UNIVERSITAS INDONESIA

KERANGKA KERJA *ENTERPRISE AGILE* BERDASARKAN *THE ESSENCE OF SOFTWARE ENGINEERING* MENGGUNAKAN METODOLOGI *DESIGN SCIENCE RESEARCH (DSR)*

RINGKASAN DISERTASI

TEGUH RAHARJO
1806261490

FAKULTAS ILMU KOMPUTER
PROGRAM DOKTOR ILMU KOMPUTER
DEPOK
JULI 2023

UNIVERSITAS INDONESIA

KERANGKA KERJA *ENTERPRISE AGILE* BERDASARKAN *THE ESSENCE OF SOFTWARE ENGINEERING* MENGGUNAKAN METODOLOGI *DESIGN SCIENCE RESEARCH (DSR)*

RINGKASAN DISERTASI

TEGUH RAHARJO
1806261490

FAKULTAS ILMU KOMPUTER
PROGRAM DOKTOR ILMU KOMPUTER
DEPOK
JULI 2023

KATA PENGANTAR

Segala puji bagi Allah Subhaanahu wata'ala karena berkat rahmat-Nya penulis berhasil menyelesaikan Hasil Penelitian Karya Akhir ini tepat pada waktu yang direncanakan.

Penulis menyadari bahwa dalam penulisan proposal ini masih jauh dari sempurna, oleh karena itu kritik dan saran dari semua pihak yang bersifat membangun selalu penulis harapkan demi kesempurnaan proposal ini. Selanjutnya penulis menyampaikan terima kasih kepada:

1. Orang Tua yang telah mendukung penulis di dalam banyak doanya agar hasil penelitian ini dapat berhasil dengan baik.
2. Bapak Prof Eko K Budiardjo sebagai promotor utama di dalam penelitian ini
3. Ibu Betty Purwandari, Ph.D sebagai dosen pembimbing akademik yang banyak sekali membantu penulis di dalam memberikan semangat dan motivasi untuk menyelesaikan proposal ini.
4. Teman-teman kelas Doktor Ilmu Komputer, yang telah memberikan dukungan dalam penyelesaian proposal penelitian ini.

Akhir kata semoga hasil penelilitan ini dapat bermanfaat bagi penulis dan juga untuk pengembangan ilmu pengetahuan.

Jakarta, Juli 2023

Penulis

ABSTRAK

Nama	:	Teguh Raharjo
Program Studi	:	Doktor Ilmu Komputer
Judul	:	KERANGKA KERJA *ENTERPRISE AGILE* BERDASARKAN *THE ESSENCE OF SOFTWARE ENGINEERING* MENGGUNAKAN METODOLOGI *DESIGN SCIENCE RESEARCH (DSR)*

Survei yang dilakukan oleh Project Management Institute dan VersionOne memaparkan bahwa pengembangan proyek menggunakan pendekatan *agile* telah berkembang sangat pesat. Kerangka kerja *enterprise agile* atau *scaling agile* diharapkan dapat menangani problem yang lebih kompleks pada saat organisasi beralih dari pendekatan tradisional ke *agile*. Meskpun saat ini terdapat metode-metode *scaling agile* yang populer memiliki praktik-praktik yang diajukan, tidak semua praktik dan metode tersebut cocok untuk organisasi. Oleh karena itu, memilih metode dan praktik *agile* dari beberapa metode dibandingkan adaptasi satu metode secara keseluruhan dapat dipertimbangkan sebagai strategi untuk pemilihan praktik yang diperlukan. Penelitian oleh *scrum.org* menyebutkan bahwa 49% organisasi tidak memiliki kerangkan kerja untuk penerapan *scaling agile* mereka, sedangkan pemilihan kerangka kerja merupakan salah satu proses yang penting. Saat ini, belum ada penelitian yang membahas mengenai kerangka kerja yang bersifat *general* untuk *enterprise agile*. Penelitian ini bertujuan untuk membangun kerangka kerja *enterprise agile* berdasarkan konsep *The Essence SEMAT Kernel* dan *general theory of software engineering*. *Design science research* (DSR) digunakan sebagai metodologi penelitian, karena karakteristiknya yang sesuai dengan tujuan penelitian ini. Penelitian ini menghasilkan luaran berupa kerangka kerja *enterprise agile* yang sudah dilakukan proses uji coba pada beberapa organisasi di Indonesia. Proses evaluasi berdasarkan konsep DSR sudah dilakukan sampai beberapa kali iterasi. Evaluasi dilakukan pada setiap proses desain dan setelah tahapan uji coba. Evaluasi oleh para pakar, praktisi dan pakar internasional dilakukan menggunakan metode wawancara daring menggunakan *video conference*, *focus group discussion (FGD)* dan *structured-interview* dengan formulir daring. Untuk kontribusi akademik, penelitian ini memberikan rujukan kepada penelitian lainnya pada bidang *enterprise agile*. Berdasarkan Software Engineering Body of Knowledge (SWEBOK), penelitian ini menambahkan referensi mengenai *scaling agile* methods pada pada area *software engineering model and methods.*

Kata kunci: *agile, enterprise agile framework, scaling agile*, *The Essence, design science research* (DSR)

ABSTRACT

Name	:	Teguh Raharjo
Study Program	:	Doktor Ilmu Komputer
Title	:	The Enterprise *agile* Framework based on the Essence of Software Engineering using design science research (DSR)

