

---

## ANALISA PERBANDINGAN SOFTWARE DEVELOPMENT MODEL ANTARA METODE SCRUM DAN METODE INCREMENTAL

Evi Dwi Wahyuni<sup>1</sup>, Firdatul Nurul Ramadha<sup>2</sup>, Eka Oktavia Andriani<sup>3</sup> Achmad Zalman Alfarizy<sup>4</sup>

<sup>1,2,3,4</sup> Fakultas Teknik, Program Informatika, Universitas Muhammadiyah Malang

E-mail:

<sup>1</sup> evidwi@umm.ac.id, <sup>2</sup> firdatulnr@webmail.umm.ac.id

<sup>3</sup> ekavia618@gmail.com, <sup>4</sup> azalmana27@gmail.com

### Abstrak

Penelitian ini membandingkan dua metode populer dalam pengembangan perangkat lunak, yaitu Scrum dan model incremental. Scrum, yang merupakan bagian dari Agile, menekankan iterasi singkat (sprint) dan kolaborasi tim yang intens untuk menghasilkan produk yang dapat digunakan secara berkala. Sebaliknya, model incremental berfokus pada pengembangan perangkat lunak secara bertahap, di mana setiap tahap menghasilkan sebagian dari fungsi sistem yang akhirnya akan diintegrasikan menjadi satu produk lengkap. Analisis komparatif ini mengeksplorasi prinsip-prinsip dasar, proses, output, kolaborasi tim, kelebihan, dan kekurangan dari kedua metode tersebut. Hasil penelitian menunjukkan bahwa Scrum cocok untuk proyek dengan persyaratan yang cenderung berubah dan tim yang terkoordinasi dengan baik, sementara model incremental lebih sesuai untuk proyek dengan persyaratan yang lebih stabil dan tim yang membutuhkan fleksibilitas dalam menyesuaikan proses. Pemahaman mendalam mengenai kelebihan dan kekurangan masing-masing metode dapat membantu pengambilan keputusan yang tepat dalam pengembangan perangkat lunak yang efektif, efisien, sesuai dengan kebutuhan proyek, meningkatkan peluang keberhasilan serta kepuasan pemangku kepentingan di lingkungan bisnis modern yang dinamis.

**Kata Kunci :** Model Perangkat Lunak, Metode Scrum, Metode Incremental

### Abstract

This study compares two popular methods in software development, Scrum and the incremental model. Scrum, as part of Agile, emphasizes short iterations (sprints) and intense team collaboration to deliver usable products incrementally. In contrast, the incremental model focuses on developing software gradually, where each stage produces a portion of the system's functionality that will eventually be integrated into a complete product. This comparative analysis explores the fundamental principles, processes, outputs, team collaboration, strengths, and weaknesses of both methods. The findings indicate that Scrum is suitable for projects with changing requirements and well-coordinated teams, while the incremental model is more appropriate for projects with more stable requirements and teams that require flexibility in tailoring the process. A deep understanding of the strengths and weaknesses of each method can aid in making informed decisions for effective, efficient, project-tailored software development endeavors, thereby enhancing the chances of success, stakeholder satisfaction, and competitiveness in today's dynamic business environment.

**Keywords:** Software Model, Scrum Method, Incremental Method

## 1. PENDAHULUAN

Dalam dunia pengembangan perangkat lunak, metodologi yang efektif dan terstruktur sangat penting untuk memastikan keberhasilan proyek. Ketika memutuskan metode manajemen proyek mana yang akan digunakan, kebutuhan para pemangku kepentingan, risiko yang terkait dengan proyek, ukuran proyek, biaya dan tentu saja kompleksitas proyek yang harus diperhitungkan [1]. Dua pendekatan yang sering digunakan adalah metode *Scrum* dan *incremental*. Menurut Fernando & Pedro (2021), *Scrum* diciptakan oleh Jeff Sutherland dan Ken Schwaber pada tahun 1993 dengan tujuan ingin membuat metode *Scrum* menjadi cara yang lebih cepat, efektif, dan lebih dapat diandalkan untuk mengembangkan perangkat lunak untuk industri teknologi [2]. Model *incremental* merupakan model pengembangan system pada *software engineering* yang menggabungkan model linier dan model system iteratif, Dimana pengembangannya dilakukan bertahap dan atau berulang agar fungsionalitas *system* sesuai harapan [3].

