyaan penelitian berikut: **RQ1:** Bagaimana proyek OSS menggunakan GitHub Actions?

Kami bertujuan untuk memahami seberapa umum repositori menggunakan Tindakan GitHub dan kegunaannya. Sebagai hasil analisis ini, kami menemukan sebagian kecil repositori aktif (0,7% dari 416.266 repositori) mengadopsi GitHub Actions. Tindakan ini tersebar di 20 kategori, termasuk integrasi berkelanjutan, utilitas, dan penerapan. Kami juga menganalisis riwayat penerapan file yang terkait dengan alur kerja GitHub Action untuk memahami bagaimana penggunaan Action yang telah ditentukan sebelumnya berkembang seiring waktu. Secara keseluruhan, kami menemukan bahwa Tindakan tipikal ditambahkan dua kali, dan tidak pernah dihapus atau diubah.

RQ2: Bagaimana penggunaan GitHub Actions dibahas oleh pengembang?

Untuk mendapatkan wawasan tentang bagaimana pengembang memandang GitHub Actions, kami menganalisis secara manual serangkaian 209 masalah GitHub yang membahas GitHub Actions. Kami menemukan kategori diskusi berbeda terkait pemeliharaan dan implementasi GitHub Actions, termasuk peralihan alat otomatisasi lainnya ke Actions, saran untuk mengimplementasikan Actions, serta masalah dan frustrasi.

RQ3: Apa dampak dari GitHub Actions?

Dalam RQ ini, kami menyelidiki apakah indikator aktivitas proyek, seperti jumlah permintaan penarikan yang digabungkan dan tidak digabungkan, jumlah komentar, waktu untuk menutup permintaan penarikan, dan jumlah penerapan berubah setelah adopsi Tindakan GitHub.
Kami menggunakan Desain Diskontinuitas Regresi [9] untuk memodelkan efek adopsi Action di 926 proyek yang telah mengadopsi GitHub Actions selama setidaknya 6 bulan. Temuan kami menunjukkan bahwa, rata-rata, terdapat lebih banyak permintaan penarikan yang ditolak dan

1https://github.com/features/actions

lebih sedikit komitmen pada permintaan penarikan gabungan setelah mengadopsi Tindakan GitHub.

Singkatnya, kami memberikan kontribusi berikut: (i) memberikan perhatian pada GitHub Actions, sumber daya relevan namun terabaikan yang menawarkan dukungan untuk tugas pengembang; (ii) mengkarakterisasi penggunaan GitHub Actions, dan (iii) memberikan pemahaman tentang bagaimana adopsi GitHub Actions berdampak pada aktivitas proyek dan apa yang didiskusikan oleh pengembang tentang hal tersebut.

II. OTOMATISASI ALUR KERJA DENGAN TINDAKAN GITHUB

GitHub Actions adalah API berbasis peristiwa yang disediakan oleh platform GitHub untuk mengotomatiskan alur kerja pengembangan. Tindakan GitHub dapat menjalankan serangkaian perintah setelah peristiwa tertentu terjadi. Peristiwa adalah aktivitas spesifik yang memicu jalannya alur kerja, seperti yang ditunjukkan pada Gambar 1 (lihat ikon). Misalnya, alur kerja dipicu ketika permintaan penarikan dibuat untuk repositori atau ketika permintaan penarikan digabungkan ke dalam cabang utama. Alur kerja ditentukan di direktori .github/**workflows/** dan menggunakan sintaksis YAML, yang memiliki ekstensi file .yml atau .yaml.

Alur kerja dapat berisi satu atau beberapa Tindakan. Pengembang dapat membuat Tindakan mereka sendiri dengan menulis kode khusus yang berinteraksi dengan repositori mereka, dan menggunakannya dalam alur kerja atau mempublikasikannya di GitHub Marketplace. GitHub memungkinkan pengembang untuk membangun Tindakan Docker dan JavaScript dan keduanya memerlukan file metadata untuk menentukan input, output, dan titik masuk utama Action.

Setelah alur kerja berhasil dijalankan, output dapat ditampilkan dengan cara yang berbeda. Salah satu kemungkinannya adalah melalui bot GitHub Action. Bot ini, seperti bot lainnya di GitHub, diimplementasikan sebagai pengguna GitHub yang dapat mengirimkan kontribusi kode, berinteraksi melalui komentar, dan menggabungkan atau menutup permintaan penarikan [10]. Baru-baru ini, pengembang menerbitkan varian GitHub Action untuk banyak bot terkenal (misalnya, Coveralls, Codecov, Snyk) dan popularitas Action ini meningkat pesat [11].

Sebagai contoh adopsi GitHub Actions, pertimbangkan kasus proyek Grammapy2 , paket Python sumber terbuka untuk astronomi sinar gamma. Mulai 13 November 2019, komunitas Grammapy mengadopsi Aksi GitHub yang disebut Interaksi Pertama3 yang bertanggung jawab untuk mengidentifikasi dan menyambut pendatang baru ketika mereka membuat terbitan pertama atau membuka permintaan penarikan pertama pada sebuah proyek. Seperti yang ditunjukkan pada Listing 2, Grammapy membuat alur kerja yang disebut Salam yang mungkin dipicu oleh permintaan tarik baru dan masalah, seperti yang didefinisikan oleh kata kunci **on .** Output dari Tindakan Interaksi Pertama ditampilkan melalui komentar masalah/permintaan tarik yang diposting oleh GitHub Action Bot ketika permintaan tarik atau masalah baru dibuat oleh kontributor baru. Contoh interaksi Tindakan pada masalah GitHub ditunjukkan pada Gambar 3.

2https://github.com/gammapy/gammapy 3https://github.com/marketplace/actions/first-interaction

AKU AKU AKU. DESAIN PENELITIAN

Penelitian ini bertujuan untuk memahami penggunaan GitHub Actions dalam proyek GitHub. Berikut ini, kami menyajikan desain penelitian, pengumpulan data, dan prosedur analisis kami.

A. Memilih Proyek

Kami mengumpulkan kumpulan data proyek sumber terbuka GitHub yang mengadopsi Tindakan GitHub pada suatu saat dalam sejarahnya. Untuk menyusun sampel penelitian kami, kami mulai dengan memilih repositori dari GitHub. Untuk ini, kami menggunakan metadata proyek GitHub dari kumpulan data RepoReapers Munaiah dkk. [12], yang berisi 446.862 repositori GitHub yang diklasifikasikan sebagai berisi proyek perangkat lunak yang direkayasa.

Kami kemudian memfilter kumpulan data ini untuk mempertahankan proyek perangkat lunak sumber terbuka yang pada suatu saat telah mengadopsi Tindakan GitHub. Untuk mengidentifikasi proyek-proyek ini, kami mengambil data dari GitHub API menggunakan toolkit Ruby yang disebut Octokit.rb.4 Kami memverifikasi apakah repositori memiliki file berformat yaml di direktori ./github/workflows. Kumpulan data yang difilter ini terdiri dari 3.190 proyek.

B. Menganalisis penggunaan GitHub Actions

Pertama, kami mengumpulkan dan menganalisis secara kuantitatif jumlah proyek yang menggunakan GitHub Actions dan jumlah Action per proyek **(RQ1).** Kami juga menganalisis file alur kerja proyek yang dipelajari untuk mencari kategori, deskripsi, dan apakah Tindakan tersebut diverifikasi oleh GitHub. Untuk memahami evolusi Tindakan GitHub, kami mengambil riwayat penerapan setiap file alur kerja yang digunakan oleh proyek yang dipelajari.
Untuk tujuan ini, kami membandingkan riwayat penerapan untuk mencari perubahan terkait Tindakan, yang mencakup penambahan, penghapusan, perubahan konfigurasi, atau pembaruan versi.

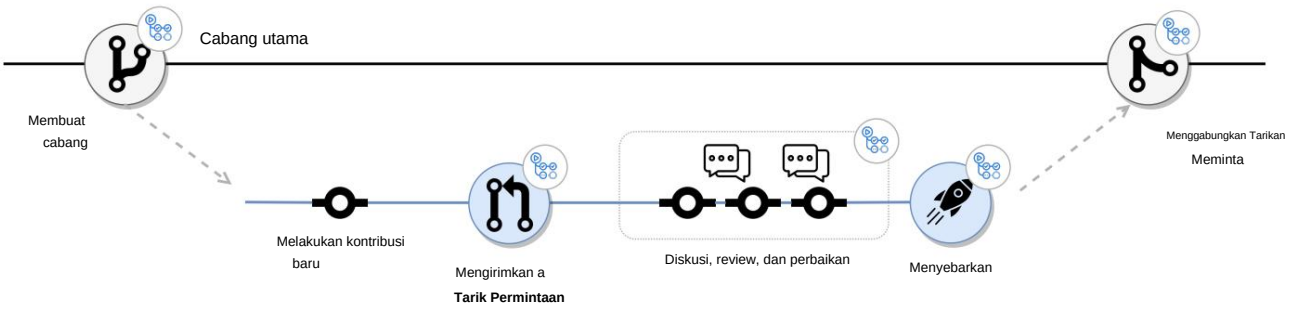
C. Mengkategorikan Diskusi Tindakan GitHub

Untuk menjawab **RQ2,** kami mengumpulkan masalah dari repositori yang menyebutkan “tindakan github” atau “tindakan github”, memiliki setidaknya satu komentar, dan diposting setelah rilis fitur Tindakan GitHub. Kami mengumpulkan 209 isu yang memenuhi kriteria ini. Setelah mengumpulkan masalah-masalah ini, kami menganalisis dan mengkategorikannya secara manual. Dua peneliti secara independen melakukan klasifikasi manual. Penulis pertama makalah ini melakukan klasifikasi manual terhadap 209 isu.
Peneliti lain mengkategorikan subset dari 25 isu acak dengan menggunakan model yang sama. Kami mencetak nilai kappa marjinal bebas sebesar 0,66. Kami kemudian melakukan putaran negosiasi kedua dan mendapatkan nilai kappa margin bebas sebesar 0,76. Fleiss dkk. [13] menyatakan bahwa aturan praktisnya adalah nilai kappa yang kurang dari 0,40 adalah buruk, nilai dari 0,40 hingga 0,75 adalah sedang hingga baik, dan nilai di atas 0,75 adalah sangat baik.