A survey conducted by the Project Management Institute and VersionOne showed that the development of projects using the agile approach had developed very rapidly. The enterprise agile framework or scaling agile framework is expected to solve the complex problem while they are transitioning from traditional to agile approach. Although there are several agile methods each encompasses multiple practices, not all of them may fit for all organizations. Therefore, picking appropriate agile practices from different methods rather than adapting to a whole agile method, can be considered as practice selection strategy. The research by Scrum.org explores that 49% of organization have no framework for their implementation of scaled agile, whereas the selection of the framework is of the important process. Currently, there are no studies to explore the generic framework of the enterprise agile. In the other related software engineering field, there are the common ground concept. This study aims to develop an enterprise agile framework based on the concept of The Essence SEMAT kernel and general theory of software engineering. Design science research (DSR) is used as a research methodology, because of its characteristics that are in accordance with the objectives of this research. This research produces an output in the form of an enterprise agile framework that has been evaluated in several organizations in Indonesia. The evaluation process based on the DSR concept has been carried out for several iterations. Evaluation is carried out at each design process and after the trial phase. Evaluation by experts, practitioners and international experts is carried out using online interview methods using video conferencing, focus group discussions (FGD) and structured-interviews with online forms. In the academic field, this research contributes to adding agile scaling methods to SWEBOK in the Software Engineering Model and Methods area. For practitioners, the framework that was built can be used as a guide for implementing agile scaling in organizations. For academic contributions, this research provides references to other research in the field of enterprise agile. Based on the Software Engineering Body of Knowledge (SWEBOK), this study adds a reference to scaling agile methods in software engineering models and methods.

Keywords: agile, enterprise agile framework, scaling agile, The Essence, design science research (DSR)

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

Bab ini membahas latar belakang penelitian, perumusan masalah, pertanyaan penelitian, tujuan penelitian dan manfaat penelitian. Latar belakang masalah dijabarkan menurut beberapa survei dan penelitian sebelumnya mengenai perkembangan implementasi *agile*.

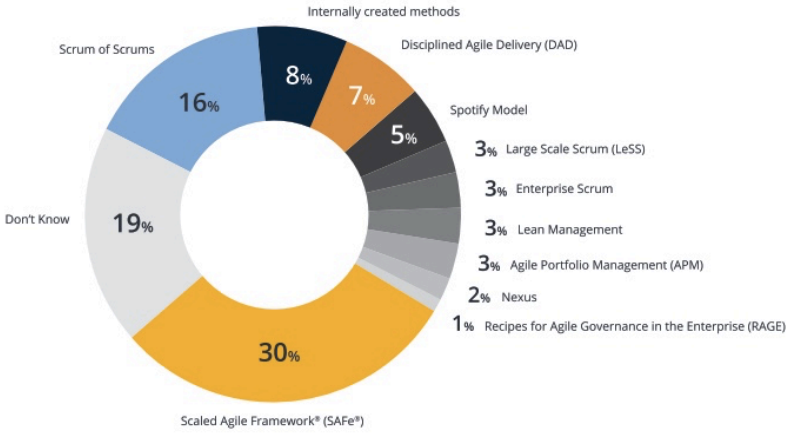
1.1. Latar Belakang

Survei yang dilakukan oleh Project Management Institute melalui kegiatan *global survey* mereka pada laporan tahun 2017 pada PMI's Pulse of The Profession (PMI, 2017c) dan laporan tahunan ke-13 yang dilakukan oleh VersionOne (VersionOne, 2019) menunjukkan bahwa pengembangan proyek menggunakan metode *agile* meningkat sangat pesat. Kenyataannya, 71 persen organisasi melaporkan menggunakan pendekatan *agile* pada proyek-proyek mereka. Fakta lainnya selama 12 bulan sebelumnya pada tahun 2017 satu dari lima proyek menggunakan pendekatan *agile*, sedangkan satu dari sisanya menggunakan penggunaan kombinasi antara *agile* dan tradisional (PMI, 2017c). Implementasi *agile* sendiri memiliki kemajuan yang sangat pesat sejak deklarasi *Agile Manifesto* sejak tahun 2001 (PMI, 2017b). Survei VersionOne melaporkan beberapa alasan penggunaan *agile*, seperti ditunjukkan pada Gambar 1.1.

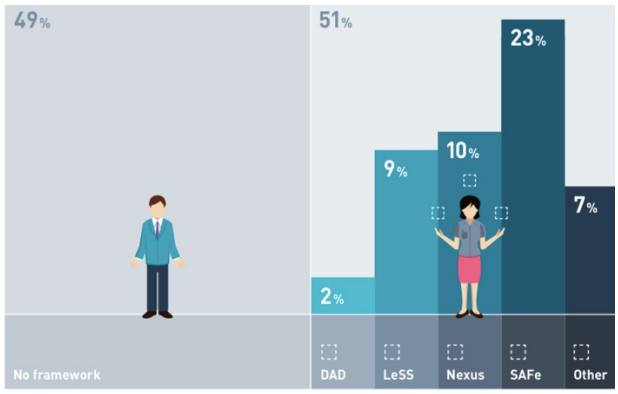
Gambar 1.1 Alasan Penggunaan *agile* (VersionOne, 2019)

1.2. Perumusan Masalah Penelitian

Saat ini terdapat berbagai macam kerangka kerja yang populer dan sudah dipakai di beberapa organisasi, seperti: SAFe, Less, DAD, Nexus, dan Scrum of Scrum. Hal tersebut seperti dipaparkan oleh survei yang dilakukan oleh (VersionOne, 2019) pada Gambar 1.2 dan (Scrum.org, 2019) pada Gambar 1.3. Dari metode yang ada, tidak ada satu metode yang cocok dipakai untuk semua organisasi (PMI, 2017a). Salah satu pendekatan yang dapat dipakai adalah mengambil beberapa metode yang cocok untuk digabungkan dan diterapkan pada organisasi, dibandingkan adopsi hanya satu metode saja secara menyeluruh (Gandomani & Nafchi, 2015). Survei yang dilakukan oleh Scrum.org (Scrum.org, 2019) menyebutkan bahwa 49% organisasi belum memiliki kerangka kerja di dalam implementasi *scaling agile*, di mana pemilihan kerangka kerja merupakan salah satu proses yang penting. Survei oleh VersionOne juga menunjukkan hal yang serupa dengan hasil 19% responden tidak mengetahui akan menggunakan kerangka kerja tertentu.