*Scrum*, menurut Andipradana & Hartono (2021), adalah teknik rekayasa software yang menggunakan prinsip-prinsip agile dan berfokus pada proses iterasi, peningkatan produk, dan kemampuan kolaborasi tim untuk mencapai hasil akhir. *Scrum*, menurut Schwaber & Sutherland, adalah kerangka kerja yang memiliki kemampuan untuk menangani masalah kompleks yang terus berubah. Selain itu, dinilai memiliki kemampuan untuk menyediakan produk yang berkualitas tinggi yang sesuai dengan keinginan pengguna yang inovatif dan produktif [4]. *Scrum* menekankan kolaborasi yang era tantara tim pengembang dan

pemangku kepentingan serta transparansi dalam proses pengembangan. Metodologi ini memungkinkan kita untuk beradaptasi dengan perubahan persyaratan dan prioritas dengan cepat sehingga oroduk akhir sesuai dengan kebutuhan pelanggan.

Metode *incremental* merupakan metode yang pengembangan sistemnya dibagi menjadi beberapa modul kemudian menentukan *timeline* dari setiap penyelesaian modul. Metode *incremental* ini merupakan salah satu metode pengembangan perangkat lunak yang mampu meminimalisir ketidaksesuaian dalam proses pengembangan perangkat lunak. Pada model *incremental*, spesifikasi kebutuhan yang sudah diperoleh ditempatkan pada setiap *increment* atau disebut dengan fase independent dalam pembagian system modul [5]. Proses pengembangan ini dibagi menjadi beberapa dan setiap fase nya menghasilkan peningkatan fungsionalitas yang lebih besar dibandingkan dengan *Scrum*. Pendekatan *incremental* memungkinkan tim untuk mengembangkan dan menguji setiap komponen secara terpisah sebelum menggabungkannya menjadi produk akhir.

Artikel perbandingan ini dapat memberikan wawasan berharga bagi praktisi, manajer proyek, dan pemimpin organisasi dalam mengambil keputusan yang tepat mengenai pendekatan pengembangan perangkat lunak yang akan diterapkan. Dengan menganalisis aspek-aspek seperti prinsip-prinsip dasar, *output*, proses, kolaborasi tim dalam *Scrum* dan model *incremental*, artikel ini dapat membantu pembaca memahami kekuatan dan kelemahan masing-masing model.

## 2. METODE PENELITIAN

Dalam penelitian ini, pendekatan kualitatif bersifat deskriptif digunakan untuk mengeksplorasi dan

membandingkan model *Scrum* dengan model *incremental* dalam pengembangan perangkat lunak. Penelitian kualitatif bersifat deskriptif merupakan penelitian yang menggunakan data yang telah dikumpulkan dan disajikan dalam bentuk kata-kata dan gambar-gambar. Laporan penelitian memuat kutipan-kutipan data sebagai ilustrasi dan dukungan fakta pada penyajian. Analisis dalam penelitian kualitatif cenderung dilakukan secara induktif [6]. Pendekatan kualitatif memungkinkan peneliti untuk memperoleh pemahaman yang lebih komprehensif tentang *Scrum* dan model *incremental* melalui studi literatur dan analisis komparatif.

Studi literatur merupakan pendekatan penelitian yang dilakukan dengan mencari referensi berdasarkan landasan teori yang relevan dengan kasus atau permasalahan yang ditemukan. Referensi dapat dicari dari buku, jurnal, artikel laporan penelitian dan situs-situs online di internet [7]. Studi literatur dilakukan dengan mengumpulkan dan mengkaji sumber-sumber terkait yang membahas kedua model tersebut. Dari sumber-sumber tersebut nantinya peneliti akan mencari informasi penting seperti prinsip-prinsip dasar, proses, kolaborasi tim, *output*, kekuatan dan kelemahan dari *Scrum* dan *incremental*. Sehingga peneliti dapat membandingkan *software* model dari *Scrum* dan *incremental*.

