Kompatibilitas Metode Design Sprint dengan Kerangka Kerja Scrum dan Kinerja Pengembang yang Membangun Website "Kerjayuk" Untuk Mahasiswa Universitas Telkom

Muhammad Sulthan Angka Kurniawan[1], Dana Sulistyo Kusomo[2], Jati Hiliamsyah Husen[3]

[1,2,3] Universitas Telkom, Bandung

[1] **sulthanangka@student.telkomuniversity.ac.id,** [2]**danakusumo@telkomuniversity.ac.id,** [3]**jatihusen@telkomuniversity.ac.id**

Abstrak

Metode *Design* Sprint dan Kerangka kerja Scrum mengalami peningkatan popularitas sebagai Metode dan Kerangka Kerja yang banyak digunakan oleh perusahaan besar maupun perusahaan yang baru merintis usaha. Hal tersebut memicu penelitian ini untuk mengetahui kemampuan penggabungan metode dan kerangka kerja seperti Design Sprint (DS) dan Scrum yang diharapkan dapat meningkatkan tingkat keberhasilan pengembangan perangkat lunak. Penelitian ini mengamati penerapan Design Sprint yang akan digunakan ketika stakeholder dan kelompok pengembang merumuskan rincian perangkat lunak yang ingin dibangun dan Scrum akan digunakan ketika membangun perangkat lunak secara adaptif dengan kebutuhan. Analisis dilakukan dengan melakukan survei kepada tim internal dan perhitungan menggunakan metrik *Software Development Performance Index* (SDPI). Penelitian ini menemukan bahwa kompatibilitas kedua metode dapat dinyatakan layak dan performa pengembang ketika menggabungkan kedua metode dapat dinyatakan bagus. Pernyataan tersebut didukung dengan hasil penelitian dengan survei pada tim pengembang dan penelitian yang menggunakan metrik untuk mengukur stabilitas dan dedikasi tim pengembang.

Kata Kunci: ***Website, Kerjayuk, Design Sprint, Scrum.***

Abstract

The Design Sprint Method and the Scrum Framework are gaining popularity as the Methods and Frameworks that are widely used by large companies as well as start-up companies. This triggered this research to find out the capabilities of integration and frameworks such as Design Sprint (DS) and Scrum which are expected to increase the level of software development. This study observes when the implementation of the Design Sprint will be used by stakeholders and the developer group formulates the details of the software to be built and Scrum will be used when the software is adaptive to needs. The analysis is carried out by conducting a survey to the internal team and calculating using *Software Development Performance Index* (SDPI) metrics. This research finds that the second method can be declared feasible and the developer's performance when combining the two methods can be stated as good. This statement is supported by the results of qualitative research with a survey on the development team and quantitative which uses metrics to measure the dedication and dedication of the development team.

Keywords: ***Website, Kerjayuk, Design Sprint, Scrum.***

1. Pendahuluan

Latar Belakang

Penelitian Standish pada tahun 1995 menunjukkan 31,1% proyek yang sulit akan dibatalkan sebelum selesai. hasil lebih lanjut menunjukkan 52,7% proyek akan menelan biaya 189% dari perkiraan semula. Pada sisi keberhasilan, rata-rata hanya 16,2% untuk proyek perangkat lunak yang diselesaikan tepat waktu dan sesuai anggaran. Di perusahaan yang lebih besar, hanya 9% dari proyek mereka yang datang tepat waktu dan sesuai anggaran [1]. Hal ini memicu pencarian metode dan kerangka kerja yang diklaim dapat meningkatkan tingkat keberhasilan pengembangan perangkat lunak seperti Design Sprint (DS) dan Scrum.

Design Sprint (DS) adalah proses unik Google Venture yang dilakukan selama lima hari yang

digunakan untuk menyelesaikannya masalah kritis melalui pembuatan prototipe dan curah pendapat dengan klien. Semua ide terbaik dipadatkan dan diatur dalam waktu singkat untuk menciptakan ide yang sangat bagus. Dalam proses ini, deskripsi langkah demi langkah tentang apa yang harus dilakukan dalam setiap lima hari ini diberikan secara rinci [2].

Scrum adalah kerangka kerja yang berulang dan bertahap untuk pengembangan proyek dan produk atau aplikasi. Ia menyusun perkembangan dalam siklus kerja yang disebut Sprint. Ini adalah suatu iterasi yang masing-masing tidak lebih dari satu bulan, dan berlangsung satu-satu tanpa jeda. Sprint diberi batas waktu dan akan berakhir pada tanggal tertentu. Apakah pekerjaan telah selesai atau belum, sprint tidak pernah diperpanjang. Di awal setiap Sprint, tim memilih item (persyaratan klien) dari daftar yang diprioritaskan. Tim berkomitmen untuk menyelesaikan item pada akhir dari Sprint. Selama Sprint, item yang dipilih tidak berubah [3].

Tugas akhir ini akan menganalisis Kompatibilitas Metode Design Sprint dan Scrum ketika dikolaborasikan dan performa pengembang ketika menerapkan Design Sprint dan Scrum ketika membangun perangkat lunak. Metode Design Sprint dan Kerangka Kerja Scrum dipilih sebagai objek penelitian karena kedua metode tersebut merupakan metode kontemporer yang secara relatif baru dikembangkan sehingga perlu diketahui lebih lanjut potensi, kelebihan, dan kekurangan yang dimiliki. Penelitian ini akan melibatkan pemegang kepentingan dan kelompok pengembang merumuskan rincian perangkat lunak yang ingin dibangun menggunakan Design Sprint dan Scrum akan digunakan ketika membangun perangkat lunak secara adaptif.

Perumusan Masalah

a. Bagaimana Kompatibilitas Metode Design Sprint dengan Kerangka Kerja Scrum untuk Membangun Perangkat Lunak?
b. Bagaimana Performa Pengembang Dalam Penggabungan Design Sprint dan Scrum dalam membangun Perangkat Lunak?