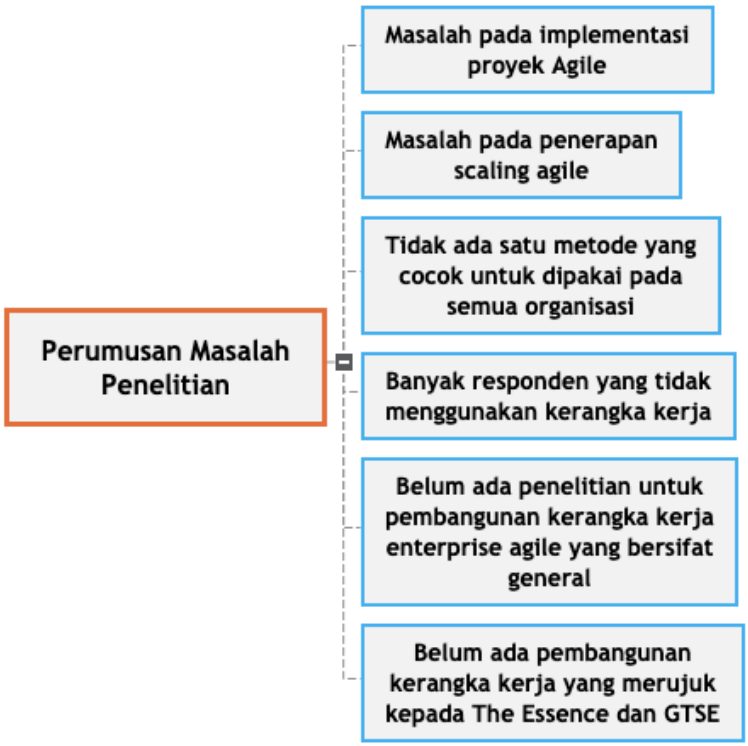
Gambar 1.2 Perbandingan Penggunaan *scaling agile* (VersionOne, 2019)



Gambar 1.1 Perbandingan Penggunaan *scaling agile* (Scrum.org, 2019)

Berdasarkan penelusuran yang dilakukan oleh penulis, belum ada penelitian yang membangun suatu kerangka kerja untuk *enterprise agile* atau *scaling agile*. Pada bidang rekayasa perangkat lunak, sudah ada pendekatan yang telah dibuat, yaitu The Essence of the software engineering The SEMAT Kernel (Jacobson, Ng, McMahon, Spence, & Lidman, 2012) dan *general theory of software engineering* (Johnson & Ekstedt, 2016). Sesuai paparan perumusan masalah sebelumnya, penulis berpandangan akan pentingnya pembangunan kerangka kerja yang bersifat umum yang dapat dipakai untuk semua organisasi.

Istilah *enterprise agile framework* menurut beberapa sumber (Rose, 2018), (Deloitte, 2019), (D. Leffingwell, 2020) merujuk kepada *framework scaling agile*, seperti SAFe, LeSS, Discipline *agile* dan Spotify. *Scaling agile framework* sendiri merupakan pengembangan dari *agile* pada level tim seperti Scrum dan Kanban. Implementasi *scaling agile* dapat mulai dari level kecil, minimal 2 tim (small implementation) sampai level *enterprise* (large implementation) (Larman & Vodde, 2016). Penjelasan lebih terperinci mengenai jumlah tim *agile* terdapat pada bab 2 pada subbab metode *scaling agile*.



Gambar 1.2 Perumusan Masalah Penelitian

Penelitian ini bertujuan untuk membangun kerangka kerja *agile* untuk organisasi (*enterprise agile framework*) atau *scaling agile* yang bersifat umum sebagai sebuah *common ground*. Istilah *common ground* merujuk kepada The Essence of the *software engineering* The SEMAT Kernel (Jacobson et al., 2012). The Essense dipakai sebagai rujukan karena pendekatan ini menggabungkan metode rekayasa perangkat yang sudah ada serta ditunjang oleh *best practices* dan komunitas rekayasa perangkat lunak. *General teory of software engineering* juga dipakai sebagai model rujukan untuk penelitian ini. Beberapa penelitian sebelumnya (Park, McMahon, & Myburgh, 2016), (Graziotin, 2012) menggunakan The Essense sebagai landasan utama di dalam membangun keluaran penelitian mereka.

1.3. State of The Arts Penelitian

State of the arts dari penelitian ini adalah:

1. Penelitian dalam bidang di *agile* dari sisi organisasi dan pengembangan perangkat lunak masih menjadi tren di 5 tahun terakhir (Sánchez-morcilio et al., 2016), (PMI, 2017a)
2. Masalah transformasi dan *scaling agile* menjadi salah satu topik *challenges* terbesar. Hal ini berdasarkan penelusuran penulisan melalui penelitian SLR sebelumnya (Raharjo & Purwandari, 2020a)
3. Belum ada penelitian untuk pembangunan kerangka kerja untuk *scaling agile* atau *enterprise agile* berdasarkan The Essence dan teori *software engineering*
4. *The Essense of software engineering* sebagai satu satu landasan berpikir, masih menjadi tren yang terus dikembangkan (OMG, 2018), (Jacobson et al., 2012)
5. Metodologi pengembangan *framework* dengan mengacu pada *The Essence of software engineering* dan *general theory of software engineering* masih sedikit. Penelusuran yang dilakukan oleh penulis, hanya terdapat 5 penelitian yang merujuk langsung kepada The Essence.