### 3. HASIL DAN PEMBAHASAN

Membandingkan *Scrum* dengan *incremental* adalah langkah penting untuk memahami kelebihan dan kekurangan masing-masing metodologi dalam konteks

pengembangan perangkat lunak modern. Selain itu masih banyak hal yang dapat dibandingkan antara *Scrum* dengan *incremental*, berikut saya jabarkan perbedaan antara keduanya :

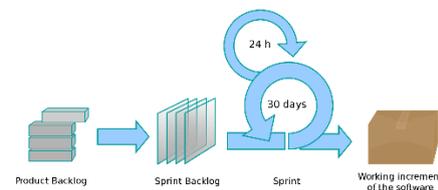
#### 3.1 Metode *Scrum*

*Scrum* yang merupakan bagian dari *agile*, menekankan pada iterasi yang disebut *sprint* dan kolaborasi tim yang intens untuk menghasilkan perangkat lunak yang dapat diubah dengan cepat sesuai dengan umpan balik pengguna.

##### 3.1.1 Prinsip-prinsip Dasar

*Scrum* yang didasarkan pada prinsip-prinsip *agile*, *Scrum* yang merupakan metode *agile* yang lumayan terkenal Dimana produktifitas menjadi lebih tinggi. Prinsip kerja dari *Scrum* adalah bekerja secara iterative dan bertahap sampai mencapai waktu yang telah ditentukan, sehingga proyek yang sedang dikembangkan dapat memenuhi kebutuhan yang diinginkan oleh konsumen [8].

##### 3.1.2 Proses



Gambar 1. Proses Metode *Scrum*

*Scrum* memiliki proses yang terdefinisi dengan jelas, yang terdiri dari beberapa fase yang disebut *sprint*. *Sprint* adalah periode waktu terbatas di mana tim pengembang bekerja untuk menghasilkan *incremental* produk yang siap digunakan. Siklus *Scrum* memiliki struktur yang terdefinisi dengan baik dan terdiri dari serangkaian kegiatan berulang dalam setiap iterasi. Tim *Scrum* terus mengulang siklus ini hingga mencapai tujuan proyek yang ditetapkan atau hingga proyek selesai [9].

Tahapan *Scrum* dibuat untuk

menciptakan keselarasan dan mengurangi tahapan yang kurang berkontribusi dalam proses *Scrum*. Jika pada tahapan *Scrum* ada salah satu tahapan yang tidak terlaksana maka akan dapat mengurangi transparansi dan menghilangkan kesempatan untuk melakukan peninjauan ulang dan melakukan suatu perubahan untuk perbaikan. Hal yang paling penting dalam tahapan *Scrum* biasa disebut *sprint*. *Sprint* merupakan suatu fase pengembangan perangkat lunak yang memiliki batasan waktu maksimal satu bulan, dengan durasi konsisten selama proses pengembangan produk [8].

Dalam *sprint* terdiri dari *sprint planning*, *daily Scrum*, *sprint review*, dan *sprint retrospective*. Berikut ini merupakan jabaran dari 4 tahapan *Scrum* [8] :

- *Sprint Planning*

Semua hal yang akan dikerjakan di dalam *sprint* akan direncanakan dalam *sprint planning*. Perencanaan ini dibuat sesuai dengan kesepakatan bersama dari seluruh anggota tim *Scrum* yang telah disepakati secara kolaboratif. *Sprint planning* dibatasi paling lama selama delapan jam untuk *sprint* yang berdurasi satu bulan.

- *Daily Scrum*

*Daily Scrum* sendiri merupakan kegiatan yang dilakukan dengan Batasan waktu maksimal selama 15 menit untuk tim pengembang dapat agar dapat mensinkronisasikan pekerjaan dan membuat perencanaan untuk 24 jam kedepan.

- *Sprint Review*

*Sprint Review* diadakan di setiap akhir *sprint*. *Sprint review* diadakan untuk merubah *product backlog* jika diperlukan. Pada saat *sprint review*, tim *Scrum* dan *stakeholder* berkolaborasi untuk membahas apa

yang sudah dikerjakan dalam *sprint* yang baru selesai.

- *Sprint Restropective*

Merupakan kesempatan bagi tim *Scrum* untuk membuat perencanaan mengenai peningkatan yang dapat dilakukan pada *sprint* berikutnya. *Sprint restropective* dilangsungkan setelah *sprint review* selesai dan sebelum *sprint planning* berikutnya dimulai. Pada tahap ini perlu dilakukan dengan durasi tiga jam untuk *sprint* yang berdurasi 1 bulan.