D. Analisis deret waktu

Untuk menjawab **RQ3** dilakukan analisis deret waktu. Kami mengumpulkan data longitudinal untuk variabel hasil yang berbeda dan menangani penerapan GitHub Actions oleh setiap proyek pada tahun tersebut

4http://octokit.github.io/octokit.rb/



Gambar 1. Otomatisasi alur kerja GitHub dengan GitHub Actions (diadaptasi dari GitHub).

```yaml
name: Greetings

on: [pull_request, issues]

jobs:
  greeting:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/first-interaction@v1
      with:
        repo-token: ${{ secrets.GITHUB_TOKEN }}
        issue-message: 'Merci! We will respond
        to your issue shortly. In the meantime,
        try `import gammapy; gammapy.song()`'
        pr-message: 'Graçias! We will review
        your pull request shortly. In the
        meantime, try `import gammapy;
        gammapy.song(karaoke=True)`'
```

Gambar 2. Alur kerja salam Gammapy – salam.yml

Gambar 3. Contoh bot tindakan github menyapa pendatang baru.

kumpulan data kami sebagai “intervensi”. Dengan cara ini, kita dapat menyelaraskan seluruh rangkaian waktu variabel hasil tingkat proyek pada tanggal intervensi dan membandingkan trennya sebelum dan sesudah mengadopsi Tindakan.

Di subbagian berikut, kami merinci berbagai langkah yang terlibat, mulai dari memfilter kumpulan data awal hingga menjalankan model statistik.

1) Menggabungkan variabel-variabel proyek: Kami menganalisis data dari 6 bulan sebelum dan 6 bulan setelah penerapan Aksi.
Mirip dengan pekerjaan sebelumnya [5, 6, 14], kami mengecualikan 30 hari di sekitar tanggal adopsi Tindakan untuk menghindari pengaruh ketidakstabilan yang disebabkan selama periode ini. Setelah itu, kami mengumpulkan data permintaan penarikan individual ke dalam periode bulanan, dengan mempertimbangkan 6 bulan sebelum dan setelah pengenalan Tindakan. Setelah itu, kami memeriksa tingkat aktivitas proyek kandidat, karena banyak proyek di GitHub tidak aktif [2]. Kami menghapus dari kumpulan data kami (i) proyek yang tidak menerima permintaan penarikan apa pun atau (ii) yang menonaktifkan Tindakan selama periode kami menjalankannya.

dipertimbangkan. Setelah menerapkan semua filter, kumpulan data kami terdiri dari 926 proyek aktif yang telah menggunakan setidaknya satu GitHub Action selama 6 bulan. Kami fokus pada variabel terkait permintaan penarikan yang sama seperti pada pekerjaan sebelumnya [14]: **Permintaan**

penarikan yang digabungkan/tidak digabungkan: jumlah kontribusi bulanan (permintaan tarik) yang telah digabungkan, atau ditutup tetapi tidak digabungkan ke dalam proyek, dihitung semua permintaan penarikan tertutup di setiap kerangka waktu.

Komentar pada permintaan penarikan yang digabungkan/tidak digabungkan: jumlah median komentar bulanan yang dihitung atas semua permintaan penarikan yang digabungkan dan tidak digabungkan dalam setiap kerangka waktu.

Permintaan penarikan waktu penggabungan/waktu untuk menutup: median latensi permintaan penarikan bulanan (dalam jam), dihitung sebagai selisih antara waktu saat permintaan penarikan ditutup dan waktu saat permintaan penarikan dibuka. Median dihitung menggunakan semua permintaan penarikan yang digabungkan dan tidak digabungkan di setiap jangka waktu.

Komit permintaan penarikan yang digabungkan/tidak digabungkan: median komitmen bulanan dihitung atas semua permintaan penarikan yang digabungkan dan tidak digabungkan dalam setiap kerangka waktu.

Berdasarkan pekerjaan sebelumnya [5, 6, 14], kami juga mengumpulkan enam kovariat yang diketahui untuk setiap proyek:

Nama proyek: nama proyek yang memiliki permintaan tarik. Nama ini digunakan untuk mengidentifikasi proyek secara unik di GitHub.

Bahasa pemrograman: bahasa pemrograman proyek utama, yang disediakan secara otomatis oleh GitHub.

Waktu sejak permintaan penarikan pertama: dalam bulan, dihitung sejak permintaan penarikan paling awal yang tercatat dalam keseluruhan riwayat proyek. Kami menggunakan variabel ini untuk mengetahui kematangan proyek terkait penggunaan permintaan tarik.

Jumlah total penulis pull request: kami menghitung berapa banyak kontributor yang mengirimkan pull request ke proyek sebagai proksi ukuran komunitas proyek.

Jumlah total komitmen: kami menghitung jumlah total komitmen sebagai proksi untuk tingkat aktivitas suatu proyek.

Jumlah permintaan tarik yang dibuka: jumlah kontribusi bulanan (permintaan tarik) yang diterima dalam setiap jangka waktu. Kami berharap proyek dengan jumlah kontribusi yang tinggi juga memiliki jumlah komentar, latensi, komitmen, dan kontribusi yang digabungkan dan tidak digabungkan yang tinggi.

2) Pendekatan Statistik: Kami memodelkan efek adopsi GitHub Action dari waktu ke waktu di seluruh repositori GitHub menggunakan Regression Discontinuity Design (RDD) [9, 15], mengikuti karya Wessel et al. [14]. RDD adalah teknik yang digunakan untuk memodelkan sejauh mana diskontinuitas pada saat intervensi dan lama setelah intervensi. Teknik ini didasarkan pada asumsi bahwa jika intervensi tidak mempengaruhi hasil, maka tidak akan ada diskontinuitas, dan hasil akan berkelanjutan sepanjang waktu [16]. Model statistik di balik RDD adalah

$$y_i = \ddot{y} + \ddot{y} \cdot \text{waktu}_i + \ddot{y} \cdot \text{intervensi}_i + \ddot{y} \cdot \text{waktu setelah intervensi}_i + \ddot{y} \cdot \text{kontrol}_{si} + \dot{y}_i$$

di mana saya menunjukkan pengamatan untuk proyek tertentu.

Untuk memodelkan perjalanan waktu serta pengenalan GitHub Action, kami mengandalkan tiga variabel: waktu, waktu setelah intervensi, dan intervensi. Variabel waktu diukur dalam bulan pada waktu j dari awal hingga akhir periode pengamatan kami untuk setiap proyek. Kami mempertimbangkan jangka waktu 12 bulan untuk penelitian ini, 6 bulan sebelum dan sesudah adopsi bot.

Variabel intervensi merupakan nilai biner yang digunakan untuk menunjukkan apakah waktu j terjadi sebelum (intervensi = 0) atau setelah peristiwa adopsi (intervensi = 1). Variabel waktu setelah intervensi menghitung jumlah bulan pada waktu j sejak penerapan Tindakan, dan variabel diatur ke 0 sebelum penerapan.
Variabel kontrol memungkinkan analisis dampak adopsi tindakan, dibandingkan mengacaukan dampak yang mempengaruhi variabel terikat. Untuk observasi sebelum intervensi, dengan kontrol konstan, garis regresi yang dihasilkan memiliki kemiringan ÿ, dan setelah intervensi ÿ + ÿ. Besarnya efek intervensi diukur sebagai selisih sebesar ÿ antara dua nilai regresi yi pada saat intervensi.

Mengingat bahwa kami tertarik pada efek Tindakan GitHub pada tren bulanan jumlah permintaan penarikan, jumlah komentar, waktu penutupan permintaan penarikan, dan jumlah penerapan untuk permintaan penarikan yang digabungkan dan yang tidak digabungkan, kami memasangkan delapan model (4 variabel × 2 kasus). Untuk menyeimbangkan positif palsu dan negatif palsu, kami melaporkan nilai p yang dikoreksi setelah menerapkan beberapa koreksi menggunakan metode Benjamini dan Hochberg [17]. Kami menerapkan model RDD sebagai regresi linier efek campuran menggunakan paket R age lmerTest [18].

Mengikuti karya Wessel et al. [14], kami memodelkan nama proyek dan bahasa pemrograman sebagai efek acak [19] untuk menangkap variabilitas proyek-ke-proyek dan bahasa-ke-bahasa [5]. Kami mengevaluasi kecocokan model menggunakan skor marginal (R^2_m) dan kondisional (R^2_c), seperti yang dijelaskan oleh Nakagawa dan Schielzeth [20]. R^2_m dapat diartikan sebagai varians yang dijelaskan oleh efek tetap saja, dan R^2_c sebagai varians yang dijelaskan oleh efek tetap dan acak secara bersama-sama .