1.4. Pertanyaan Penelitian

Pertanyaan penelitian atau *research question* (RQ) yang akan dibahas pada penelitian ini yaitu:

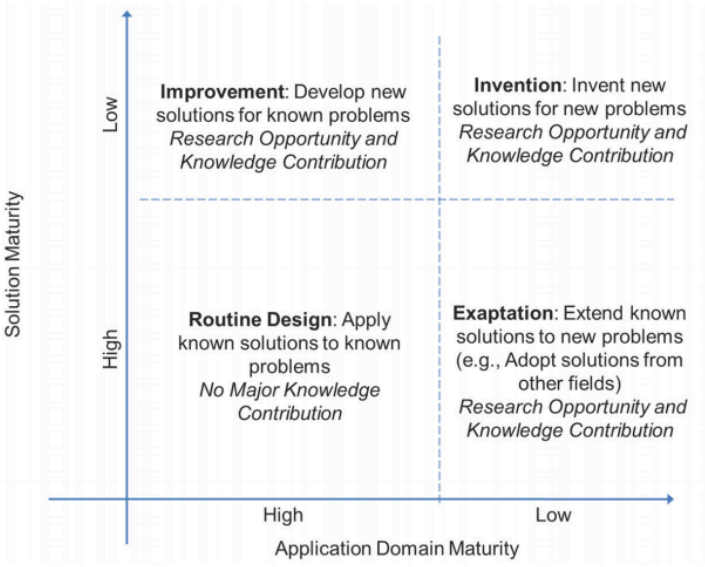
1. Bagaimana *common ground* untuk *enterprise agile* pada organisasi berdasarkan pemikiran *The Essence of software engineering* dan *general theory of software engineering*?

2. Bagaimana model konseptual untuk *enterprise agile* yang dikembangkan dari komponen *common ground*?
3. Bagaimana *current practice*s berdasarkan *systematic literature review (SLR)* yang sesuai menurut *common ground*?
4. Bagaimana pemetaan The Essence SEMAT Kernel pada *common ground* dari kerangka kerja *enterprise agile*?
5. Bagaimana susunan kerangka kerja dari *enterprise agile?*

Ekstensi The Essence SEMAT Kernel dibutuhkan karena Kernel saat ini yang ada hanya bersifat umum untuk *software engineering*. Ekstensi Kernel dimungkinan untuk dilakukan (OMG, 2018). Ekstensi Kernel ini sudah pernah dilakukan sebelumnya pada area *business analysis.*

1.5 Kontribusi Penelitian

Penelitian ini merupakan Design Science Research (DSR) yang memberikan kontribusi kepada pengembangan pengetahuan yang baru. Penjelasan dan justifikasi penelitian ini menggunakan DSR dijelaskan pada bab 2 sub bab metodologi penelitian. Terdapat kuadran kontribusi penelitian pada ilmu pengetahun seperti ditunjukkan pada Gambar 1.5. Penelitian ini memiliki kontribusi pada kuadaran *improvement*, yaitu kontribusi pada ilmu pengetahuan dengan membangun suatu solusi yang baru pada masalah yang sudah diketahui.



Gambar 1.5 *DSR Knowledge Contribution Framework* (Gregor & Hevner, 2013)

1.6 Tujuan Penelitian

Tujuan dari penelitian ini untuk menjawab pertanyaan penelitian yang telah didefinisikan yaitu:

1. Membangun *common practice* untuk *enterprise agile* pada organisasi berdasarkan pemikiran The Essence of *Software* Engineering dan General Theory Of *Software* Engineering
2. Mengembangkan suatu *common framework* untuk *enterprise agile* berdasarkan *common ground scaling agile* methods
3. Mengembangkan ekstensi Kernel pada The Essence SEMAT Kernel untuk *enterprise agile*
4. Menyediakan strategi implementasi untuk penerapan *enterprise agile framework* pada organisasi

1.7 Manfaat Penelitian

Penelitian ini memiliki manfaat secara keilmuwan dan praktik. Manfaat penelitian secara keilmuwan atau akademik adalah:

1. Mengisi literatur untuk *generic scaling agile framework*
2. Mengisi literatur pada *agile* model pada SWEBOK *knowledge area software engineering model and methods* pada sub *knowledge area agile models*.
3. Sebagai referensi penggunaan *common ground* untuk pembuatan kerangka kerja *agile*

Sedangkan manfaat penelitian secara bagi para praktisi adalah:

1. Kerangka kerja *scaling agile* dapat digunakan sebagai penduan untuk penerapan *scaling agile* pada organisasi
2. Ekstensi SEMAT kernel dapat digunakan oleh praktisi yang memanfaatkan SEMAT kernel pada implementasi *scaling agile*

1.8 Ruang Lingkup Penelitian

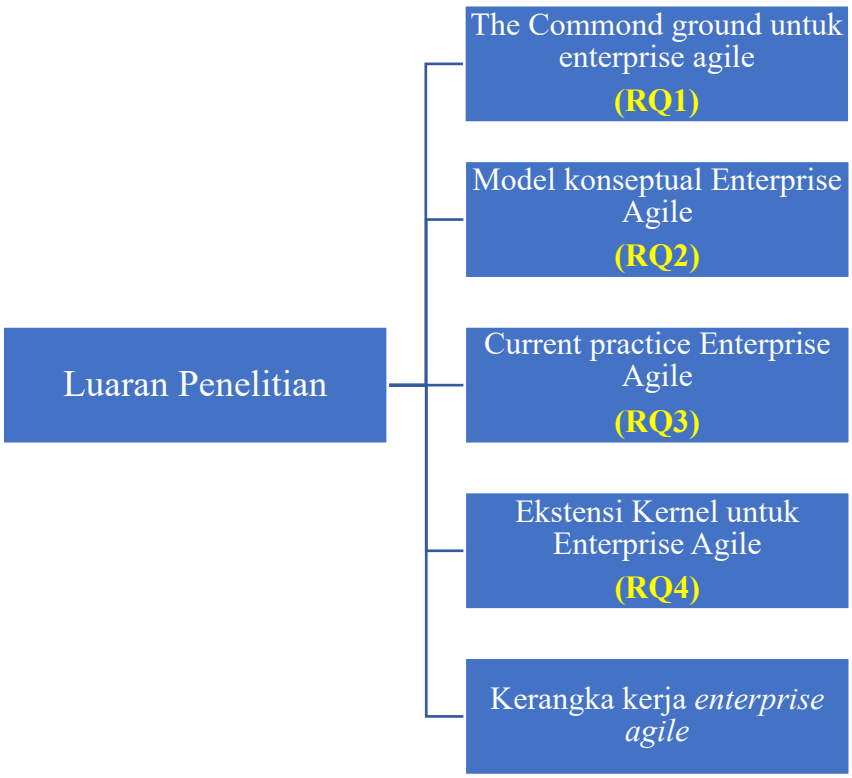
Levelisasi implementasi *agile* seperti ditunjukkan dapat dibagi menjadi beberapa level. Pada level paling bawah terdapat *team methods*, seperti Scrum, XP, dan Kanban (Rose, 2018). Pada level *enterprise agile* terdapat metode *enterprise agile* yang merupakan implementasi dari *scaling agile* seperti SAFe, LeSS, Discipline *agile*, Nexus dan *framework* lainnya. Pada level *busines agility* terdapat implmentasi *agile* untuk semua bidang bisnis, seperti *human resource*, *finance*, *marketing* dan lainnya.