### 3.1.3 Output

Istilah *backlog* adalah bagian tak terpisahkan dari *Scrum*. Ada *product backlog* ada *sprint backlog*. Produk backlog adalah daftar seluruh karakteristik dari hasil akhir yang perlu dikembangkan agar hasilnya memuaskan. *Sprint backlog* adalah Kumpulan semua fitur yang disediakan oleh pemilik produk beserta tim yang memutuskan untuk *sprint* berikutnya [1]. Jadi *output* dari metode *Scrum* yaitu produk akhir yang dapat digunakan pada akhir setiap *sprint*. Setiap *sprint* harus menghasilkan peningkatan nilai yang dapat dilihat dan digunakan oleh pengguna atau konsumen. Ini dapat memungkinkan umpan balik yang cepat dan penyesuaian terhadap perubahan persyaratan.

### 3.1.4 Kolaborasi tim

Dalam penggunaan metode *Scrum* ini tidak membutuhkan tim yang besar karena pengelolaan perangkat lunak dalam metode *Scrum* tidak terlalu kompleks [10]. Tim *Scrum* tidak mencakup peran tradisional seperti programmer, desainer, tester atau analisa sistem. Tetapi para *stakeholder* yang bekerja sama dalam proyek untuk menyelesaikan pekerjaan mereka secara kolektif dan berkomitmen untuk menyelesaikannya. Terdiri atas *Product owner*, Tim Pengembang dan *Scrum master*.

*Scrum master* Fasilitator yang

melayani tim tetapi bukanlah manajer proyek memastikan fase metodologi *Scrum* dilakukan dengan baik, melindungi tim, dan memastikan mereka tidak terlalu berkomitmen pada apa yang dikerjakan karena perubahan dapat terjadi setiap *Sprint*. Tim pengembang adalah kelompok orang yang bertanggung jawab untuk menghasilkan produk perangkat lunak. *Product owner* satu-satunya orang yang bertanggung jawab mengelola backlog produk. Mereka juga bertanggung jawab atas sukses pengembangan produk dan hasil kerja tim pengembang. dibentuk dan didukung oleh organisasi sehingga mereka dapat mengatur dan mengelola operasi mereka sendiri. Efektivitas dan efisiensi tim pengembang secara keseluruhan akan meningkat sebagai hasil dari sinergi yang ada di dalam tim [8].

### 3.1.5 Kelebihan

Kelebihan *Scrum* menurut Schwaber & Sutherland, *Scrum* adalah kerangka kerja yang dapat mengatasi masalah kompleks yang selalu berubah dan juga dinilai dapat memberikan produk berkualitas tinggi yang sesuai dengan keinginan pengguna secara kreatif dan produktif. *Scrum* juga memudahkan para pemangku kepentingan dalam menjalankan proyek karena perencanaannya lebih mudah dipahami dan pengembangannya lebih fleksibel. Karena memiliki tingkat produktivitas yang tinggi, *Scrum* menjadi metode pengembangan sistem yang populer [4].

### 3.1.6 Kekurangan

*Scrum* memiliki kekurangan, seperti potensi tekanan yang berlebihan pada tim karena *sprint*

yang singkat dan tuntutan untuk selalu menghasilkan produk yang dapat digunakan. Ini dapat menyebabkan kelelahan dan berkurangnya kualitas kode jika tidak dikelola dengan baik. Selain itu, tingkat formalitas dan struktur yang tinggi dalam *Scrum* dapat membatasi fleksibilitas tim dalam menyesuaikan proses sesuai dengan kebutuhan proyek yang unik. Tim *Scrum* juga sangat bergantung pada komitmen dan kedisiplinan tim, jika tim tidak berkomitmen maka proses *Scrum* dapat gagal [8] [11].

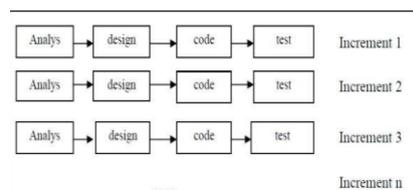
## 3.2 Metode Incremental

*Incremental development* berfokus pada pengembangan perangkat lunak secara bertahap, di mana setiap tahap menghasilkan sebagian dari fungsi sistem yang akhirnya akan terintegrasi menjadi setiap tahap menghasilkan sebagian dari fungsi sistem yang akhirnya akan terintegrasi menjadi satu produk yang lengkap

### 3.2.1 Prinsip-prinsip Dasar

Prinsip dasar metode *incremental* adalah model iterative dan linier yang memungkinkan kesalahan dikenali sampai pengujian penerimaan, Ketika mungkin sudah terlambat untuk memperbaiki kesalahan. Dengan *life cycle* pengembangan *incremental*, risiko mengembangkan hal yang salah berkurang dengan memecah proyek menjadi serangkaian sub proyek kecil [12].