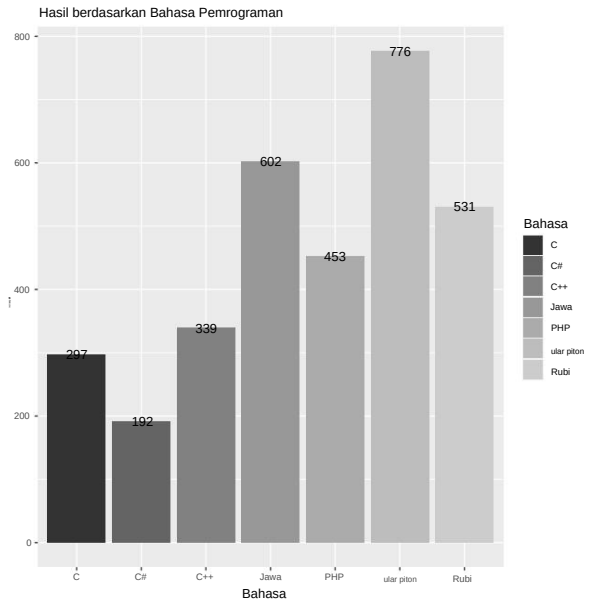
Dalam regresi efek campuran, variabel-variabel yang digunakan untuk memodelkan intervensi bersama dengan efek tetap lainnya dikumpulkan di seluruh proyek, sehingga menghasilkan koefisien yang berguna untuk interpretasi. Interpretasi dari koefisien regresi ini mendukung diskusi mengenai intervensi dan dampaknya, jika ada. Oleh karena itu, kami melaporkan koefisien signifikan ($p < 0,05$) dalam regresi serta variansnya, yang diperoleh dengan menggunakan ANOVA. Selain itu, kami mentransformasikan efek tetap dan variabel terikat yang memiliki varian tinggi [21]. Kami juga memperhitungkan multikolinearitas, tidak termasuk efek tetap dimana faktor inflasi varians (VIF) lebih tinggi dari 5 [21].

IV. HASIL

Berikut ini, kami melaporkan hasil penelitian kami per pertanyaan penelitian.

A. Bagaimana proyek OSS menggunakan GitHub Actions (RQ1)?

Menganalisis sekumpulan 416.266 repositori aktif (yaitu, menerima permintaan penarikan dalam jangka waktu yang relevan, lihat bagian sebelumnya), kami mengidentifikasi 3.190 (0,7%) proyek perangkat lunak sumber terbuka yang telah mengadopsi setidaknya satu GitHub Action pada saat data kami dikumpulkan koleksi. Gambar 4 melaporkan jumlah absolut repositori yang menggunakan GitHub Actions yang dikelompokkan berdasarkan bahasa pemrograman, menunjukkan bahwa pengguna GitHub Actions yang paling menonjol adalah repositori Python, diikuti oleh Java dan Ruby.



Gambar 4. Jumlah repositori yang menggunakan GitHub Actions.

Kami mengumpulkan data hanya 10 bulan setelah GitHub Actions dirilis ke publik dan data kami menunjukkan bahwa sejumlah proyek telah mengadopsi teknologi tersebut. Dari 3.190 repositori GitHub yang menggunakan GitHub Actions, kami menemukan total 708 Action berbeda yang telah ditentukan sebelumnya.

Kami mengumpulkan data dari setiap repositori Action dan juga dari halaman GitHub Marketplace5 untuk mengkategorikan Actions.
Jika dipublikasikan di pasar, suatu Tindakan diklasifikasikan dalam 1–2 kategori oleh penerbit. Tabel I menyajikan kategorisasi Tindakan yang kami temukan. Perhatikan bahwa persentasenya tidak bertambah

5https://github.com/marketplace?type=actions

hingga 100 karena sekitar setengah dari Tindakan ditugaskan ke dua kategori, yaitu kategori primer dan sekunder.

TABEL I
KATEGORISASI TINDAKAN YANG DITEMUKAN DALAM TINDAKAN GITHUB ALUR KERJA.

Kategori Tindakan	**# Tindakan %**
Integrasi berkelanjutan	192 27.12
Keperluan	173 24.44
Penyebaran	87 12.29
Penerbitan	70 9,89
Kualitas kode	53 7.49
Tinjauan kode	45 6.36
Manajemen ketergantungan	36 5.08
Pengujian	33 4.66
Manajemen Sumber Terbuka	30 4.24
Manajemen proyek	27 3.81
Kontainer CI	25 3.53
Mengobrol	18 2.54
Keamanan	13 1.84
Masyarakat	6 0,85
Alat desktop	5 0,70
Seluler	5 0,70
CI Seluler	4 0,56
IDE	3 0,42
Pemantauan	3 0,42
Lokalisasi	2 0,28
Tindakan total	280 39,55
yang tidak dikategorikan	**708 156,77**

Lima kategori Tindakan yang paling sering dilakukan adalah sebagai berikut:

Integrasi berkelanjutan: Tindakan yang bertanggung jawab untuk menjalankan Pipeline CI dan memberi tahu kontributor tentang kegagalan pengujian di CI alat (misalnya, Coba Ulang Langkah, Pengiriman Koki).

Utilitas: Tindakan yang dibuat untuk mengotomatiskan berbagai langkah dalam sistem alur kerja pengembangan pada platform GitHub, sering kali dalam dukungan Tindakan lainnya. Tindakan Properti Baca, untuk misalnya, memeriksa file .properties Java yang mencari yang telah ditentukan sebelumnya properti. Contoh lain dari tindakan utilitas adalah Ganti string, yang menggantikan string yang cocok dengan string reguler yang telah ditentukan sebelumnya ekspresi.

Deployment: Tindakan yang dirancang untuk membangun dan menyebarkan aplikasi berdasarkan permintaan. Salah satu contohnya adalah Action yang disebut Jekyll Deploy, bertanggung jawab untuk membangun dan menyebarkan situs Jekyll ke Halaman GitHub.

Penerbitan: Tindakan yang bertanggung jawab untuk menerbitkan secara otomatis paket ke registri. Misalnya, Aksi Untuk Semantik Rilis adalah Tindakan yang memanfaatkan rilis semantik secara penuh mengotomatiskan alur kerja rilis paket, menentukan selanjutnya nomor versi, menghasilkan catatan rilis dan penerbitan paket.

Kualitas kode: Tindakan yang menganalisis kode sumber (misalnya, kode gaya, cakupan kode, kualitas kode, dan bau) dikirimkan melalui permintaan tarik dan memberikan umpan balik kepada pengembang melalui Pemeriksaan atau komentar GitHub.

Selain itu, kami menemukan bahwa 42 (5,93%) dari 708 Tindakan diverifikasi oleh GitHub. Kreator terverifikasi jika mereka memiliki hubungan yang ada dengan GitHub, dan GitHub telah berhasil erat dengan pembuatnya untuk membuat Tindakan ini.

Lima Action yang paling populer adalah sebagai berikut:

TABEL II
TINDAKAN GITHUB YANG PALING SERING DITAMBAHKAN.

Tindakan	# Keterangan
tindakan/tindakan	7,962 Periksa repositori
checkout/tindakan setup-python/tindakan	1.756 Menyiapkan alur kerja dengan Python
cache/tindakan artefak	1.729 Ketergantungan cache/output build
unggah/tindakan setup-java/download-artifact	1.441 Unggah artefak dari alur kerja
shivammathur/tindakan setup-php/setup-ruby	877 Mengatur alur kerja dengan Java
codecov/tindakan codecov-action/setup-dotnet	580 Unduh artefak dari build
	434 Mengatur alur kerja dengan PHP
	373 Menyiapkan alur kerja dengan Ruby
	253 Unggah liputan ke Codecov
	225 Siapkan alur kerja dengan .NET

TABEL III
TINDAKAN GITHUB YANG SERING DIHAPUS.

Tindakan	rm/tambahkan Deskripsi
masa-iwasaki/setup-rbenv	Pengaturan 1,00 Rbenv
meeDamian/github-release	0,94 Rilis Github
eregon/use-ruby-action	0.81 Ruby bawaan
jakejarvis/s3-sync-action	0,77 Sinkronisasi dengan S3
harmoni758/postgresql-action	0,73 pengaturan PostgreSQL
kiegroup/github-action-build-chain	0.67 Membangun banyak proyek
alexjurkiewicz/setup-ccache	0,67 Pengaturan cache
action/labeler	0.64 Tarik pelabelan permintaan
SamKirkland/FTP-Deploy-Action	0,64 penyebaran server FTP
coverallsapp/github-action	0.62 Unggahan baju

tindakan/checkout: Tindakan utilitas terverifikasi yang melakukan check-out repositori di bawah $GITHUB WORKSPACE. Oleh karena itu, sebuah alur kerja dapat mengakses repositori untuk tugas alur kerja lebih lanjut.

tindakan/setup-python: Tindakan utilitas terverifikasi yang menyiapkan a Lingkungan Python untuk digunakan dalam alur kerja, memungkinkan penggunaan fitur dan perintah Python.

tindakan/cache: Utilitas terverifikasi dan manajemen ketergantungan Tindakan yang memungkinkan cache dependensi dan membangun keluaran untuk meningkatkan waktu eksekusi alur kerja.

action/upload-artifact: Tindakan utilitas terverifikasi yang diunggah artefak dari alur kerja, memungkinkan pengembang untuk berbagi data antara pekerjaan dan menyimpan data setelah alur kerja selesai.

tindakan/setup-java: Tindakan utilitas terverifikasi yang menyiapkan Java lingkungan untuk digunakan dalam alur kerja, memungkinkan penggunaan Java fitur dan perintah, seperti kompilasi dan eksekusi.

Menganalisis riwayat versi repositori GitHub ini, kami menemukan bahwa selain menambahkan Aksi GitHub, repositori juga menghapus Tindakan, mengubah argumennya, dan memperbarui versinya. Kami menyelidiki seberapa sering hal ini terjadi peristiwa yang terjadi dan Tindakan mana yang paling terpengaruh.

Tabel II menunjukkan 10 Tindakan GitHub teratas yang ditambahkan ke repositori paling sering. Di median, Tindakan ditambahkan dua kali (rata-rata: 30).

Tentu saja, Tindakan yang paling sering ditambahkan juga demikian yang paling sering dihapus. Bukannya mutlak angka, oleh karena itu kami menganalisis frekuensi relatif penghapusan, yaitu seberapa sering berbagai Tindakan dihapus dibandingkan dengan seberapa sering mereka ditambahkan. Tabel III menunjukkan sepuluh Tindakan yang paling sering dihapus secara relatif.

TABEL IV
TINDAKAN GITHUB DENGAN BANYAK ARGUMEN YANG DIMODIFIKASI.