1.9 Luaran atau *Output* dari Penelitian

Luaran penelitian ini berdasarkan jenis penelitian design science research (DSR) yaitu menghasilkan artefak. Artefak yang dihasilkan merupakan kerangka kerja yang terdiri dari metode dan praktik. Metode terdiri dari praktik yang dapat digunakan di dalam implementasi *scaling agile*.

Luaran dari penelitian seperti diilustrasikan pada Gambar 1.6 yaitu jawaban dari pertanyaan penelitian dan strategi implementasi kerangka kerja. Secara garis besar luaran penelitian yaitu:

1. *Common ground* dari *enterprise agile* berdasarkan RQ1.
2. Model konseptual dari *enterprise agile* berdasarkan RQ2
3. *The common practice of enterprise agile* berdasarkan RQ3
4. Implementasi *The Essence language* untuk *enterprise agile* berdasarkan RQ4
5. Susunan kerangka kerja untuk *enterprise agile* berdasarkan RQ5



Gambar 1.6 Luaran Penelitian

BAB 2
STUDI PUSTAKA

Hubungan antar kata kunci pada penelitian ini yaitu istilah kerangka kerja agile atau ***enterprise agile framework*** yang merupakan *scaling agile framework* sampai level *enterprise*. The *Essence of software engineering* digunakan sebagai rujukan dasar sebagai *framework* penelitian. *The Essense* membuat *common ground* pada level dasar (*kernel*), sedangkan penelitian ini membuat *common ground* pada level praktik. *Common ground* sendiri dikembangkan menjadi suatu model konseptual yang merujuk kepada *general theory of software engineering*

2.1 Konsep *agile*

Istilah "*agile*" sangat umum dan banyak penafsiran yang berbeda-beda pada setiap individu dan komunitas (Axelos, Prince2 *agile*, 2018). Beberapa referensi menjelaskan tentang pengertian *agile* adalah suatu cara berpikir, bukan suatu metodologi seperti pengembangan perangkat lunak atau lainnya. (Boral, 2016). *agile* juga bisa diartikan sebagai kemampuan untuk berpikir dan bertindak cepat dengan koordinasi yang baik (Koch, 2005). Sebagai pembahasan lebih lanjut, Davis (2013) memperkenal istilah *big agile* dan *little agile*. *Big agile* adalah *agile* sebagai pola pikir, sedangkan *little agile* digambarkan sebagai penggunaan praktis dan metodologi di dalam menjalankan proyek berbasis *agile* (Davis, 2013).

2.2 Justifikasi Pemakaian *agile*

Tidak semua tipe proyek pengembangan perangkat lunak cocok menggunakan metode *agile*. Metode *waterfall* masih banyak dipergunakan untuk proyek-proyek yang memiliki kebutuhan atau spesifikasi yang sudah dapat ditentukan di awal (Phillips, 2019). Justifikasi pemakaian metode *agile* atau pun *waterfall* dapat merujuk kepada *stacey matrix* (PMI, 2017b), yang membuat kategori pemilihan metode pengembangan proyek berdasarkan kategori *uncertainty* atau ketidakpastian. *Uncertainty* dibedakan atas kebutuhan perangkat lunak dan teknologi. Untuk tipe-tipe proyek dengan *uncertainty* tinggi, maka metode *agile* lebih cocok untuk digunakan. Pendekatan *waterfall* masih lebih baik digunakan untuk tipe proyek dengan

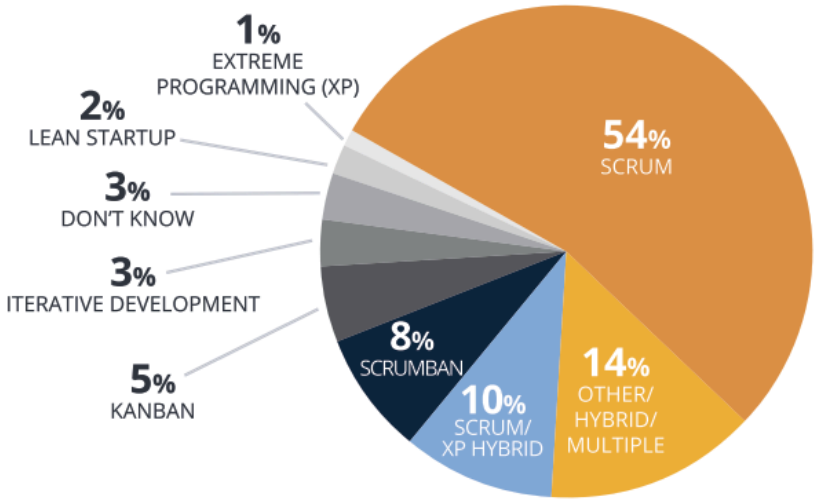
kebutuhan yang dapat diprediksi dengan jelas dan terperinci di awal, misalnya pembuatan aplikasi *core banking* yang sudah jelas kebutuhan dan aturan-aturan baku pada awal implementasi. Pada kebanyakan aplikasi yang melibatkan pengguna secara publik, misalnya *e-commernce*, pendekatan *agile* lebih banyak digunakan.