### 3.2.2 Proses



Gambar 2. Proses Metode *Incremental*

Model *incremental* adalah model pengembangan sistem pada rekayasa

perangkat lunak yang dipecah menjadi beberapa fungsi atau bagian sehingga model pengembangannya secara bertahap [13]. Proses dari metode *incremental* sendiri terdiri dari 4 tahapan metode *incremental* yaitu, analisis kebutuhan, desain (*User interface* dan basis data), implementasi (*coding*), pengujian (*testing*). 4 Tahapan itu akan terus berulang hingga iterasi selesai [14].

Penjelasan lebih detail mengenai 4 tahapan diatas adalah sebagai berikut [13] :

- Analisa diartikan sebagai proses dari mengolah data menjadi informasi sehingga informasi-informasi tersebut dapat digunakan untuk menjawab masalah-masalah yang berkaitan dengan proyek yang dikerjakan.
- Desain menghasilkan rancangan yang memenuhi kebutuhan yang ditentukan selama tahapan analisis. Hasil akhirnya berupa spesifikasi rancangan yang sangat rinci sehingga mudah diwujudkan saat proses *coding*.
- Implementasi yaitu proses mengubah hasil desain ke dalam kode yang dimengerti oleh mesin *komputer*.
- Pengujian yaitu menguji apakah hasil *coding* sudah dapat digunakan dengan baik atau belum.

### 3.2.3 Output

Pada model *incremental* yang pertama sering disebut sebagai *core product*. *Core product* adalah kebutuhan yang diperlukan oleh pengguna, terkadang banyaknya tambahan fitur yang diperlukan dapat menyebabkan tidak semuanya dapat tersampaikan. Oleh karena itu, hasil evaluasi dari *core product* dapat dijadikan sebagai rencana

perkembangan untuk *incremental* selanjutnya dengan cara memodifikasi *core product* agar menjadi lebih baik untuk memenuhi kebutuhan pengguna (fitur dan fungsi) [15]. *Output* dari metode *incremental* ini nantinya akan dijadikan masukan/*input* untuk *incremental* selanjutnya [3].

### 3.2.4 Kolaborasi Tim

Untuk kolaborasi tim dalam model *incremental* ini tim pengembangan berupaya menyelesaikan setiap pembangunan tambahan secara tepat waktu. Rilis reguler mewakili peningkatan bertahap dalam fungsi dan kualitas, sehingga tujuannya adalah untuk memberikan produk yang berfungsi sedikit demi sedikit [16]. Kolaborasi tim dalam model *incremental* dapat bervariasi tergantung pada praktik-praktik yang diimplementasikan oleh tim. Meskipun model ini tidak memiliki struktur dan praktik yang terdefinisi secara eksplisit seperti *Scrum*. Kolaborasi yang efektif masih dapat dicapai melalui perencanaan yang baik, komunikasi yang terbuka, dan pembagian tanggung jawab yang jelas di antara anggota tim. Evaluasi setelah setiap iterasi juga dapat memberikan kesempatan bagi tim untuk berkolaborasi dalam mengidentifikasi masalah dan area perbaikan.

### 3.2.5 Kelebihan

Kelebihan dari model *Incremental* antara lain [14] :

- perangkat lunak yang akan dikembangkan membutuhkan waktu yang lebih singkat untuk diselesaikan pada setiap tahapan *incremental*.
- Termasuk model yang fleksibel dan lebih bisa mengakomodasi kebutuhan pengguna jika harus merubah kebutuhan.
- Pada setiap tahapan *incremental* lebih mudah untuk melakukan perubahan sesuai dengan keinginan pengguna, dan

Pengguna bisa memberikan umpan balik pada setiap tahapan *incremental*.