Tujuan

a. Menjabarkan Kompatibilitas Metode Design Sprint dengan Kerangka Kerja Scrum untuk Membangun Perangkat Lunak.
b. Menjabarkan Performa Penggabungan Design Sprint dan Scrum dalam membangun Perangkat Lunak.

Batasan Masalah

a. Pembahasan difokuskan pada penerapan pengembangan perangkat lunak menggunakan *design-sprint* dan *Scrum.*
b. Tidak meneliti bahasa pemrograman dan tools yang digunakan pengembang perangkat lunak. Hal ini bertujuan agar pokok penelitian dapat lebih mendalam. Oleh sebab itu, telah ditentukan bahwa tools yang akan digunakan adalah XAMPP, Github Desktop, dan VS Code. Bahasa Pemrograman dan Kerangka Kerja yang digunakan adalah PHP dan Laravel.
c. Perangkat lunak akan dibangun hingga tahap *production*, tidak akan menganalisis business outcome. Hal ini dikarenakan penelitian ini memiliki tujuan untuk meneliti proses pengembangan.
d. Penelitian ini tidak menyertakan modul *budgeting* untuk memfokuskan pada proses pengembangan perangkat lunak.
e. Proses pencarian sumber daya manusia tidak diteliti agar bisa berfokus pada proses pengembangan perangkat lunak.

Organisasi Penulisan

Penulisan Tugas Akhir ini disusun dalam beberapa bagian, yaitu: Bagian 1 – Pendahuluan, Bagian 2 – Kajian Pustaka, Bagian 3 – Metodologi Penelitian, Bagian 4 – Hasil dan Analisis Penelitian, dan Bagian 5 – Kesimpulan.

2. Studi Terkait

Dalam penelitian ini, dibutuhkan dukungan-dukungan penelitian sebelumnya yang berkaitan. Penelitian-penelitian tersebut merupakan metode yang digunakan dalam penelitian ini mengenai kompatibilitas antara metode *Design Sprint* dengan kerangka kerja Scrum dan mengenai performa

pengembang yang menggunakan *Design Sprint* dan Scrum.

Kompatibilitas pada penelitian ini memiliki arti sebagai kecocokan metode dan kerangka kerja tersebut apabila digunakan Bersama. Pada penelitian ini diteliti bagaimana proses Langkah Langkah pada metode dan kerangka kerja apakah setiap proses berjalan lancar, memiliki kendala, maupun dapat menghalangi proses satu sama lain apabila digunakan.

Performa pengembang pada penelitian ini memiliki arti sebagai nilai yang diberikan berdasarkan kinerja tiap anggota tim dalam membangun perangkat lunak. Pada tahapan ini terdapat penilaian yang berupa stabilitas tim dan dedikasi tiap anggota.

2.1 Design Sprint

Design Sprint adalah proses lima hari yang unik yang dibuat oleh Google Venture untuk memecahkan masalah kritis melalui prototipe dan bertukar pikiran dengan klien [2]. metodologi dan penerapannya meliputi kegiatan-kegiatan berikut:

2.1.1 Mempersiapkan Lingkungan

Sebelum Sprint dimulai perlu dibentuknya sebuah tim, lalu tentukan ruang, waktu, dan tujuan yang akan diraih tim tersebut. Seperti yang disebutkan sebelumnya, tujuan yang jelas sangat diperlukan sebelum memulai lima hari Sprint dalam untuk menghasilkan prototipe yang akan menghasilkan produk yang sukses. Selain tujuan, perlu dibentuk sebuah tim yang terdiri dari maksimum tujuh orang dan ruang yang nyaman bagi semua orang untuk melaksanakan Sprint [2].

2.1.2 Merumuskan Permasalahan

Sprint berguna dalam situasi menantang yang berisiko tinggi. Sprint adalah saat yang tepat untuk memeriksa ide awal dan mengubah arah dalam jangka waktu singkat. Sprint juga cocok untuk digunakan ketika tim hanya memiliki waktu pendek untuk menguji hasil proyek (prototipe).

Sprint merupakan metode yang berfokus dalam menemukan solusi yang bagus dalam waktu singkat. Hal yang penting agar dapat mencapai hal tersebut adalah dengan memfokuskan sumber daya kepada hal-hal yang besar. Hal ini dikarenakan pokok permasalahan yang besar inilah yang akan menarik klien [2].

2.1.3 Membagi Permasalahan pada Tim

Agar Sprint berpeluang sukses, disarankan untuk menyiapkannya sebuah tim yang terdiri dari 7 orang atau kurang. Pada tim tersebut, akan sangat baik apabila anggotanya merupakan Definer, Spesialis Keuangan, Spesialis Pemasaran, Spesialis Konsumen, Spesialis Teknologi, dan Spesialis Desain.

Saat memilih tim, penting untuk memilih orang-orang dengan relasi yang baik antara satu sama lain, karena mereka akan bekerja sama penuh waktu selama 5 hari. Selain itu, tim tersebut juga membutuhkan seseorang yang secara alami merupakan seorang "Pembuat masalah". Hal ini dikarenakan biasanya orang-orang ini cerdas dan melihat masalah secara berbeda dari siapapun. Jika 7 orang dirasa terlalu sedikit untuk menjalankan proyek, dapat pula disertakan pakar lain untuk dimintai konsultasi pada awal sprint.

Waktu yang tepat untuk masa konsultasi ini adalah Senin sore. Selama kunjungan ini, mereka dapat memberikan tim apa saja yang mereka ketahui dan memberikan pendapat mereka. Ketika tim sudah dirakit, saatnya untuk menentukan fasilitator.

Seorang fasilitator adalah orang yang mengelola waktu, debat, dan proses secara umum dengan sebaik mungkin. Mereka merupakan orang yang punya pengalaman dalam melakukan pertemuan dan tahu bagaimana menengahi diskusi, memaksakan waktu untuk berhenti dan beralih ke topik lain. Fasilitator tidak boleh memihak tentang keputusan. Spesialis yang termasuk dalam tim harus berasal dari area dan posisi yang berbeda karena masing-masing akan membuat kontribusi penting, baik itu dengan informasi dasar, ide baru, atau bahkan informasi berguna tentang visi klien [2].