Tindakan	mod/tambahkan Deskripsi
andstor/copycat-action	5.45 Penyalinan file
reactivecircus/android-emulator-runner	0.62 Emulator Android
julianoes/Publish-Docker-Github-Action archive/	0.51 Publikasikan buruh pelabuhan
github-actions-slack nanasess/	0,50 Pesan ke Slack
setup-chromedriver elgohr/Publish-	0,45 penyiapan ChromeDriver
Docker-Github-Action google/oss-fuzz docker/	0.43 Publikasikan buruh pelabuhan
build -tindakan	0,42 Pengujian fuzz
push-tindakan/setup-dotnet	0.41 Docker dengan Buildx
	0,38 pengaturan .NET
SamKirkland/FTP-Deploy-Action	0,36 penyebaran server FTP

TABEL V
TINDAKAN GITHUB DENGAN BANYAK VERSI YANG BERUBAH.

Tindakan	vc/tambahkan Deskripsi
gaurav-nelson/github-action -penurunan harga-tautan-periksa	1.79 Pemeriksa tautan
SamKirkland/FTP-Deploy-Action	0.91 Penyebaran server FTP
technote-space/get-diff-action	0,75 Git diff
tindakan/skrip github	0.71 GitHub melalui JavaScript
homoluctus/slatify	0.64 Pemberitahuan Slack
stefanzweifel/git-auto-commit-action	0.60 Berkomitmen Secara Otomatis
crazy-max/ghaction-docker-buildx	0.60 Docker dengan Buildx
peter-evans/create-pull-request	0.53 Pembuatan permintaan tarik
cirrus-actions/rebase	0,50 Permintaan penarikan rebase
dolllabs/action-litmus parallel	0.47 File alur kerja org.

Kami membatasi analisis ini dan analisis berikutnya pada bagian ini Tindakan yang ditambahkan setidaknya sepuluh kali. Seperti yang ditunjukkan tabel, masa-iwasaki/setup-rbenv telah dihapus dalam semua kasus di mana sebelumnya telah ditambahkan. Melihat Aksinya File README,6 ini tidak mengherankan karena berisi catatan 'Jangan Gunakan Tindakan Ini'. Tindakan lain lebih sedikit dihapus sering kali, dengan median nol kepindahan (rata-rata: 5). Beberapa dari Tindakan di 10 teratas memiliki beberapa masalah yang dilaporkan terhadap mereka karena tidak dapat menjalankan beberapa operasi sistem.7

Tabel IV menunjukkan Tindakan GitHub yang argumennya paling sering diubah, secara relatif. Misalnya, argumen andstor/copycat-action, sebuah Action untuk menyalin file dari repositori ke repositori eksternal lain, diubah 5,45 kali lebih sering setelah Action ditambahkan. Tindakan ini memiliki 15 argumen, termasuk argumen yang menunjukkan sumber dan tujuan salinan. Di median, GitHub Argumen tindakan diubah sebanyak nol kali, dengan rata-rata 3.

Beberapa Tindakan GitHub juga sering diperbarui (usia rata-rata: 3, median: 0). Tabel V menunjukkan Tindakan yang telah dilakukan versi paling sering dimodifikasi, dibandingkan dengan seberapa sering versi tersebut ditambahkan. Tindakan di bagian atas daftar ini sedang aktif pengembangan, dengan sembilan rilis dalam enam bulan terakhir di waktu penulisan.

6https://github.com/masa-iwasaki/setup-rbenv

[7] misalnya, https://github.com/meeDamian/github-release/issues/20

Jawaban untuk RQ1. Kami mengidentifikasi 3.190 repositori GitHub aktif yang telah mengadopsi fitur Tindakan GitHub. Kami menemukan 708 Tindakan unik yang telah ditentukan sebelumnya digunakan dalam alur kerja. Tindakan ini tersebar di 20 kategori. Yang paling sering terjadi adalah integrasi berkelanjutan, utilitas, dan penerapan. Tindakan GitHub (median) yang khas adalah ditambahkan dua kali, dan tidak pernah dihapus atau diubah. Beberapa di antaranya Tindakan dihapus, argumennya diubah, dan tindakannya diubah versi berubah berkali-kali, yang mungkin bisa dijelaskan berdasarkan karakteristiknya, seperti riwayat atau nomor rilis argumen.

B. Bagaimana penggunaan GitHub Actions dibahas oleh pengembang? (RQ2)

Kami mengkategorikan 209 masalah GitHub berdasarkan kontennya Diskusi. Tabel VI menunjukkan gambaran umum kategorisasi ini, yang menunjukkan berapa banyak permasalahan yang kami temukan di setiap kategori. Berikut ini kami sajikan kategori-kategorinya.

TABEL VI
KATEGORISASI PEMBAHASAN.

Kategori Masalah	**# Masalah**	**%**
Pemeliharaan Tindakan GitHub	43	20.57
Pengumuman Tindakan GitHub	35	16.74
Meminta Tindakan GitHub untuk diterapkan	34	16.27
Mengalihkan alat CI/CD ke GitHub Actions	32	15.31
Masalah dan frustrasi Tindakan GitHub	31	14.83
Lainnya	31	14.83
jumlah	**209**	**100**

Pemeliharaan Tindakan GitHub: Topik yang paling sering dibahas oleh kontributor dan pengelola sumber terbuka mengenai Tindakan GitHub adalah pemeliharaannya. Dalam kategori ini, kami mengklasifikasikan masalah yang biasa didiskusikan oleh pengembang memelihara Tindakan GitHub, menambahkan atau meminta fitur Tindakan GitHub yang sudah ada sebelumnya, perbaikan cepat, solusi, dan meminta hak admin. Salah satu pengelola, misalnya, membuka masalah untuk meminta perubahan dalam suatu Tindakan yang bertanggung jawab perbedaan pelaporan: “Saat ini, GitHub Action untuk laporan perbedaan generasi dipicu oleh perubahan PR dan penambahan komentar. [...] Permintaan tarik harus dihapus dan hanya pemicu komentar tersisa.” Contoh lain berkaitan dengan isu-isu yang mengarah pada hal-hal kecil perbaikan pada file README, misalnya akibat pemindahan ke Tindakan GitHub.

Pengumuman Tindakan GitHub: Topik berulang lainnya yang dibahas oleh pengembang sumber terbuka tentang masalah GitHub adalah kapan Aksi baru diumumkan. Kategori ini juga terdiri dari masalah yang mengumumkan bahwa GitHub Actions telah diterapkan (tidak menggantikan platform CI/CD yang sudah ada sebelumnya).

Meminta Tindakan GitHub untuk diterapkan: Kami juga menemukan masalah yang menyarankan Tindakan GitHub untuk diperhatikan atau meminta penggunaan Tindakan GitHub. Kami menemukan, misalnya, pengembang yang meminta untuk membuat Tindakan untuk menambahkan label “had PR” untuk suatu masalah setelah permintaan tarik yang menyelesaikan masalah tertentu masalah diserahkan.

Mengalihkan alat CI/CD ke GitHub Actions: Sering kali dilakukan oleh pengembang membuat masalah untuk mendiskusikan peralihan dari CI/CD yang sudah ada sebelumnya platform (misalnya, CircleCI, Jenkins, TravisCI) ke GitHub Actions (tidak termasuk mengimplementasikan GitHub Actions secara paralel dengan alat CI/CD yang sudah ada sebelumnya).

Masalah dan frustrasi Tindakan GitHub: Kategori ini mencakup bug, build rusak, kesalahan, dan frustrasi terkait dengan Tindakan GitHub. Ada bug yang disebabkan oleh kegagalan dari layanan yang diandalkan oleh Action. Misalnya saja seorang pengelola membuka masalah untuk melaporkan bahwa "GitHub Actions on Mac dan Windows gagal karena Numpy hilang."

Lainnya: Masalah dalam kategori ini berkaitan dengan bug yang disebutkan oleh GitHub Tindakan, kebisingan, atau diskusi lain yang tidak terjadi ke dalam kategori lain.

Jawaban untuk RQ2. Secara keseluruhan, diskusi melibatkan permasalahan dan rasa frustrasinya sebanding dengan pengumuman itu Tindakan GitHub telah diterapkan, meminta implementasi, dan mengganti alat CI/CD.

C. Apa Dampak Tindakan GitHub? (RQ3)

Untuk menjawab pertanyaan ini, kami menyelidiki dampaknya Adopsi GitHub Action pada aktivitas proyek dalam empat dimensi: (i) pull request yang digabungkan dan yang tidak digabungkan, (ii) permintaan tarik manusia percakapan, (iii) efisiensi untuk menutup permintaan penarikan, dan (iv) upaya modifikasi. Kita mulai dengan menyelidiki bagaimana Action adopsi berdampak pada jumlah tarikan yang digabungkan dan yang tidak digabungkan permintaan. Kami memasangkan dua model RDD efek campuran, seperti yang dijelaskan dalam Bagian III-D2. Untuk model ini, jumlah permintaan penarikan yang digabungkan/ tidak digabungkan per bulan adalah variabel dependen.
Tabel VII merangkum hasil model-model ini. Selain itu ke koefisien model, tabel juga menunjukkan jumlah kuadrat, dengan penjelasan varians untuk setiap variabel.

TABEL VII
PENGARUH TINDAKAN GITHUB TERHADAP PRS. RESPONNYA ADALAH **LOG(JUMLAH** PRS **YANG DIGABUNG/TIDAK DIGABUNG)** PER BULAN.