- Pengguna bisa memberikan umpan balik pada setiap tahapan *incremental*
- Keunggulan metode *incremental* dibandingkan metode yang lain adalah metode ini mempunyai fleksibilitas tinggi dengan resiko yang lebih rendah dalam kemungkinan munculnya bugs atau kesalahan dalam implementasi

3.2.6 Kekurangan

Kekurangan dari metode ini [14] :

- Membutuhkan analisa dan perencanaan desain yang bagus dan detail.
- Jika terjadi kesalahan analisa di awal, besar kemungkinan semua tahapan iterasi *incremental* akan terpengaruh, sehingga membutuhkan cost yang lebih besar daripada menggunakan model yang konvensional (*waterfall*).
- Setiap perulangan/iterasi *incremental* tidak bisa saling *overlap*, harus diselesaikan secara berurutan

Berikut penulis menjabarkan perbedaan antara kedua metode tersebut:

Tabel 1. Perbandingan *Scrum* dan *Waterfall*

| Aspek                | <i>Scrum</i>  | <i>Incremental</i>   |
|----------------------|---|--|
| <b>Prinsip Dasar</b> | Didasarkan pada prinsip-prinsip <i>Agile</i> . Produktivitas tinggi melalui iterasi dan kolaborasi tim. | Kombinasi model iterative dan linier. Risiko berkurang dengan memecah proyek menjadi sub-proyek. |
| <b>Proses</b>        | Terdiri dari beberapa fase  | Terdiri dari 4 tahapan:  |

|                       |   |  |
|-----------------------|---|--|
|                       | yang disebut <i>sprint</i> ( <i>product backlog</i> , <i>sprint planning</i> , <i>sprint</i> , <i>sprint review</i> , retrospective). Proses berulang hingga tujuan tercapai. | analisis kebutuhan, desain ( <i>User interface</i> dan basis data), implementasi ( <i>coding</i> ), pengujian ( <i>testing</i> ). Tahapan berulang hingga iterasi selesai. |
| <b>Output</b>         | Produk akhir dapat digunakan pada akhir setiap <i>sprint</i> . Fitur ditambahkan berdasarkan prioritas backlog.   | <i>Core product</i> yang dihasilkan pada setiap tahap <i>incremental</i> . Evaluasi <i>core product</i> menjadi dasar untuk tahap berikutnya.                              |
| <b>Kolaborasi Tim</b> | Melibatkan <i>Product owner</i> , Tim Pengembang, dan <i>Scrum master</i> . Tim bekerja secara mandiri dan kolaboratif.   | Bervariasi tergantung pada praktik yang diterapkan oleh tim. Kolaborasi melalui perencanaan dan komunikasi terbuka.  |
| <b>Kelebihan</b>      | 1. Dapat mengatasi masalah kompleks yang berubah-ubah. 2. Produktivitas tinggi. 3.  | 1. Waktu penyelesaian setiap tahapan lebih singkat. 2. Fleksibel dan dapat   |

|  |   |   |
|--|---|---|
|  | Fleksibilitas dalam pengembangan n. 4. Umpan balik cepat. | mengakomodasi perubahan kebutuhan. 3. Risiko bugs lebih rendah. |
|--|---|---|

#### 4. KESIMPULAN

*Scrum* dan model *incremental* memiliki perbedaan signifikan dalam proses, struktur, dan filosofi yang mendasari meskipun keduanya bertujuan untuk menghadapi kompleksitas dan ketidakpastian dalam pengembangan perangkat lunak. *Scrum* memiliki proses yang terdefinisi dengan jelas, melibatkan kolaborasi tim yang intens, dan menghasilkan produk yang dapat digunakan pada akhir setiap *sprint*. Di sisi lain, model *incremental* berfokus pada pengembangan perangkat lunak secara bertahap dengan setiap tahap menghasilkan peningkatan fungsionalitas yang lebih besar, serta memungkinkan fleksibilitas yang lebih tinggi dalam penyesuaian proses sesuai dengan

Model *incremental* dapat cocok untuk proyek dengan persyaratan yang lebih stabil dan tim yang membutuhkan fleksibilitas yang lebih besar dalam menyesuaikan proses, sementara *Scrum* dapat menjadi pilihan yang baik untuk proyek dengan tim yang terkoordinasi dengan baik dan persyaratan yang cenderung berubah. Pemahaman yang mendalam tentang kelebihan dan kekurangan setiap metode akan membantu membuat keputusan yang tepat dan meningkatkan peluang keberhasilan proyek pengembangan perangkat lunak merek.