2.1.4 Waktu dan Tempat

Untuk melakukan Desain Sprint selama seminggu, kesediaan tim untuk mengabdikan seluruh waktu mereka untuk berdedikasi penuh sangat diperlukan. Dengan demikian, aspek terbaik dari Desain Sprint akan mungkin dicapai, yaitu kebebasan untuk bekerja seperti yang diinginkan, dengan jadwal tanpa kompromi dan tercapainya tujuan yang menantang.

Pada hari-hari biasa dalam sprint, tim mencurahkan 6 jam untuk berada di ruangan yang sama dari jam 10 pagi sampai jam 5 sore, Senin sampai Kamis. Hari Jumat akan didedikasikan untuk pengujian dengan pengguna. Perangkat elektronik dilarang di sprint kamar. Hal ini dikarenakan Ada penelitian

yang membuktikan bahwa seseorang memiliki pikirannya sendiri terputus, perlu beberapa saat untuk kembali ke titik pemikiran yang sama. Sehingga larangan elektronik berguna di ruang sprint.

Penting untuk mengadakan Sprint di ruangan yang sama sepanjang minggu, sehingga papan tulis dan aset lainnya tetap terjaga dan sinkron dengan Sprint selama seminggu. Hal ini sangat membantu karena aset tersebut dapat menyimpan ide terbaik dan memeriksa apakah fokus pekerjaan tidak menjauhkan diri dari ide utama. Papan putih paling baik untuk melihat tujuan ini [2].

2.2 Scrum

Scrum adalah kerangka kerja berulang dan bertahap untuk pengembangan proyek dan produk atau aplikasi. Ia menyusun perkembangan dalam siklus kerja yang disebut Sprint. Ini adalah iterasi masing-masing tidak lebih dari satu bulan, dan berlangsung satu demi satu tanpa jeda. Sprint diberi batas waktu dan akan berakhir pada tanggal tertentu, tanpa melihat apakah pekerjaan telah selesai atau belum. Di awal setiap Sprint, anggota tim memilih item (persyaratan pelanggan) dari daftar yang diprioritaskan. Tim berkomitmen untuk menyelesaikan item pada akhir dari Sprint. Selama Sprint, item yang dipilih tidak berubah. Setiap hari tim berkumpul sebentar untuk memeriksa kemajuannya, dan menyesuaikan langkah selanjutnya yang diperlukan untuk menyelesaikan pekerjaan yang tersisa.

Di akhir Sprint, tim meninjau Sprint dengan pemangku kepentingan, dan mendemonstrasikan apa yang telah dibangunnya. Anggota saling mendapat tanggapan yang dapat diimplementasikan pada sprint berikutnya. Scrum menekankan pada hasil kerja di akhir Sprint yang benar-benar “selesai”. dalam kasus perangkat lunak, ini berarti kode yang terintegrasi, diuji sepenuhnya, dan berpotensi dapat dikirim. Tema utama dalam Scrum adalah "memeriksa dan beradaptasi". Karena pembangunan pasti melibatkan pembelajaran, inovasi, dan hal hal yang tidak terduga, Scrum menekankan pada langkah pengembangan yang singkat, langsung memeriksa produk yang dihasilkan, dan mengamati kemanjuran praktik saat ini. Setelah melakukan hal tersebut, perhatikan kembali tujuan produk lalu ulangi pengerjaan apabila masih ada yang perlu diperbaiki [3].

2.2.1 Peran pada Scrum

Di Scrum, ada tiga peran: Pemilik Produk, Tim, dan Scrum Master. Bersama-sama, ketiga ini dikenal sebagai Tim Scrum.

2.2.2 Memulai Scrum

Langkah pertama dalam Scrum adalah agar Pemilik Produk mengartikulasikan visi produk. Pada akhirnya, visi ini berkembang menjadi daftar fitur yang disempurnakan dan diprioritaskan yang disebut Product Backlog. Backlog ini akan ada (dan berkembang) selama masa keberlangsungan produk. Pada titik mana pun, Product Backlog adalah satu-satunya tampilan definitif dari segala sesuatu yang bisa dilakukan oleh Tim dalam urutan prioritas. Hanya akan ada satu Product Backlog. Ini berarti Pemilik Produk harus membuat keputusan mewakili kepentingan pemangku kepentingan dan dipengaruhi oleh tim.

Product Backlog terus diperbarui oleh Pemilik Produk untuk mencerminkan perubahan di kebutuhan pelanggan, ide-ide baru, atau wawasan. Perubahan Backlog digerakkan oleh persaingan, rintangan teknis, dan kebutuhan lain. Tim perlu memberikan perkiraan sumber daya yang dibutuhkan kepada Pemilik Produk untuk setiap item di Product Backlog. Hal itu dilakukan agar Pemilik Produk yang bertanggung jawab untuk menetapkan perkiraan nilai bisnis untuk setiap item mengerti akan sumber daya yang dibutuhkan.

Dengan mempertimbangkan usaha, nilai, dan resiko, Pemilik Produk memprioritaskan backlog untuk memaksimalkan ROI (memilih item bernilai tertinggi dengan usaha terendah). Perkiraan upaya dan nilai ini mungkin akan diperbaharui setiap rapat Sprint. Pembaharuan backlog ini adalah kegiatan ulang yang berkelanjutan aktivitasnya karena Product Backlog akan terus berkembang.

Seiring waktu, Tim melacak seberapa banyak pekerjaan yang dapat dilakukannya setiap Sprint. misalnya, rata-rata 26 poin per Sprint. Dengan informasi ini mereka dapat memproyeksikan tanggal rilis untuk melengkapi semua fitur, atau berapa banyak fitur yang dapat diselesaikan dengan tanggal tetap, jika rata-rata berlanjut dan tidak ada perubahan. Rata-rata ini disebut "velocity" tim. “Velocity” diekspresikan sebagai perkiraan ukuran item Product Backlog.