	PR yang digabungkan		PR yang tidak digabungkan	
	Koefisien	Jumlah Sq.	Koefisien	Jumlah Sq.
Mencegat	-0,203***		-0,159**	
Log Time SinceFirstPR	0,0002	47,7	-0,001**	6.95
(TotalPRAuthors) log	-0,002	638,6	0,028***	133.69
(TotalCommits) log	0,020***	236,5	0,017**	34.65
(OpenedPRs) log	0,770***	3393,8	0,230***	530.86
(PRComments) log	0,048***	48,9	0,342***	410.05
(PRCommits)	0,246***	105,8	0,200***	73.24
	0,004	1,0	-0,004*	0,01
intervensi waktu	0,014	0,1	0,002	0,03
Waktu sebenarnya setelah intervensi	-0,004	0,2	0,008**	0,47
R marjinal ^{2}	0,88		0,64	
bersyarat R ^{2}	0,91		0,78	

*** p <0,001, ** p <0,01, * hal <0,05

Menganalisis model permintaan tarik gabungan, kami menemukan hal itu bagian efek tetap cocok dengan data (R^2_M = 0,88). Namun, mempertimbangkan R^2_C = 0,91, variabilitas juga muncul dari proyek ke-proyek dan bahasa-ke-bahasa. Di antara efek tetap, kami mencatat bahwa jumlah permintaan penarikan bulanan menjelaskan sebagian besar variabilitas dalam model, menunjukkan bahwa proyek menerima lebih banyak kontribusi cenderung memiliki daya tarik yang lebih besar permintaan, dengan variabel lain tetap konstan. Tak satupun dari Prediktor terkait tindakan memiliki efek yang signifikan secara statistik, artinya tren jumlah permintaan tarik gabungan adalah stasioner dari waktu ke waktu, dan tetap tidak terpengaruh oleh Aksi adopsi.

Mirip dengan model sebelumnya, bagian efek tetap dari model permintaan tarik yang tidak digabungkan cocok dengan data dengan baik (R^2_M = 0,64), meskipun terdapat variabilitas yang cukup besar dijelaskan oleh efek acak (R^2_C = 0,78). Kami mencatat hal serupa hasil pada efek tetap: proyek menerima lebih banyak kontribusi cenderung memiliki lebih banyak permintaan tarik yang tidak digabungkan. Selain itu, tarik permintaan yang menerima lebih banyak komentar cenderung ditolak. Itu efek adopsi Tindakan pada permintaan penarikan yang tidak digabungkan berbeda dengan model sebelumnya. Mengenai rangkaian waktu prediktor, model tidak mendeteksi adanya diskontinuitas pada waktu adopsi. Namun, tren angkanya negatif permintaan penarikan yang tidak digabungkan sebelum adopsi Tindakan terbalik, menuju peningkatan setelah adopsi.

TABEL VIII
PENGARUH TINDAKAN GITHUB TERHADAP KOMENTAR PERMINTAAN TARIK . ITU RESPON ADALAH **LOG (MEDIAN KOMENTAR)** PER BULAN.

	PR yang digabungkan		PR yang tidak digabungkan	
	Koefisien	Jumlah Sq.	Koefisien	Jumlah Sq.
Mencegat	0,020		-0,058**	
Log Time SinceFirstPR	-0,001***	3,29	-0,001***	10.98
(TotalPRAuthors) log	0,055***	47,49	0,031***	169.33
(TotalCommits) log	-0,008	2,71	0,002	25.67
(OpenedPRs) log	-0,005	40,87	0,051***	217.05
(TimeToClosePRs) log	0,077***	376,95	0,113***	930.91
(PRCommits)	0,213***	70,48	0,199***	75.38
	-0,009***	1,54	0,002	0,00
intervensi waktu	0,001	0,02	-0,002	0,01
Waktu sebenarnya setelah intervensi	0,009***	0,70	-0,003	0,05
R marjinal ^{2}	0,38		0,65	
bersyarat R ^{2}	0,54		0,70	

*** p <0,001, ** p <0,01, * hal <0,05

Untuk menyelidiki dampak adopsi Tindakan pada permintaan tarik komunikasi, kami menyesuaikan satu model untuk menggabungkan permintaan tarik dan satu lagi untuk yang tidak digabungkan. Median permintaan tarik komentar per bulan adalah variabel terikat. Tabel VIII menunjukkan hasil model yang dipasang. Mengingat modelnya komentar pada permintaan penarikan gabungan, kami menemukan bahwa gabungan efek tetap dan acak (R^2_C = 0,54) sesuai data lebih baik daripada efek tetap (R^2_M = 0,38), menunjukkan bahwa sebagian besar variabilitas yang dijelaskan dalam data terkait dengan variabilitas proyek-ke-proyek dan bahasa-ke-bahasa, daripada efek tetap. Selain itu, kami juga mengamati bahwa permintaan penarikan waktu-untuk-menutup menjelaskan jumlah terbesar variabilitas dalam model, menunjukkan bahwa komunikasi selama peninjauan permintaan tarik sangat terkait dengan waktu untuk menggabungkannya. Mengenai efek Aksi, kami mencatat a penurunan tren waktu dasar sebelum adopsi; tidak secara statistik diskontinuitas yang signifikan pada saat adopsi; dan jelas

netralisasi tren waktu yang disebutkan di atas setelah adopsi, sebagai ÿ(waktu) + ÿ(waktu setelah intervensi) 0.

Beralih ke model komentar pada tarikan yang tidak digabungkan permintaan, model cocok dengan data dengan baik ($R2_M$ = 0,65) dan ada juga variabilitas yang dijelaskan oleh variabel acak ($R2_C$ = 0,70). Model ini juga menyarankan komunikasi selama peninjauan permintaan tarik sangat terkait dengan waktu untuk menolak permintaan tarik. Tabel VIII menunjukkan bahwa tidak ada dari prediktor terkait Tindakan memiliki signifikansi secara statistik efek, artinya tren komentar dalam tarikan yang tidak digabungkan permintaan tidak berubah seiring berjalannya waktu, dan tetap tidak terpengaruh oleh adopsi Tindakan.

TABEL IX
PENGARUH TINDAKAN GITHUB TERHADAP PRS TIME-TO-CLOSE . ITU RESPON ADALAH **LOG (MEDIAN PRS WAKTU UNTUK TUTUP)** PER BULAN.

	PR yang digabungkan		PR yang tidak digabungkan	
	Koefisien	Jumlah Sq.	Koefisien	Jumlah Sq.
Mencegat	-0,374**		0,053	
Log Time SinceFirstPR	-0,003**	130,3	0,0004	322.6
(TotalPRAuthors) log	0,218***	1386,1	0,067**	3733.9
(TotalCommits) log	0,021	155,6	-0,028	530.3
(OpenedPRs) log	-0,139***	1046,3	0,089***	4588.4
(PRComments) log	1,528***	8543,4	2,816***	23589.5
(PRCommits)	1,520***	4145,8	1,011***	1967.1
	0,013	2,5	-0,005	0,3
intervensi waktu	-0,053	2,1	-0,003	0,0
Waktu sebenarnya setelah intervensi	-0,004	0,1	0,006	0,3
R marjinal ^{2}	0,47		0,63	
bersyarat R^2	0,57		0,67	

*** p <0,001, ** p <0,01, * hal <0,05

Kami memasang dua model RDD dengan median waktu untuk menutup permintaan penarikan per bulan adalah variabel terikat. Hasil ditunjukkan pada Tabel IX. Menganalisis hasil hingga efeknya Tindakan GitHub tentang latensi untuk menggabungkan permintaan tarik, kami menemukan bahwa gabungan efek tetap dan acak sesuai dengan data lebih baik daripada efek tetap. Meskipun ada beberapa variabel yang mempengaruhi tren latensi permintaan tarik, komunikasi selama permintaan tarik bertanggung jawab atas sebagian besar variabilitas data. Hal ini menunjukkan hasil yang diharapkan: semakin banyak usaha kontributor menghabiskan waktu untuk mendiskusikan kontribusinya, semakin banyak waktu kontribusi yang diperlukan untuk menggabungkan. Jumlah komitmen juga menjelaskan besarnya variabilitas data, sejak suatu proyek dengan banyak perubahan memerlukan lebih banyak waktu untuk meninjau dan menggabungkan mereka. Namun, tidak ada satu pun prediktor terkait Tindakan yang memilikinya efek yang signifikan secara statistik pada waktu yang dihabiskan untuk menggabungkan tarikan meminta.

Beralih ke model permintaan tarik yang tidak digabungkan, kami perhatikan agar sesuai dengan data ($R2_M$ = 0,63), dan ada juga a variabilitas dijelaskan oleh variabel acak ($R2_C$ = 0,67). Sebagai di atas, komunikasi selama permintaan tarik bertanggung jawab untuk sebagian besar variabilitas yang ditemui dalam hasil. Serupa dibandingkan model sebelumnya, tidak ada satu pun prediktor terkait Tindakan memiliki efek signifikan secara statistik pada waktu yang dihabiskan untuk menolak permintaan tarik.

Yang terakhir, kami mempelajari apakah adopsi tindakan mempengaruhi jumlah komitmen yang dibuat sebelum dan selama permintaan penarikan

TABEL X
PENGARUH TINDAKAN GITHUB TERHADAP KOMITMEN PERMINTAAN TARIK . ITU RESPON ADALAH **LOG (MEDIAN KOMITMEN)** PER BULAN.

	PR yang digabungkan		PR yang tidak digabungkan	
	Koefisien	Jumlah Sq.	Koefisien	Jumlah Sq.
Mencegat	-0,020***		-0,075**	
Log Time SinceFirstPR	0,001**	7,02	-0,00004	14.14
(TotalPRAuthors) log	-0,060***	83,99	-0,012*	233.35
(TotalCommits) log	0,019**	41,86	0,019***	71,96
(OpenedPRs) log	0,247***	440,55	0,117***	352,79
(PRComments)	0,541***	441,40	0,611***	758.36
	0,010***	1,97	0,002	0,00
intervensi waktu	0,039**	0,56	-0,009	0,07
Waktu sebenarnya setelah intervensi	0,015***	3,18	-0,002	0,03
R marjinal ^{2}	0,47		0,49	
bersyarat R^2	0,60		0,54	

*** p <0,001, ** p <0,01, * hal <0,05

tinjauan. Sekali lagi, kami memasang dua model untuk permintaan penarikan gabungan dan non-gabungan, dengan median penerapan permintaan penarikan per bulan adalah variabel dependen. Hasilnya adalah ditunjukkan pada Tabel X. Menganalisis model komitmen saat digabungkan permintaan tarik, kami menemukan bahwa gabungannya tetap dan acak dampak ($R2_C$ = 0,60) lebih cocok dengan data dibandingkan efek tetap ($R2_M$ = 0,47). Signifikansi statistik dari semua tindakan yang terkait koefisien menunjukkan bahwa penerapan Tindakan terpengaruh jumlah komitmen. Kami mencatat tren peningkatan sebelumnya adopsi dan diskontinuitas yang signifikan secara statistik di waktu adopsi. Selanjutnya, tren positif dalam jumlah permintaan penarikan gabungan sebelum adopsi Tindakan dibatalkan, menuju penurunan setelah adopsi. Selain itu, kita juga bisa perhatikan jumlah komentar permintaan tarik dan jumlah kontribusi per bulan menjelaskan sebagian besarnya variabilitas dalam hasilnya. Hasil ini menunjukkan bahwa semakin banyak ada komentar dan permintaan tarik, semakin banyak komitmen di sana akan.