#### 5. SARAN

Melakukan eksperimen langsung dengan kedua metode tersebut untuk mendapatkan pemahaman lebih mendalam mengenai efektifitas dan efisiensi metode *incremental* dengan *Scrum*. Dengan melakukan hal tersebut diharapkan dapat mencapai tujuan yang lebih baik kedepannya.

#### DAFTAR PUSTAKA

- [1] M. Gaborov, D. Karuovic, M. Kavalic, D. Radosav, D. Milosavljevic, S. Stanisljevic And J. Bushati, "Comparative Analysis Of Agile And Traditional Methodologies In It Project Management," *Journal Of Applied Technical And Educational Sciences Jates*, Pp. 1-24, 2021.
- [2] F. Almeida And P. Carneiro, "Perceived Importance Of Metrics For Agile *Scrum* Environments," *Information*, Vol. 14, 2023.
- [3] D. C. E. Manalu And A. Rachman, "Rancang Bangun Sistem Informasi Penjualan Batik Berbasis Web Menggunakan Model *Incremental*," *Jurnal Riset Inovasi Bidang Informatika Dan Pendidikan Informatika (Kernel)*, Pp. 41-48, 2022.
- [4] A. Andipradana And K. D. Hartono, "Rancang Bangun Aplikasi Penjualan Online Berbasis Web Menggunakan Metode *Scrum*," *Jurnal Algoritma*, Pp. 161-172, 2021.
- [5] R. Basatha, T. Rahmawati And W. Ardianto, "Analisis Dan Perancangan Sistem Informasi Layanan Kerusakan Mesin Edc Menggunakan Metode *Incremental*," *Konstelasi: Konvergensi Teknologi Dan Sistem Informasi*, Vol. 3 No.2, 2023.
- [6] S. Romlah, "Penelitian Kualitatif Dan Kuantitatif (Pendekatan Penelitian Kualitatif Dan Kuantitatif)," *Pancawahana: Jurnal Studi Islam*, Vol. 16 No.1, 2021.
- [7] M. Rizky And Y. Sugiarti, "Penggunaan Metode *Scrum* Dalam Pengembangan Perangkat Lunak: Literature Review," *Journal Of Komputer Science An Engineering (Jcse)*, Vol. 3 No. 1, Pp. 41-48,

2021. San Francisco, 2023.
- [8] A. Bimantasa And J. Sutresna, "Perancangan Dashboard Sistem Informasi Dengan Metode *Scrum* Menggunakan Azura Board," *Jurnal Ilmu Komputer Dan Science*, Pp. 519-526, 2024.
- [9] L. Magdalena, *Scrum Agile: Optimalisasi Kualitas Produk Manajemen*, Sonpedia Publishing Indonesia, 2023.
- [10] N. T. Rachman And D. Sushandoyo, "Analysis Of *Scrum* Implementation In Digital Startup Product," *Costing:Journal Of Economic, Business And Accounting*, Pp. 190-196, 2021.
- [11] A. P. M. & S. S. M. H. Raza, "Critical Analysis Of *Scrum* Process: A Student's Perspective," *International Journal Of Modern Education And Komputer Science*, Pp. 1-10, 2021.
- [12] A. Thakur, D. Singh And H. C. Maurya, "A Survey On *Incremental* Software Development Life Cycle Model," *Of Engineering Technology And Komputer Research (Ijetcr)*, Vol. 3, No. 2, Pp. 102-105.
- [13] D. A. Prabowo, S. Y. Reswan And M. , "Implementasi Metode *Incremental* Pada Sistem Pengelolaan Badan Usaha Milik Desa Berbasis Web," *Jukomika - (Jurnal Ilmu Komputer Dan Informatika)*, Vol. 4 No.2, 2021.
- [14] M. H. Murdani, N. Widhiyanta, S. Priyambudi And M. Asrori, "Rancang Bangun Sistem Informasi Akuntansi Menggunakan Metode *Incremental* - Studi Kasus Di Koperasi Karyawan Coca Cola Sier," *Stiki Informatika Jurnal*, Pp. 67-74, 2022.
- [15] N. Ahmad, E. Krisnanik And F. G. J. Rupielele, *Analisa & Perancangan Sistem Informasi Berorientasi Objek*, Penerbit Widina, 2022.
- [16] Plutora, "*Incremental* Model: What It Is And How To Implement It," <https://www.plutora.com/blog/incremental-model-what-and-how-to-implement-it>,
-