Item dalam Product Backlog dapat bervariasi dalam ukuran atau upaya. Item yang lebih besar dapat dipecah menjadi item-item yang lebih kecil selama lokakarya Product Backlog Refinement atau saat rapat perencanaan Sprint. Item Product Backlog untuk Sprint yang akan datang harus berukuran kecil dan dipahami oleh Tim. Hal ini ditujukan agar ukuran item memungkinkan komitmen yang dibuat dalam rapat Perencanaan Sprint menjadi berarti. Ini disebut ukuran yang "actionable" [3].

2.2.3 Merencanakan Scrum

Di awal setiap Sprint, Rapat Perencanaan Sprint berlangsung. Ini dibagi menjadi dua sub-rapat bagian satu dan sub-rapat bagian dua.

2.2.3.1 Perancangan Sprint Bagian Pertama

Sprint Planning Bagian Pertama, Product Owner dan Tim (dengan fasilitasi dari Scrum Master) mereview item prioritas tinggi di Product Backlog. Diskusi ini meliputi pembahasan tujuan, konteks item, dan memberikan tim wawasan tentang pemikiran Product Owner. Product Owner dan Tim juga meninjau "Definition of Done" (DOD) yang didirikan sebelumnya. DOD dapat ditentukan sendiri oleh Tim dan Product Owner, namun DOD yang baik biasanya ditentukan ketika produk sesuai dengan standar, telah ditinjau, diimplementasikan dengan unit test-driven development (TDD), diuji dengan uji 100 persen otomatisasi, terintegrasi, dan didokumentasikan. Bagian Satu Perancangan Sprint berfokus pada pemahaman tentang keinginan Product Owner [3].

2.2.3.2 Perancangan Sprint Bagian Kedua

Perencanaan Sprint Bagian kedua berfokus pada bagaimana mengimplementasikan item yang dipilih oleh Tim. Tim memilih item dari Product Backlog lalu berkomitmen untuk menyelesaikannya pada akhir Sprint. Memilih Item dimulai dari bagian atas Product Backlog (dalam kata lain, dimulai dengan item yang merupakan prioritas tertinggi untuk Pemilik Produk).

Tim memutuskan seberapa banyak pekerjaan yang akan diselesaikan, lalu mengambil tugas yang diberikan Product Owner. Hal ini membuat komitmen lebih dapat diandalkan karena Tim membuatnya berdasarkan kemampuan, analisis, dan perencanaan sendiri daripada diputuskan oleh orang lain. Meskipun Product Owner tidak memiliki kendali atas seberapa besar komitmen Tim, dia tahu bahwa item yang akan dikerjakan oleh Tim diambil dari atas Product Backlog yang dinilai sebagai yang paling penting [3].

2.2.4 Keseharian Dalam Scrum

Setelah Sprint dimulai, Tim akan melakukan praktik utama Scrum Harian. Ini adalah rapat singkat (15 menit atau kurang) yang dilakukan setiap hari kerja pada waktu yang ditentukan. Semua orang di Tim hadir. Untuk membuatnya tetap singkat, disarankan agar semua orang tetap berdiri. Ini adalah kesempatan Tim untuk menyinkronkan pekerjaan, laporan, dan mengemukakan rintangan mereka. Dalam rapat harian Scrum, setiap anggota Tim melapor tiga (dan hanya tiga) hal kepada anggota Tim yang lain:

1. Apa yang dapat mereka selesaikan sejak pertemuan terakhir
2. Apa yang mereka rencanakan untuk diselesaikan pada pertemuan berikutnya
3. Apa saja rintangan atau penghalang yang menghalangi mereka

Perhatikan bahwa rapat harian Scrum bukanlah rapat status untuk melaporkan kepada manajer. ini adalah waktu bagi Tim yang mengatur dirinya sendiri untuk saling berbagi dan untuk membantu mereka berkoordinasi. Ketika rapat ini diselenggarakan, anggota menyampaikan catatan mereka yang berisi masalah yang dihadapi dan Scrum Master bertanggung jawab untuk membantu anggota Tim menyelesaikannya. Tidak ada diskusi selama rapat harian Scrum, melainkan hanya melaporkan jawaban dari tiga pertanyaan. jika diskusi diperlukan, itu terjadi segera setelah rapat harian Scrum dalam pertemuan lanjutan. Pertemuan lanjut ini adalah saatnya Tim untuk menggunakan informasi yang mereka dapatkan di rapat harian Scrum.

Umumnya, tidak disarankan untuk menjalani Daily Scrum dengan manajer atau orang lain dalam posisi yang lebih tinggi menghadiri Daily Scrum. Risiko ini membuat Tim merasa diawasi di bawah tekanan untuk melaporkan kemajuan besar setiap hari (harapan yang tidak realistis). Rapat harian bersama atasan juga dapat menghambat diskusi masalah sehingga dapat merusak manajemen Tim [3].

2.2.5 Durasi Sprint

Salah satu prinsip utama Scrum adalah durasi Sprint tidak pernah diperpanjang. Sprint berakhir pada tanggal penugasan terlepas dari apakah Tim telah menyelesaikan pekerjaan yang menjadi komitmennya. Sebuah Tim biasanya melakukan over-commit pada beberapa Sprint pertamanya dan gagal untuk menyelesaikannya komitmen. Namun, pada sprint ketiga atau keempat Tim biasanya telah mengetahui kapasitas yang mampu mereka kerjakan apa yang mampu mereka kerjakan dan mereka akan memenuhi tujuan Sprint mereka dengan lebih andal setelah itu.