2.3 Metode *Agile*

Pada subbab ini dibahas beberapa metode *agile* yang populer seperti:

- Scrum
- Kanban
- Extreme Programming
- Agile Unified Process

Survei yang dilakukan oleh VersionOne (VersionOne, 2019) seperti pada Gambar 2.1 menunjukkan bahwa Scrum meruapakan metode *agile* yang paling populer, diikuti oleh Kanban, dan Extreme Programming. Beberapa responden memberikan tanggapan bahwa mereka tidak menyebutkan spesifik metode yang digunakan.

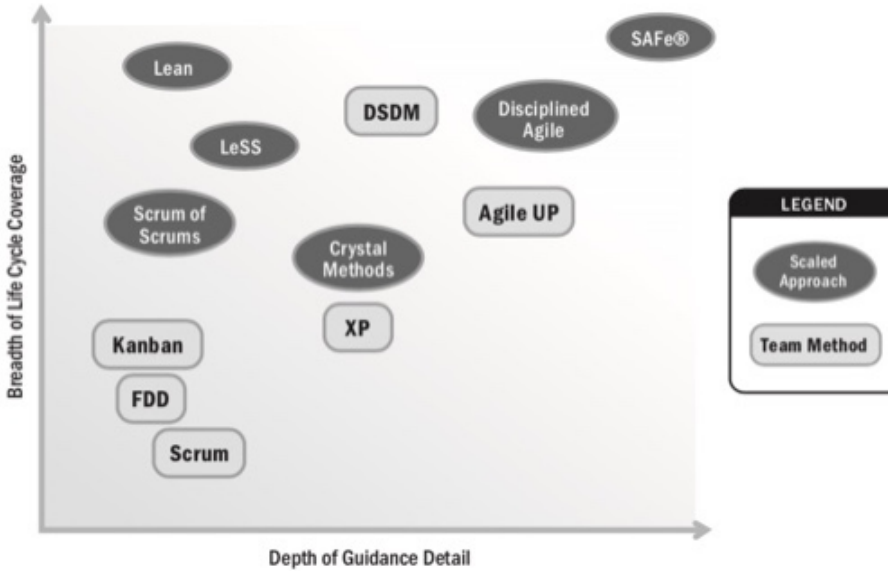


Gambar 2.1 Prosentase Penggunaan Metode *agile* (VersionOne, 2019)

2.4 Metode *Scaling agile*

Beberapa metode *agile* yang populer seperti Scrum, Kanban dan Extreme Programming dapat digunakan secara efektif untuk pembangunan produk pada tim yang kecil. Untuk level *enterprise*, metode tersebut perlu dilakukan *scale up* karena tidak dapat memecahkan masalah yang lebih kompleks. Project Management Institute (PMI, 2017b) membuat kategori metode

agile berdasarkan level dari besaran tim, seperti ditunjukkan pada Gambar 2.2. Beberapa ilmuwan di bidang *agile*, seperti Dean Leffingwell dan Scott Ambler mengembangkan metode yang bersifat *scaling* yang mereka namakan *Scaled agile framework* (D. Leffingwell, 2020) and *Discipline agile delivery* (PMI, 2019). Tidak ada penamaan secara formal mengenai metode *scaling* ini. Beberapa sumber referensi menyebut sebagai *enterprise agile framework*, *scaling* methods and *approach* (VersionOne, 2019), *scaling framework* (Scrum.org, 2019), dan *scaling agile* methods (Alqudah & Razali, 2016). *Scaled agile framework* (SAFe), *Large-scaled Scrum* (LeSS), *Discipline agile Delivery* (DAD) dan Nexus merupakan beberapa metode *scaling agile* yang sangat populer berdasarkan jumlah pemakainya (VersionOne, 2019), (Scrum.org, 2019).



Gambar 2.1 Klasifikasi Metode *agile* (PMI, 2017b)

2.5 Enterprise Agility & Business Agility

Pada metode *agile*, dikenal beberapa kategori seperti tim *agile*, *enterprise agility* dan *business agility* (Rose, 2018). Tim *agile* adalah metode praktik *agile* pada implementasi di level tim, seperti Scrum, XP, Kanban, FDD, dan DSDM (PMI, 2017b). *Enterprise agility* mengandung implementasi *agile* yang lebih besar dari tim *agile* pada organisasi yang saling bekerja sama untuk membangun satu produk besar. *Business agility* berhubungan dengan adopsi pola pikir dan prinsip-prinsip *agile* yang berhubungan dengan semua domain pada organisasi, termasuk yang di luar dari proses pengembangan perangkat lunak, seperti departemen sumber daya manusia, kepemimpinan, desain organisasi dan anggaran (Rose, 2018).

Kerangka kerja yang dihasilkan pada penelitian ini berhubungan dengan konsep *enterprise agile* dan juga tim *agile* yang dapat diterapkan pada organisasi di dalam membangun metode atau prosedur cara kerja mereka. Kerangka kerja dikaitkan dengan pertanyaan penelitian yang semuanya merupakan komponen dari pembentukan kerangka kerja tersebut. Organisasi dapat membentuk prosedur kerja sesuai dengan panduan yang terdapat pada kerangka kerja, yaitu comman ground, model, panduan implementasi dan dokumen *template*. Sedangkan bahasan mengenai *business agility* merupakan di luar ruang lingkup penelitian ini, karena terkait dengan disiplin ilmu yang berbeda di luar cakupun manajemen proyek dan pengembngan perangkat lunak.