Menyelidiki hasil permintaan penarikan yang tidak digabungkan model, kami juga menemukan bahwa gabungan tetap dan acak efek lebih cocok dengan data dibandingkan efek tetap. Mirip dengan model sebelumnya, jumlah komentar permintaan tarik per bulan menjelaskan sebagian besar variabilitas hasilnya. Mengenai prediktor deret waktu, model tidak mendeteksi diskontinuitas apa pun pada waktu adopsi. Namun trennya positif di median komitmen sebelum adopsi bot dibatalkan, menuju penurunan setelah adopsi.

Jawab RQ3. Setelah mengadopsi GitHub Actions, aktif rata-rata, permintaan penarikan yang ditolak lebih banyak dan lebih sedikit berkomitmen pada permintaan tarik gabungan.

V. PEMBAHASAN

Baru-baru ini, cara otomatisasi yang mudah, dapat digunakan kembali, dan portabel alur kerja pengembang di GitHub dimungkinkan oleh munculnya Tindakan GitHub. Sejauh ini, literatur menyajikannya bukti langka tentang penggunaan GitHub Actions oleh GitHub repositori [11]. Dalam karya ini, kami berkontribusi dengan memperkenalkan

dan mensistematisasikan bukti mengenai penggunaan, evolusi, dan dampak dari Tindakan tersebut. Temuan kami menyumbangkan pengetahuan baru tentang bagaimana pengembang perangkat lunak menggunakan GitHub Actions. Pertama, kami menunjukkan bahwa 3.190 (0,7%) dari 416.266 repositori di kumpulan data kami telah mengadopsi GitHub Actions pada saat analisis. Selain itu, kami menemukan bahwa 708 Tindakan unik yang telah ditentukan sebelumnya telah digunakan dalam repositori menggunakan Tindakan GitHub. Meskipun 39,55% dari Action ini tidak dikategorikan, 5,93% telah diverifikasi, yang menunjukkan bahwa sebagian besar Action di GitHub dibuat oleh komunitas. Tindakan Tak Berkategori adalah Tindakan yang belum dipublikasikan ke GitHub Marketplace, sehingga persentase ini menunjukkan komunitas aktif di sekitar GitHub Actions.

Menganalisis evolusi historis Tindakan, kami menemukan bahwa beberapa Tindakan memerlukan pemeliharaan setelah ditambahkan ke repositori. Meskipun suatu Tindakan (median) biasa ditambahkan dua kali (ke repositori yang sama atau berbeda) dan tidak pernah dihapus atau diubah, sebagian besar Tindakan akan dihapus, argumennya diubah, atau diperbarui ke versi baru. Bagi pengelola repositori, ini berarti bahwa menambahkan Tindakan GitHub secara efektif menambah ketergantungan lain pada proyek yang perlu dipertahankan, mungkin menjadi usang, dan perlu disesuaikan seiring berjalannya waktu. Di sisi lain, hal ini biasa terjadi pada fitur baru, dan faktanya semua repositori yang telah mengadopsi GitHub Actions hingga saat ini dapat dianggap sebagai pengguna awal. Tindakan tipikal juga tampaknya tidak memerlukan pemeliharaan berkelanjutan, setidaknya tidak dalam jangka waktu yang dipertimbangkan dalam penelitian kami.

Pengumuman bahwa GitHub Actions telah diimplementasikan, permintaan implementasi dan peralihan alat CI/CD melebihi diskusi yang melibatkan masalah dan frustrasi, sehingga menunjukkan persepsi positif terhadap GitHub Actions.
Dengan fitur baru seperti GitHub Actions, tidak mengherankan jika beberapa sentimen negatif ditemukan di forum diskusi terkait, khususnya menanyakan mengapa fitur lain diperlukan. Namun, secara anekdot, para pengembang tampaknya mengapresiasi premis GitHub Actions untuk membantu membakukan penggunaan bot di GitHub, dan faktanya, sebagian besar diskusi adalah tentang peralihan alat CI/CD yang sudah digunakan oleh repositori ke GitHub Actions yang setara. .

Kami menemukan bahwa dua indikator aktivitas memiliki pengaruh yang signifikan secara statistik pada proses permintaan penarikan setelah penerapan Tindakan GitHub. Berdasarkan hasil regresi, jumlah median permintaan penarikan yang ditolak meningkat setelah penerapan Tindakan. Hal ini mungkin menunjukkan bahwa pengelola proyek mulai mendapatkan masukan yang lebih cepat dan jelas mengenai permintaan penarikan, sehingga membantu mereka mengidentifikasi masalah besar pada sejumlah besar kontribusi. Selain itu, Tindakan GitHub menghasilkan efek yang berbeda pada permintaan penarikan yang tidak digabungkan jika dibandingkan dengan efek penggunaan bot perangkat lunak—Wessel dkk. [14] melaporkan bahwa pengenalan bot pada ulasan permintaan tarik menyebabkan lebih sedikit permintaan tarik yang ditolak.

Dari hasil regresi, kami juga melihat peningkatan jumlah median penerapan pada permintaan penarikan gabungan tepat setelah adopsi bot. Hal ini masuk akal dari sisi kontributor, karena Tindakan tersebut memperkenalkan langkah evaluasi sekunder pada permintaan penarikan. Terutama di awal adopsi, itu Tindakan mungkin meningkatkan jumlah komitmen karena kebutuhan untuk memenuhi semua persyaratan dan mendapatkan kode yang stabil. Namun setelah itu, terjadi penurunan jumlah median komitmen pada permintaan penarikan gabungan per bulan.

Pekerjaan kami mempunyai implikasi bagi para peneliti dan praktisi. Bagi peneliti yang tertarik dengan bot perangkat lunak, penting untuk memahami peran GitHub Actions dalam lanskap bot.
Didukung oleh GitHub, GitHub Actions kemungkinan besar akan tetap ada, dan kami sudah melihat bukti perangkat lunak yang ada, seperti alat cakupan pengujian, diintegrasikan ke dalam dan dikemas sebagai GitHub Actions. Penting untuk memahami bagaimana Tindakan tersebut mempengaruhi interaksi pengembang dalam upaya mereka mengembangkan perangkat lunak, dan penelitian kami memberikan langkah pertama ke arah ini. Upaya tambahan juga diperlukan untuk menyelidiki dampaknya terhadap pendatang baru, yang sudah menghadapi berbagai hambatan [22, 23]. Pendidik juga dapat melihat peluang di GitHub Actions untuk membangun alat otomatisasi guna mendukung tugas OSS mereka dengan lebih baik [24].

Praktisi perlu mengambil keputusan berdasarkan informasi apakah akan mengadopsi GitHub Actions (atau bot perangkat lunak secara umum) ke dalam proyek mereka dan bagaimana menggunakannya secara efektif. Selain itu, Tindakan GitHub memungkinkan mereka mengotomatiskan tugas berulang dalam proyek mereka dengan Tindakan GitHub kustom mereka sendiri. Saat ini, pada tahap pengguna awal, GitHub Actions menyediakan ratusan Action yang berbeda, sehingga berpotensi menyulitkan praktisi untuk memutuskan Action mana yang akan digunakan, jika ada. Pekerjaan kami memberikan data empiris pertama tentang Tindakan mana yang saat ini digunakan, bagaimana tindakan tersebut berkembang, dan apa dampaknya terhadap proses pembangunan. Kami berharap karya ini akan menginspirasi lebih banyak repositori untuk mengadopsi GitHub Actions untuk proyek mereka.

VI. BATASAN DAN ANCAMAN TERHADAP VALIDITAS

Pada bagian ini, kami membahas keterbatasan dan ancaman terhadap validitas dan cara kami memitigasinya. Untuk tujuan replikasi, kami membuat data dan kode sumber kami tersedia untuk umum.8 **Validitas Eksternal:** Karena kami memilih proyek perangkat lunak yang direkayasa, temuan kami mungkin tidak dapat digeneralisasikan ke proyek GitHub lainnya atau semua. Salah satu cara untuk mengatasi ancaman ini adalah dengan mempelajari proyek-proyek non-rekayasa yang dihosting di GitHub. Selain itu, meskipun kami mempertimbangkan sejumlah besar proyek dan hasil kami menunjukkan tren umum, kami menyarankan untuk menjalankan analisis tersegmentasi saat menerapkan hasil kami pada proyek tertentu. Selain itu, kami fokus pada variabel terkait permintaan tarik yang sama seperti pada pekerjaan sebelumnya [5, 6, 14], meninggalkan efek dan artefak lain untuk pekerjaan selanjutnya.

Validitas Konstruk: Seperti yang dinyatakan oleh Kalliamvakou et al. [25], banyak permintaan penarikan gabungan tampak tidak digabungkan. Karena kami mempertimbangkan jumlah permintaan penarikan gabungan, hasil kami mungkin terpengaruh oleh ancaman ini. Studi kami dapat direplikasi ketika cara otomatis untuk mendeteksi masalah ini dikembangkan.