Tim akan didorong untuk memilih satu durasi untuk Sprint mereka (katakanlah, dua minggu) dan tidak mengubahnya. Ini membantu Tim mempelajari seberapa banyak yang dapat dicapai, membantu dalam estimasi, dan perencanaan rilis jangka panjang. Ini juga membantu Tim mencapai ritme untuk pekerjaan mereka. ini sering disebut sebagai "detak jantung" Tim di Scrum [3].

2.4 Software Development Performance Index (SDPI)

Kerangka kerja Indeks Kinerja Pengembangan Perangkat Lunak (SDPI) menyusun serangkaian ukuran yang mengukur hasil pengembangan suatu perangkat lunak. SDPI mampu memberikan umpan balik mengenai kinerja anggota tim pengembang pada suatu organisasi [4]. Pengukuran ini digunakan agar penelitian memiliki acuan yang dapat diukur secara eksak sehingga dapat memperoleh kesimpulan.

2.4.1 Percent Dedicated Work

Pengukuran ini menunjukkan seberapa banyak pekerjaan untuk tim tertentu yang dilakukan oleh orang-orang yang berdedikasi pada tim tersebut.

2.4.1.1 Rumus

1. Temukan semua transaksi (snapshot) untuk cerita, cacat, dan tugas yang sedang berlangsung, tidak memiliki anak, memiliki pemilik (pengguna), dan tidak diblokir.
2. Jumlahkan semua transaksi berdasarkan total U pengguna, total proyek P, dan kontribusi pengguna ke proyek U_{total} di mana U_{total} > 5 (misalnya, pengguna dengan jumlah transaksi total kurang dari atau sama dengan 5 tidak dihitung terhadap proyek U atau P total).
3. Temukan persentase dari total pekerjaan pengguna yang diwakili oleh setiap proyek: $U_{\text{percent}} = \frac{U_{project}}{U_{total}} . 100$
4. Hitung sebagai didedikasikan untuk proyek tertentu pengguna yang $U_{percent}$ lebih besar dari 70% untuk proyek itu. Ambang batas ini ditentukan oleh eksperimen dengan kumpulan data pelatihan dari tim dengan anggota yang dikenal dan berdedikasi.
5. Untuk setiap proyek, jumlahkan transaksi pengguna khusus: $P_{dedicated} = \sum U_{project}$, untuk semua anggota khusus.
6. Temukan persentase pekerjaan khusus untuk setiap proyek: $P_{percentdedicated} = \frac{P_{dedicated}}{P_{total}} . 100$

[4]

2.4.2 Team Stability

Team Stability merupakan indikasi stabilitas tim.

2.4.2.1 Rumus

1. Temukan semua transaksi (snapshot) untuk cerita, cacat, dan tugas yang sedang berlangsung, tidak memiliki anak, memiliki pemilik (pengguna), dan tidak diblokir.
2. Jumlahkan semua transaksi berdasarkan total U pengguna , total proyek P , dan kontribusi pengguna ke proyek U_{total} di mana U_{total} > 5 (misalnya, pengguna dengan jumlah total transaksi kurang dari atau sama dengan 5 tidak dihitung terhadap proyek U atau P total).
3. Temukan fraksi dari total pekerjaan pengguna yang diwakili setiap proyek untuk semua periode waktu: $Ufte = \frac{Uproject}{Utotal}$
4. Jumlahkan ekuivalen waktu penuh untuk setiap proyek untuk semua periode waktu: $P_{fte} = \sum U_{fte}$.
5. Untuk setiap proyek dan setiap pasangan periode waktu yang berdekatan (t dan t-1) hitung:

   Team growth: $\frac{\sum \max(0, U_{fte,t} - U_{fte,t-1}}{P_{fte,t}} . 100$

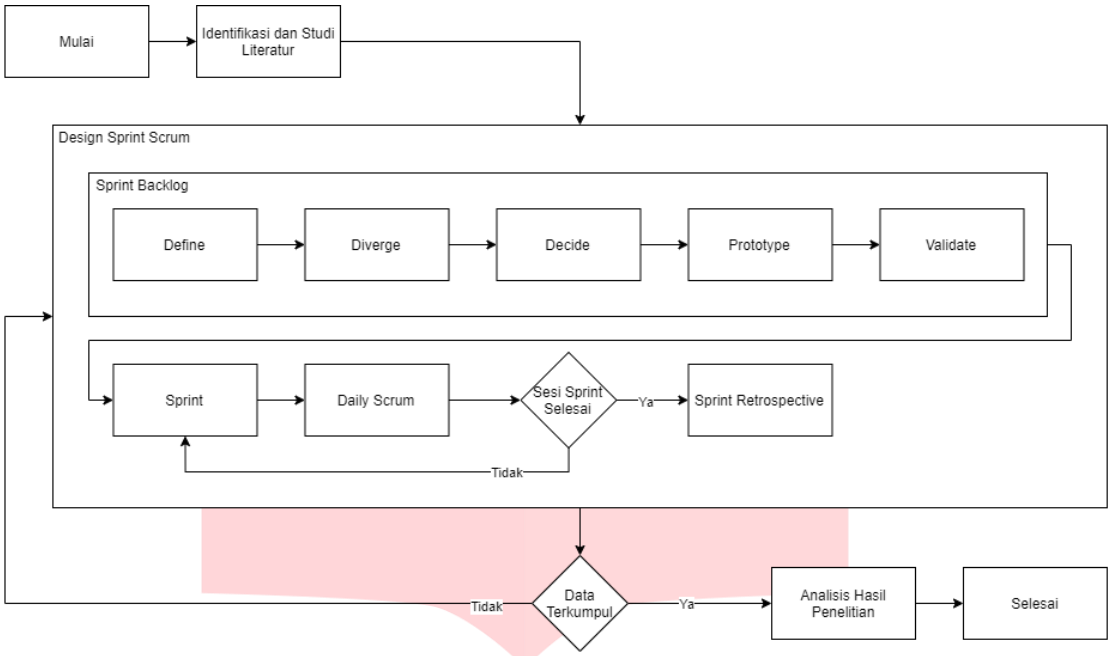
   Team shrinkage: $\frac{\sum \max(0, U_{fte,t-1} - U_{fte,t}}{P_{fte,t-1}} . 100$