2.6 Pengertian dan Proses Bisnis Enterprise

Pengertian *enterprise* pada penelitian ini merujuk pada definisi *enterprise* menurut (D.Leffingwell, 2020), di mana *enterprise* merupakan suatu fungsi yang memiliki tema strategi di dalam menjalankan bisnis proses yang terdiri dari level portofolio, program, dan proyek. *Portfolio backlogs* merupakan *backlogs* pada level *enterprise*, misalnya strategi dan capaian yang ingin dicapai oleh organisasi, misalnya perusahaan ingin membangun bisnis digital syariah pada bank. *Portfolio backlogs* didefinisikan oleh *epic* yang melibatkan *key process improvement* (KPI) yang ingin dicapai oleh organisasi. Portfolio backlog diturunkan menjadi *program backlog*s. *Progam backlogs* ini yang merupakan penjabaran untuk *scaling agile*. *Program backlogs* pada level proyek akan dijabarkan pada *project backlogs* dan *team backlogs.*

2.7 *Software Engineering Theory*

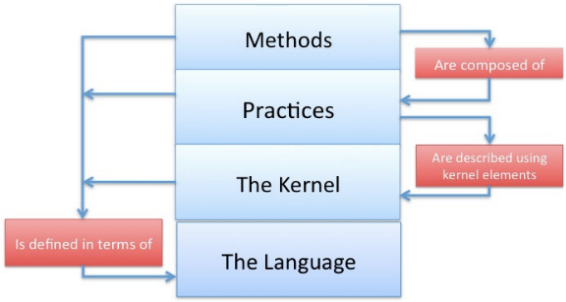
Penulis memaparkan bahasan mengenai teori secara umum dan dalam aspek software engineering pada bahasan subbab di bawah ini. Bahasan mengambil referensi dari (D.I. Sjøberg, T. Dybå, B.C. Anda, 2008) pada tulisan yang berjudul Building Theories in Software Engineering. Referensi ini banyak digunakan pada penelitian sebelumnya seperti (Johnson & Ekstedt, 2016) dan (Bjarnason, Smolander, Engström, & Runeson, 2016).

2.8 *The Common ground* pada *Software Engineering*

Istilah the *common ground* pada bidang rekayasa perangkat lunak diperkenalkan oleh Ivaar Jacobson (Jacobson et al., 2012). Konsep ini kemudian dinamakan sebagai the Essense of Kernel for software engineering. The *common ground* dibuat berdasarkan pemikiran bahwa tidak ada metode pengembangan perangkat lunak yang cocok untuk semua orang. Kekuatan dari *common ground* yaitu menyediakan kerangka kerja yang bersifat umum untuk memahami konsep umum dari metode pengembangan perangkat lunak. The *common ground* dapat digunakan untuk media berbagi pengetahuan, peningkatan proses, perbandingan dari metode dan praktik dari pengembangan perangkat lunak. Pendekatan the *common ground* untuk pengembangan perangkat lunak ini menjadi inspirasi penulis untuk mengembang the *common ground* untuk metode *scaling agile*.

2.9 The Essence – The SEMAT Kernel

The Essence merupakan suatu standar pada rekayasa perangkat lunak yang menyediakan bahasa yang universal untuk menentukan metode dan praktik (OMG, 2018). The Essence dibuat pada tahun 2014 oleh Software Engineering Methods and Theory (SEMAT) (Semat.org, 2020) dan Object Management Group (OMG). The Essence dibuat dengan pemikiran perlu adanya standarisasi atau *common ground* dari metode-metode yang ada sebelumnya. The Essence terdiri dari metode, praktik, the Kernel dan Bahasa yang universal. Metode terdiri dari beberapa praktis. Praktis merupakan pendekatan yang berulang untuk melaksanakan aktivitas dengan tujuan tertentu. Praktis dijelaskan menggunakan *kernel*, sedangkan *kernel* didefinisikan oleh universal *language*. Hubungan antar metode, praktik, Kernel dan *language* dijabarkan pada Gambar 2.3.



Gambar 2.3 SEMAT Kernel (Jacobson et al., 2012)

2.10 Penelitian Sebelumnya

An empirically-developed framework for agile transition and adoption: A Grounded Theory approach. The Journal of Systems and Software (Gandomani & Nafchi, 2015).

Penelitian ini memiliki latar belakang mengenai transisi dan adopsi kerangka kerja *agile* menjadi hal yang diusulkan untuk organisasi, namun masih terdapat banyak tantangan di dalam implementasinya. Untuk proses transisi dan adopsi tersebut diperlukan *overhead* organisasi karena berhubungan dengan struktur yang komplek dan kurang fleksibel. Tujuan dari penelitian ini membangun kerangka kerja untuk adopsi dan transisi *agile* pada organisasi. Kerangka kerja yang dihasilkan tidak tergantung dari besaran organisasi. Penelitian ini juga menggambarkanlangkah-langkah pada kerangka kerja untuk membantu organisasi di dalam mencapai transformasi *agile*.

A Review of Scaling agile Methods in Large Software Development. (Alqudah & Razali, 2016). International Journal on Advanced Science Engineering Information Technology.

Penelitian ini memiliki latar belakang mengenai banyak metode *agile* secara literatur, seperti Dynamic Systems Development Method (DSDM), Extreme Programming (XP), SCRUM, *agile* Modeling (AM) and Crystal Clear. Metode tersebut cocok digunakan untuk level tim. Untuk level *scaling* diperlukan modifikasi dari metode-metode tersebut. Metode *scaling agile* yang ada yaitu DAD, LeSS, LeSS huge, SAFe, Spotify, Nexus, and RAGE dapat dipergunakan untuk organisasi yang melakukan implementasi *agile* pada level yang lebih besar. Penelitian ini bertujuan untuk melakukan review terhadap metode *scaling agile* yang ada. Metode penelitian menggunakan literature review.