Validitas Internal: Untuk mengurangi ancaman internal, kami menerapkan beberapa langkah pemfilteran data pada model statistik. Kami memvariasikan kriteria pemfilteran data untuk memastikan ketangguhan kami

8https://zenodo.org/record/4626256

model. Misalnya, kami memfilter proyek yang tidak menerima permintaan penarikan selama beberapa bulan dan mengamati fenomena serupa.
Kami juga melakukan serangkaian tes plasebo [15] menggunakan model yang sama dengan adopsi yang ditetapkan secara artifisial pada tanggal berbeda untuk memastikan ketahanan model. Asumsi eksogenitas pengobatan mungkin menjadi ancaman. Kami menambahkan beberapa kontrol yang mungkin mempengaruhi variabel independen untuk mengurangi faktor perancu.

VII. PEKERJAAN TERKAIT

Belum ada penelitian yang menyelidiki GitHub Actions.
Namun penelitian sebelumnya telah menyelidiki alat otomatisasi lain seperti bot perangkat lunak, integrasi berkelanjutan, dan pengiriman berkelanjutan.

A. Bot Perangkat

Lunak Di GitHub, bot perangkat lunak sering kali diintegrasikan ke dalam alur kerja permintaan tarik [26], untuk melakukan berbagai tugas. Ini termasuk memperbaiki bug [27], memfaktorkan ulang kode sumber [28], merekomendasikan alat untuk membantu pengembang [29] dan memperbarui dependensi yang sudah ketinggalan zaman [30]. Bot perangkat lunak telah diusulkan untuk mendukung aspek teknis dan sosial dari aktivitas pengembangan perangkat lunak [31], seperti komunikasi dan pengambilan keputusan [32]. Van Tonder dan Le Goues [33] percaya bahwa bot perangkat lunak merupakan tambahan yang menjanjikan bagi perangkat pengembang karena mereka menjembatani kesenjangan antara pengembangan perangkat lunak manusia dan pengembangan perangkat lunak otomatis proses.

Namun, memahami bagaimana interaksi bot perangkat lunak memengaruhi pengembang manusia merupakan sebuah tantangan besar. Lantai dkk. [32] menyoroti bahwa potensi dampak negatif bot perangkat lunak masih diabaikan, karena cara bot perangkat lunak ini berinteraksi pada permintaan tarik dapat mengganggu dan dianggap tidak ramah [34].
Wessel dkk. [7] menyelidiki penggunaan dan dampak bot perangkat lunak untuk mendukung kontributor dan pengelola dengan permintaan tarik.
Setelah mengidentifikasi bot di repositori GitHub populer, penulis mengklasifikasikan bot ini ke dalam 13 kategori sesuai dengan tugas yang mereka lakukan. Bot ketiga yang paling sering digunakan adalah bot peninjau kode. Wessel dkk. [14] juga menggunakan desain diskontinuitas regresi pada proyek OSS, mengungkapkan bahwa adopsi bot meningkatkan jumlah permintaan penarikan gabungan bulanan, mengurangi permintaan penarikan bulanan yang tidak digabungkan, dan mengurangi komunikasi antar pengembang.

B. Integrasi Berkelanjutan dan Pengiriman Berkelanjutan (CI/CD)

Meningkatkan kualitas perangkat lunak dan mengurangi risiko adalah tujuan utama CI seperti yang dikemukakan oleh Duvall et al. [35]. Dengan memperkenalkan integrasi berkelanjutan pada proses permintaan tarik, temuan dari Vasilescu dkk. [8] dengan jelas menunjukkan manfaat CI.
Semakin banyak permintaan tarik yang diproses, semakin banyak yang diterima dan digabungkan, dan semakin banyak juga yang ditolak. Dalam konteks pendidikan Ilmu Komputer, meningkatnya pendaftaran menyulitkan instruktur dan asisten pengajar untuk memberikan umpan balik yang memadai pada pekerjaan setiap siswa. Hu dkk. [36] telah menyiapkan penganalisis kode statis dan layanan integrasi berkelanjutan di GitHub untuk membantu siswa memeriksa gaya dan fungsionalitas kode. Dengan mengimplementasikan tiga bot, penulis menunjukkan hasil yang ditemukan bahwa lebih dari 70% siswa menganggap saran yang diberikan oleh bot berguna dan dapat memberikan lebih banyak umpan balik (rata-rata enam kali lebih banyak) dibandingkan staf pengajar. Sebuah survei oleh Chen et al. [37] melaporkan bahwa dari ratusan miliar dolar yang dihabiskan untuk gaji pengembang, hingga 25% digunakan untuk memperbaiki bug [37]. Oleh karena itu, integrasi berkelanjutan memiliki potensi besar untuk mengurangi upaya dan biaya manusia dengan memperbaiki bug secara otomatis.

Pekerjaan sebelumnya juga telah menyelidiki dampak CI dan alat peninjauan kode pada proyek GitHub [4, 5, 6] sepanjang waktu. Sementara Zhao dkk. [5] dan Cassee dkk. [6] berfokus pada dampak pengenalan alat Travis CI terhadap praktik pembangunan, Kavaler dkk. [4] beralih ke dampak linter, manajer ketergantungan, dan alat pelapor liputan. Pekerjaan kami memperluas literatur ini dengan memberikan penyelidikan yang lebih mendalam tentang dampak adopsi GitHub Actions.

VIII. KESIMPULAN

Dalam tulisan ini, kami menyelidiki bagaimana pengembang perangkat lunak menggunakan Tindakan GitHub untuk mengotomatiskan alur kerja mereka, bagaimana mereka mendiskusikan Tindakan ini pada pelacak masalah, dan apa dampak penerapan Tindakan tersebut pada permintaan penarikan. Meskipun beberapa Tindakan telah diusulkan dan diadopsi oleh komunitas perangkat lunak sumber terbuka, relatif sedikit yang dilakukan untuk mengevaluasi praktiknya. Untuk memahami dampaknya terhadap praktik, kami mengumpulkan dan menganalisis data dari 3.190 repositori GitHub aktif. Lebih lanjut, untuk memahami dampaknya terhadap praktik, kami menganalisis sampel secara statistik dari 926 proyek sumber terbuka yang dihosting di GitHub.

Pertama, temuan menunjukkan bahwa hanya sebagian kecil dari 3.190 repositori yang menggunakan GitHub Actions. Kami juga menemukan bahwa 708 Tindakan unik yang telah ditentukan sebelumnya digunakan dalam alur kerja. Selanjutnya, kami mengumpulkan dan menganalisis isu-isu terkait GitHub Actions dan menemukan bahwa sebagian besar diskusi bersifat positif. Temuan ini menunjukkan bahwa GitHub Actions mendapat sambutan positif secara keseluruhan di kalangan pengembang perangkat lunak.
Dengan memodelkan data seputar pengenalan GitHub Actions, kami melihat hasil yang berbeda dari permintaan penarikan yang digabungkan dan yang tidak digabungkan. Jumlah penerapan bulanan permintaan penarikan gabungan menurun setelah adopsi Tindakan GitHub dan ada juga lebih banyak permintaan penarikan bulanan yang ditolak.

Temuan kami mengungkap bagaimana pengguna awal menggunakan, berdiskusi, dan terkena dampak GitHub Actions. Belajar dari pengguna awal tersebut dapat memberikan wawasan untuk membantu komunitas sumber terbuka memutuskan apakah akan menggunakan GitHub Actions dan bagaimana menggunakannya secara efektif. Pekerjaan di masa depan mencakup penyelidikan kualitatif terhadap dampak penerapan GitHub Actions dan perluasan analisis kami untuk mempertimbangkan dampak berbagai jenis Actions dan indikator aktivitas.

UCAPAN TERIMA KASIH

Pekerjaan ini sebagian didukung oleh Coordenac¸ao˜ de Aperfeic¸oamento de Pessoal de N´ývel Superior – Brasil
(CAPES) – Kode Keuangan 001, CNPq (hibah 141222/2018-
2), hibah NSF 1815503 dan 1900903, dan Australia

Penghargaan Peneliti Karir Awal Penemuan Dewan Riset (DECRA) skema pendanaan (DE180100153)

REFERENSI

[1] L. Dabbish, C. Stuart, J. Tsay, dan J. Herbsleb, “Social coding in GitHub: Transparansi dan kolaborasi dalam repositori perangkat lunak terbuka,” dalam Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, ser . CSCW '12. New York, NY, AS: ACM, 2012, hlm.1277–1286.

[2] G. Gousios, M. Pinzger, dan A. van Deursen, “Sebuah studi eksplorasi model pengembangan perangkat lunak berbasis tarik,” dalam Prosiding Konferensi Internasional ke-36 tentang Rekayasa Perangkat Lunak. ACM, 2014, hlm.345–355.

[3] G.Gousios, M.-A. Storey, dan A. Bacchelli, “Praktik kerja dan tantangan dalam pengembangan berbasis tarikan: Perspektif kontributor,” dalam Prosiding Konferensi Internasional ke-38 tentang Rekayasa Perangkat Lunak, ser.
ICSE '16. New York, NY, AS: ACM, 2016, hlm.285–296.

[4] D. Kavaler, A. Trockman, B. Vasilescu, dan V. Filkov, “Pilihan alat penting: alat jaminan kualitas JavaScript dan hasil penggunaan dalam proyek GitHub,” dalam Prosiding Konferensi Internasional ke-41 tentang Insinyur Perangkat Lunak. IEEE Press, 2019, hlm.476–487.

[5] Y. Zhao, A. Serebrenik, Y. Zhou, V. Filkov, dan B. Vasilescu, “Dampak integrasi berkelanjutan pada praktik pengembangan perangkat lunak lainnya: studi empiris skala besar,” dalam Prosiding IEEE ke-32 /Konferensi Internasional ACM tentang Teknik Perangkat Lunak Otomatis. IEEE Press, 2017, hlm.60–71.