   Team stability: $100 - \frac{P_{growth} + Pshrinkage}{2}$

[4]

3. Metodologi Penelitian

Gambar 3. Alur Metodologi Penelitian



Secara umum, alur penelitian dimulai dari identifikasi dan studi literatur. Selanjutnya menuju pada tahap Design Sprint, yang merupakan fase untuk mencari requirement dan user story yang dibutuhkan untuk membangun perangkat lunak. Tahap selanjutnya ada Scrum, yang merupakan fase untuk pembuatan perangkat lunak berdasarkan requirement dan user story yang didapatkan pada fase sebelumnya. Selanjutnya terdapat dua opsi, yaitu lanjut ke analisis hasil penelitian atau mengulangi fase Scrum maupun Kembali ke Design Sprint. Terakhir adalah tahap analisis hasil penelitian ketika data telah didapatkan pada fase-fase sebelumnya.

3.1.1 Identifikasi dan Studi Literatur

Tahap ini adalah saat dimana peneliti melakukan penerapan proses identifikasi masalah, perumusan masalah, Batasan masalah, serta tujuan penelitian. Setelah melakukan identifikasi, peneliti melakukan Studi Literatur dimana peneliti mencari informasi yang dibutuhkan dalam penelitian ini dengan cara membaca paper mengenai Desain Sprint dan Scrum.

3.1.2 Design Sprint

Dalam tahap perancangan ini, aktivitas yang dilakukan terdapat dalam tahapan metode Design Sprint. Tahapan-tahapan tersebut yaitu: Define, Diverge, Decide, Prototype, dan Validate [2]. Apabila dalam aktivitas Validate hasil prototype tersebut terdapat kebutuhan yang masih belum terpenuhi, maka Aktivitas dapat diulang.

3.1.3 Scrum

Scrum dilakukan setelah perancangan kebutuhan selesai dengan menggunakan metode Design Sprint selanjutnya akan diimplementasikan menggunakan Kerangka Kerja Scrum. Pada tahap ini, prototype yang sudah dibentuk dan di uji sebelumnya akan dikembangkan menjadi perangkat lunak yang dapat digunakan [3].

3.1.4 Analisis Hasil Penelitian

Tahap ini merupakan tahap dimana pengolahan data yang telah didapatkan pada tahapan sebelumnya untuk mendapatkan jawaban dari rumusan masalah.

4. Hasil dan Analisis Penelitian

4.1 Kompatibilitas Kedua Metode

Kompatibilitas Metode Design Sprint dengan Kerangka Kerja Scrum perlu diteliti bagian mana yang dapat digabungkan, oleh sebab itu analisis penggabungan Design Sprint pada Scrum dilakukan. Setelah mengetahui bagian mana yang dapat digabung, penelitian yang melibatkan anggota tim pengembang untuk memilih jawaban pada kuesioner yang berkaitan dengan kompatibilitas Design Sprint dan Scrum berdasarkan pengalaman mereka selama membangun perangkat lunak. Selain penggunaan kuesioner, analisis proses Design Sprint dan Scrum juga dilakukan untuk menjabarkan secara rinci proses mana yang berkaitan, mendukung, dan bertentangan antara satu sama lain.

4.1.2 Analisis Penggabungan Design Sprint pada Scrum

Gambar 3 menunjukan bahwa Design Sprint diintegrasikan pada Scrum di bagian *Sprint Backlog*. Oleh sebab itu, analisis ini dilakukan untuk menggambarkan proses pembangunan perangkat lunak yang hanya menggunakan Scrum dengan Design Sprint + Scrum.

Tabel 1. Analisis Penggabungan Design Sprint pada Scrum

Scrum	**Design Sprint + Scrum**
Pembuatan *Sprint backlog* tidak memiliki ketentuan khusus sehingga menggunakan skema normal scrum.	Pembuatan *Sprint backlog* dilakukan sesuai tahapan Design Sprint.
Perancangan *Sprint backlog* Tidak memiliki tenggat waktu yang ditentukan. Biasanya perancangan *Sprint backlog* dapat dapat dilakukan dalam satu hari.	Dirancang bahwa proses pembuatan *Sprint backlog* yang mengintegrasi dengan Design Sprint memakan waktu lima hari.
Pada perumusan *Sprint backlog* yang dilakukan adalah merumuskan permasalahan, dan merancang apa yang akan dilakukan saat *Scrum Sprint.* Sehingga, membuat user story dan membuat prototype dilakukan pada *Scrum Sprint*.	Pada akhir sesi Design Sprint (*Sprint Backlog* pada proses Scrum) pengembang sudah memiliki user story yang lengkap dengan prototype.

4.1.1 Kuesioner

Kuesioner ini merupakan kumpulan pertanyaan yang diberikan kepada pengembang mengenai Desain Sprint, Scrum, dan Penggabungan kedua metode tersebut. Kuesioner ini diberikan khusus kepada tim pengembang dikarenakan merekalah yang menjadi objek penelitian untuk mengetahui kompatibilitas kedua metode. Penelitian ini dapat dijustifikasi dengan penelitian serupa [5] [6].

Berdasarkan data yang didapatkan melalui kuesioner (lampiran 2), dapat diketahui bahwa pendapat pengambang terhadap metode *Desain Sprint, Scrum,* dan penggabungan kedua metode memiliki variasi. Namun, dengan mencari rata rata dari pendapat ini dapat bahwa:

- *Design Sprint*
  - mempermudah tim untuk menghasilkan rumusan requirement dengan baik
  - mengurangi volatilitas perombakan requirement
- *Scrum*
  - mengurangi miskomunikasi antar anggota ketika membangun perangkat lunak
  - membuat tahapan pembangunan perangkat lunak lebih jelas
- Penggabungan dua metode
  - layak untuk digunakan oleh pengembang
  - meningkatkan ketepatan perancangan dan pembangunan perangkat lunak

Untuk mencegah menyimpulkan hasil yang bias, hasil di atas telah diperlihatkan kepada partisipan yang mengikuti kuesioner untuk mendapatkan pendapat mereka mengenai hasil yang disimpulkan. Selain itu, analisis cakupan yang dibahas berikut ini digunakan untuk mempertimbangkan kesimpulan terhadap penggabungan Design Sprint dan Scrum.