[6] N. Cassee, B. Vasilescu, dan A. Serebrenik, “The silent helper: dampak integrasi berkelanjutan pada tinjauan kode,” dalam Konferensi Internasional IEEE ke-27 tentang Analisis, Evolusi, dan Rekayasa Ulang Perangkat Lunak. Masyarakat Komputer IEEE, 2020.

[7] M. Wessel, BM de Souza, I. Steinmacher, IS Wiese, I. Polato, AP Chaves, dan MA Gerosa, “Kekuatan bot: Mengkarakterisasi dan memahami bot dalam proyek OSS,” Proc. ACM Hum.-Komputasi. Berinteraksi., jilid. 2, tidak.
CSCW, hlm. 182:1–182:19, November 2018.

[8] B. Vasilescu, Y. Yu, H. Wang, P. Devanbu, dan V. Filkov, “Hasil kualitas dan produktivitas yang berkaitan dengan integrasi berkelanjutan di GitHub,” dalam Prosiding Pertemuan Gabungan ke-10 tahun 2015 tentang Landasan Perangkat Lunak Insinyur, ser. ESEC/FSE 2015. New York, NY, AS: ACM, 2015, hlm.805–816.

[9] DL Thistlethwaite dan DT Campbell, “Analisis diskontinuitas regresi: Sebuah alternatif terhadap eksperimen ex post facto.” Jurnal Psikologi Pendidikan, vol. 51, tidak. 6, hal. 309, 1960.

[10] M. Wessel dan I. Steinmacher, “Sisi yang tidak nyaman dari bot perangkat lunak pada permintaan tarik,” dalam Lokakarya Internasional ke-2 tentang Bot dalam Rekayasa Perangkat Lunak, ser. BotSE '20, 2020.

[11] M. Golzadeh, A. Decan, D. Legay, dan T. Mens, “Kumpulan data dan model klasifikasi kebenaran dasar untuk mendeteksi bot dalam terbitan GitHub dan komentar PR,” 2020.

[12] N. Munaiah, S. Kroh, C. Cabrey, dan M. Nagappan, “Menyeleksi GitHub untuk proyek perangkat lunak yang direkayasa,” Em pirical Software Engineering, vol. 22, tidak. 6, hal.3219–3253, 2017.

[13] J. Fleiss, B. Levin, dan M. Paik, Metode Statistik untuk Tarif dan Proporsi, ser. Seri Wiley dalam Probabilitas dan Statistik. Wiley, 2013.

[14] M. Wessel, A. Serebrenik, I. Wiese, I. Steinmacher, dan MA Gerosa, “Pengaruh adopsi bot peninjau kode pada permintaan tarik ke proyek OSS,” pada Konferensi Internasional IEEE tentang Pemeliharaan dan Evolusi Perangkat Lunak (ICSME) tahun 2020), 2020, hlm.1–11.

[15] GW Imbens dan T. Lemieux, “Desain diskontinuitas regresi: Panduan untuk berlatih,” Jurnal ekonometrik, vol. 142, tidak. 2, hal.615–635, 2008.

[16] T. Cook dan D. Campbell, Eksperimen Semu: Masalah Desain dan Analisis untuk Pengaturan Lapangan. Houghton Mifflin, 1979.

[17] Y. Benjamini dan Y. Hochberg, “Mengontrol tingkat penemuan palsu: pendekatan praktis dan kuat untuk pengujian berganda,” Jurnal masyarakat statistik Kerajaan: seri B (Metodologis), vol. 57, tidak. 1, hal.289–300, 1995.

[18] A. Kuznetsova, PB Brockhoff, dan RHB Chris tensen, “paket lmertest: tes dalam model efek campuran linier,” Journal of Statistical Software, vol. 82, tidak. 13, 2017.

[19] A. Gaÿecki dan T. Burzykowski, Model efek campuran linier menggunakan R: Pendekatan langkah demi langkah. Springer Sains & Media Bisnis, 2013.

[20] S. Nakagawa dan H. Schielzeth, “Metode umum dan sederhana untuk memperoleh R2 dari model efek campuran linier umum,” Metode dalam ekologi dan evolusi, vol. 4, tidak. 2, hal.133–142, 2013.

[21] S. Sheather, Pendekatan modern terhadap regresi dengan R. Springer Sains & Media Bisnis, 2009.

[22] S. Balali, I. Steinmacher, U. Annamalai, A. Sarma, dan MA Gerosa, “Hambatan pendatang baru... hanya itu saja? analisis hambatan mentor dan pendatang baru dalam proyek OSS,” Computer Supported Cooperative Work (CSCW), vol. 27, tidak. 3, hal.679–714, 2018.

[23] I. Steinmacher, T. Conte, MA Gerosa, dan D. Red miles, “Hambatan sosial yang dihadapi oleh pendatang baru yang memberikan kontribusi pertama mereka dalam proyek perangkat lunak sumber terbuka,” dalam Prosiding konferensi ACM ke-18 tentang Kerjasama yang didukung Komputer & komputasi sosial, 2015, hlm. 1379–1392.

[24] GHL Pinto, F. Figueira Filho, I. Steinmacher, dan MA Gerosa, “Melatih insinyur perangkat lunak menggunakan perangkat lunak sumber terbuka: perspektif para profesor,” pada Konferensi IEEE ke-30 tentang Pendidikan dan Pelatihan Rekayasa Perangkat Lunak (CSEE&T) tahun 2017. IEEE, 2017, hlm.117–121.

[25] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, DM German, dan D. Damian, “Janji dan bahaya penambangan GitHub,” dalam Proceedings of the 11th Working Conference on Mining Software Repositories, ser . MSR 2014. New York, NY, AS: ACM, 2014, hlm.92–101.

[26] L. Erlenhov, FG de Oliveira Neto, R. Scandariato, dan P. Leitner, “Bot saat ini dan masa depan dalam pengembangan perangkat lunak,” dalam Prosiding Lokakarya Internasional ke-1 tentang Bot dalam Rekayasa Perangkat Lunak, ser. BotSE '19. IEEE Press, 2019, hal. 7–11.

[27] M. Monperrus, “Kontribusi bot perangkat lunak yang dapat dijelaskan: Studi kasus perbaikan bug otomatis,” dalam Prosiding Lokakarya Internasional ke-1 tentang Bot dalam Rekayasa Perangkat Lunak, ser. BotSE '19. Piscataway, NJ, AS: IEEE Press, 2019, hlm.12–15.

[28] M. Wyrich dan J. Bogner, “Menuju bot otonom untuk pemfaktoran ulang kode sumber otomatis,” dalam Prosiding Lokakarya Internasional ke-1 tentang Bot dalam Rekayasa Perangkat Lunak, ser. BotSE '19. Piscataway, NJ, AS: IEEE Press, 2019, hlm.24–28.

[29] C. Brown dan C. Parnin, “Maaf mengganggu Anda: Merancang bot untuk rekomendasi yang efektif,” dalam Prosiding Lokakarya Internasional ke-1 tentang Bot dalam Rekayasa Perangkat Lunak, ser. BotSE '19. IEEE Press, 2019, hal. 54–58.

[30] S. Mirhosseini dan C. Parnin, “Dapatkah permintaan penarikan otomatis mendorong pengembang perangkat lunak untuk meningkatkan ketergantungan yang sudah ketinggalan zaman?” dalam Prosiding Konferensi Internasional IEEE/ACM ke-32 tentang Rekayasa Perangkat Lunak Otomatis, ser. ASE 2017. IEEE Press, 2017, hal. 84–94.

[31] B. Lin, A. Zagalsky, M. Storey, dan A. Serebrenik, “Mengapa pengembang malas: Memahami bagaimana tim perangkat lunak menggunakan Slack,” dalam Prosiding Konferensi ACM ke-19 tentang Kerja Koperasi yang Didukung Komputer dan Komputasi Sosial Rekan, Ser. Pendamping CSCW '16. New York, NY, AS: ACM, 2016, hlm.333–336.

[32] M.-A. Storey dan A. Zagalsky, “Mengganggu produktivitas pengembang satu bot pada satu waktu,” dalam Prosiding Simposium Internasional ACM SIGSOFT ke-24 tahun 2016 tentang Fondasi Rekayasa Perangkat Lunak, ser. FSES 2016. New York, NY, AS: ACM, 2016, hlm.928–931.

[33] R. van Tonder dan CL Goues, “Menuju s/engineer/bot: Prinsip-prinsip bot perbaikan program,” dalam Prosiding Lokakarya Internasional ke-1 tentang Bot dalam Rekayasa Perangkat Lunak, ser. BotSE '19. IEEE Press, 2019, hal. 43–47.

[34] M. Wessel dan I. Steinmacher, “Sisi yang tidak nyaman dari bot perangkat lunak pada permintaan tarik,” dalam Prosiding Konferensi Internasional IEEE/ACM ke-42 tentang Lokakarya Rekayasa Perangkat Lunak, ser. ICSEW'20. New York, NY, AS: Asosiasi Mesin Komputasi, 2020, hal. 51–55.

[35] P. Duvall, S. Matyas, P. Duvall, dan A. Glover, Integrasi Berkelanjutan: Meningkatkan Kualitas Perangkat Lunak dan Mengurangi Risiko, ser. Buku tanda tangan Martin Fowler. Addison-Wesley, 2007.

[36] Z. Hu dan E. Gehringer, “Gunakan bot untuk meningkatkan umpan balik permintaan tarik GitHub,” dalam Prosiding Simposium Teknis ACM ke-50 tentang Pendidikan Ilmu Komputer, ser. SIGCSE '19. New York, NY, AS: Asosiasi Mesin Komputasi, 2019, hal. 1262–1263.

[37] S.-K. Chen, WK Fuchs, dan J.-Y. Chung, “Debug reversibel menggunakan instrumentasi program,” tindakan IEEE Trans pada Rekayasa Perangkat Lunak, vol. 27, tidak. 8, hal.715–727, 2001.