4.1.2 Analisis Cakupan *Design Sprint* dengan Scrum

Tujuan analisis ini adalah untuk menentukan area proses *Design Sprint* dan Scrum mana yang saling mendukung, di mana penyesuaian perlu dilakukan dan area proses mana yang berkonflik. Sistem pemeringkatan yang mereferensi jurnal [7] digunakan untuk mengukur kompatibilitas.

- *Conflicting* (-)
- *Not addressed* (0)
- *Partially supported (+)*
- *Supported (++)*
- *Largely supported (+++)*

"*Largely supported*" menandakan jika digunakan dengan benar maka kedua metode akan memenuhi bagian utama dari komponen model masing-masing. "Supported" dan "Partially supported" menggambarkan cakupan terbatas dan "Not addressed" mencerminkan bahwa tidak ada cakupan sama

sekali. Peringkat ini tidak mengindikasi bahwa penggabungan metode tidak bisa dilakukan, melainkan hanya menunjukkan bahwa praktik tambahan harus digunakan untuk sepenuhnya digunakan dengan optimal. "Conflicting" di sisi lain menunjukkan bahwa masing-masing metode tidak dapat dipadukan.

1. Penjatahan tugas anggota tim (++)
    - *Design Sprint* dan Scrum memberikan tanggung jawab kepada anggota tim tertentu. Selain itu, pengembang bertanggung jawab atas tugas-tugas tertentu selama proyek berlangsung.
2. Identifikasi dan keterlibatan pemangku kepentingan (+++)
    - *Design Sprint* dan Scrum melibatkan pemangku kepentingan dari tim ahli hingga pemilik produk. Oleh sebab itu, tidak ada kontradiksi dari keanggotaan.
3. Perancangan kebutuhan (+)
    - Pemahaman tentang kebutuhan produk diperoleh melalui kolaborasi pelanggan bersama tim dan komunikasi intensif yang dihasilkan. Perubahan persyaratan dengan cepat didiskusikan. Meskipun ketertelusuran persyaratan bukan merupakan tujuan eksplisit *Design Sprint* dan Scrum, hal ini didukung oleh *user story*, *task log*, dan *functional test* yang mendeteksi ketidakkonsistenan antara pekerjaan proyek, kebutuhan produk, dan *unit test*.
4. Pemantauan dan pengendalian proyek (++)
    - Pada Scrum, jadwal dan estimasi pengerjaan dipantau oleh Scrum Master. Informasi tentang kemajuan proyek dikumpulkan dengan menggunakan langkah-langkah. Komunikasi yang intensif antara anggota tim, pemilik produk, dan dengan keterlibatan calon pelanggan. Tonggak pencapaian diperiksa terhadap jadwal dengan tes fungsional. Sistem iterasi pendek yang ketat dan komitmen reguler terhadap rencana memudahkan pemantauan proyek.
    - Pada *Design Sprint*, apa yang dilakukan pada poin pertama juga dilakukan dan melibatkan personel yang relatif sama, namun yang berbeda hanyalah task pokok. *Design Sprint* berfokus pada perancangan kebutuhan dan pembentukan *user story.*
5. Pengembangan kebutuhan (+++)
    - Pelanggan menggagaskan kebutuhan produk dijadikan sebagai *user*. Kebutuhan produk ini biasanya bersifat umum, oleh sebab itu detail harus didiskusikan langsung dengan pelanggan selama pengembangan. Kebutuhan produk yang berubah dan penggunaan iterasi memungkinkan analisis dan validasi persyaratan yang konstan. Konsep dan skenario operasional dibuat dengan menggunakan tes fungsional.
    - *Design Sprint* dan Scrum menerapkan skema poin pertama, sehingga integrasi kedua metode dapat dilakukan dengan lancar.
6. Proses pengembangan perangkat lunak (+)
    - Proses pengembangan perangkat lunak dilakukan pada Scrum setelah *Design Sprint* merancang kebutuhan produk. Hal ini tidak menghalangi kegiatan satu sama lain dan bersifat berulang.

Design Sprint adalah metode pencarian ide, fitur, dan *business model* Sedangkan Kerangka Kerja Scrum adalah suatu proses pengembangan produk. Dengan demikian, konsep-konsep tersebut tidak dapat secara langsung di tempel. Fokus mereka berbeda, namun tetap memiliki keterkaitan. *Design Sprint* dan Scrum tidak hanya dapat dipadukan, bahkan saling mendukung.

4.2 Performa Pengembang dalam Pembangunan

Performa Pengembang ketika menggunakan metode Design Sprint dan Scrum dapat diketahui berdasarkan hasil perangkat lunak yang dibangun. Oleh sebab itu, pengukuran dapat dilakukan menggunakan metrik Percent Dedicated Work untuk menghitung penyelesaian tugas yang diberikan anggota pengembang dan Team Stability untuk menghitung stabilitas tim. Kedua metrik tersebut dianggap cukup untuk mengukur performa karena telah mengaitkan variabel yang dibahas serta dapat memberikan indikasi yang jelas untuk menjawab permasalahan mengenai performa pengembang perangkat lunak.

4.2.1 Percent Dedicated Work

Berdasarkan data lapangan mengenai pembagian tugas dan pengerjaannya yang diolah menggunakan rumus SDPI Measurements untuk menemukan Percent Dedicated Work, ditemukan hasil dari tiap

anggota pengembang sebagai berikut.

Tabel 3. Dedikasi anggota pengembang

Nama	Tugas Selesai	Tugas Diberikan	Persentasi	Keterangan
Sulthan	23	26	88.4%	Dedicated
Arya	6	6	100%	Dedicated
Nurul	3	3	100%	Dedicated
Rian	18	18	100%	Dedicated
Naqliya	4	4	100%	Dedicated
Hema	3	3	100%	Dedicated

Tabel diatas menunjukan bahwa seluruh anggota pengembang telah mengerjakan tugas yang diberikan dengan persentase lebih dari 70%, oleh sebab itu seluruh anggota dapat dianggap berdedikasi dalam pembangunan perangkat lunak.

4.2.2 Team Stability

Berdasarkan data lapangan mengenai pembagian tugas dan pengerjaannya yang diolah menggunakan rumus SDPI Measurements untuk menemukan Team Stability, ditemukan hasil dari tiap anggota pengembang sebagai berikut.

Tabel 4. Stabilitas tim

Nama	Persentase Bulan-1	Persentase Bulan-2	Persentase Bulan-3
Sulthan	87%	100% (+13%)	88% (-12%)
Nurul	100%	100%	100%
Arya	100%	100%	100%
Naqliya	100%	100%	100%
Hema	100%	100%	100%
Rian	100%	100%	100%

Tabel diatas menunjukan persentase tugas yang telah diselesaikan tiap bulan. Dapat dilihat pada bulan ke-2 terdapat peningkatan 13% dan pada bulan ke-3 terdapat penurunan 12%. Berdasarkan pengukuran, maka team stability merupakan 100-(13+12)/2 = 87,5. Nilai ini melebihi 70 yang merupakan standar kinerja baik. Oleh sebab itu, team stability pada anggota tim ini dianggap baik.

5. Kesimpulan

Berdasarkan hasil yang didapatkan pada bab sebelumnya, maka dapat diketahui bahwa mengukur kompatibilitas metode dan performa pengembang yang menggunakan metode tersebut memerlukan komponen yang banyak. Kompatibilitas kedua metode dapat dinyatakan layak. Hal ini diperkuat dengan pendapat pengembang yang menyatakan bahwa Design Sprint mempermudah tim untuk menghasilkan rumusan requirement dengan baik dan mengurangi volatilitas perombakan requirement; Scrum mengurangi miskomunikasi antar anggota ketika membangun perangkat lunak dan membuat tahapan pembangunan perangkat lunak lebih jelas; Penggabungan dua metode layak untuk digunakan oleh pengembang dan meningkatkan ketepatan perancangan dan pembangunan perangkat lunak. Analisis proses Design Sprint dan Scrum juga memberikan hasil bahwa konsep-konsep tersebut tidak dapat secara langsung di tempel. Fokus mereka berbeda, namun tetap memiliki keterkaitan. Design Sprint dan Scrum tidak hanya dapat dipadukan, bahkan saling mendukung. Performa pengembang ketika menggabungkan kedua metode dapat dinyatakan bagus. Hal ini dapat diperkuat dengan penelitian diatas yang menunjukan bahwa dengan menggunakan rumus SDPI, percent dedicated work yang dimiliki seluruh anggota mendapatkan persentase lebih dari 70% yang memiliki makna bahwa seluruh anggota berdedikasi saat membangun perangkat lunak. Performa pengembang juga diperkuat dengan Team Stability yang memiliki nilai 87,5 yang lebih besar dari batas dedikasi. Kedua hal ini menyatakan bahwa anggota im memiliki kontribusi yang konsisten hingga akhir pembangunan perangkat lunak.

Referensi

[1] S. G. International, *The CHAOS Report,* p. 1, 1994.

[2] I. M. S. E. D. C. A. P. F. d. A. Carlos Magno Mendon¸ca de S´a Ara´ujo, "Design Thinking versus Design Sprint: A Comparative Study," *Computer Science Department - Professional Masters in Applied Computing - University of Bras´ılia (UnB),* 2019.

[3] G. B. C. L. B. V. Pete Deemer, "THE SCRUM PRIMER," pp. 5-6, 2010.

[4] Broadcom, "Software Development Performance Index (SDPI)," 2021.

[5] A. P. B. D. S. C. M. M. Carol Passos, "Analyzing the Impact of Beliefs in Software Project Practices," p. 9, 2011.

[6] M. R. Wrobel, "Towards the participant observation of emotions in software development teams," p. 4, 2016.

[7] M. F. &. P. Keil, "Agile Methods and CMMI: Compatibility or Conflict?," vol. 1, no. 1, p. 18, 2007.

[8] L. Macaulay, " Requirements Engineering," *Springer,* 1996.

[9] Y. A. a. P. B. K. Desouza, "“Managing Knowl edge in Global Software Development Efforts: Issues and Practices,”," *IEEE Software,* pp. 30-37, 2006.

[10] D. D. a. D. Zowghi, "“Requirements Engineering Challenges in Multi-Site Software Development Organizations,”," *Requirements Eng. J,* pp. 149-160, 2003.

[11] S. Overby, "“The Hidden Costs of Offshore Outsourcing,”," *CIO Magazine,* 2003.

[12] M. G. a. S. M. J. Bhat, " “Overcoming Rquirements Engineering Challenges: Lessons Learned from Offshore Outsourcing"," *IEEE Software,* pp. 38-44, 2006.

[13] D. D. a. J. Chisan, "“An Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes That Lead to Payoffs in Productivity, Quality and Risk Management,”," *J. Trans. Software Eng., vol. 32, no. 7,* pp. 433-453, 2006.

[14] D. Smite, " “Requirements Management in Distributed Projects,”," *J. Universal Knowledge Management, vol. 1, no. 2,* pp. 69-76, 2006.

[15] P. R. &. M. Höst, "Guidelines for conducting and reporting case study," *Springerlink.com,* p. 34, 2008.

[16] S. E. &. J. Aranda, "Case Studies for Software Engineer".