Implementing Large-Scale agile Frameworks: Challenges and Recommendations. (Conboy & Carroll, 2019). IEEE Software

Penelitian memiliki latar belakang mengenai tantangan implementasi *scaling agile*. Penelitian ini menggunakan data dari 13 proyek implementasi *scaling agile* pada perusahaan multinasional. Tantangan yang dihadapi yaitu misalnya mendefinisikan kerangka kerja *Large Scaled agile*, membandingkan dan memilih kerangka kerja *agile* yang sudah tersedia, dan kesiapan mengenai implementasi kerangka kerja *scaling agile.* Tantangan lainnya disebutkan pada bab 1. Penelitian yang digunakan menggunakan metode survei pada organisasi yang menerapkan *scaling agile*, seperti: Accenture, IrishBank, Dell, Intel, dan lain-lain. Tujuan dari

penelitian ini untuk memecahkan masalah dan memberikan rekomendasi pada implementasi *scaling agile*.

The Tarpit – A general theory of software engineering. Information and Software Technology. (Johnson & Ekstedt, 2016)

Penelitian ini bertujuan untuk membuat *general theory of software engineering* (GTSE). General theory ini bertujuan untuk memprediksi beberapa fenomena pada disiplin ilmu software engineering. Metode yang dipergunakan menggunakan pendekatan pembuatan teori berdasarkan teori yang ada yaitu: *language* dan *automata*, *cognitive architecture*, *problem solving*, dan stuktur organisasi.

Penelitian ini mempunyai kontribusi terhadap penelitian yang dilakukan oleh penulis sebagai referensi utama di dalam pembangunan kerangka kerja *scaling agile*. Penelitian penulis menggabungkan antara pendekatan the Essense of software engineering dan GTSE. The Essense of Software Engineering menggunakan pendekatan yang lebih kuat ke arah *best practice*, sedangkan GTSE lebih menekankan kepada pendekatan teoritis.

A theory of distances in software engineering. Information and Software Technology. (Bjarnason et al., 2016)

Tujuan dari penelitian ini untuk membangun suatu theory of distance yang menerangkan sebarapa besar suatu praktik dapat meningkatkan komunikasi selama berlangsungnya proyek berdasarkan dampak adanya perbedaan antar sumber daya, aktivitas dan artefak. *Distance* di antara sumber daya manusia atau orang bisa terdiri dari:

- Faktor geografis, misalnya jarak fisik antara orang-orang tersebut
- Faktor sosial budaya, misalnya perbedaan antar budaya pada orang-orang tersebut
- Temporal, misalnya perbedaan waktu
- Organisasi, misalnya perbedaan antar tujuan organisasi dan prioritas

A Conceptual Model of Risk Management (Almeida, Teixeira, Mira da Silva, & Faroleiro, 2019)

Penelitian ini menjadi salah satu rujukan bagi penulis di dalam pelaksaan penelitian berdasarkan metode DSR. Proses evaluasi pada penelitian ini (Almeida et al., 2019), akan menjadi salah satu rujukan di dalam pelaksanaan demonstrasi, yaitu dengan:

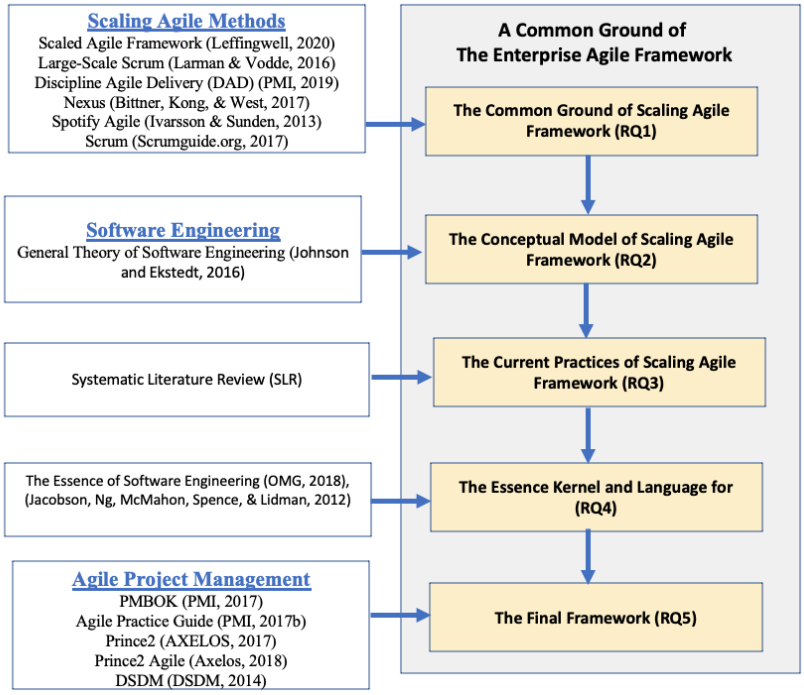
1. Melakukan interviu dengan 12 orang specialist dengan banyaknya pengalaman pada bidang sesuai dengan obyek penelitian.
2. Instrumen interviu menggunakan konfirmasi dengan skala Likert dengan rentang nilai: 1 sangat tidak setuju, 2 tidak setuju, 3 tidak setuju atau setuju, 4 setuju, 5 sangat setuju digunakan untuk melakukan penilaian terhadap model yang dibuat
3. Kesepakatan terhadap skala Likert ditunjukkan dengan diagram batang

Using agile methodologies for adopting COBIT (Amorim, Mira da Silva, Pereira, & Gonçalves, 2020)

Penelitian ini menjadi salah satu rujukan bagi penulis di dalam pelaksaan penelitian berdasarkan metode DSR. Proses evaluasi pada penelitian ini (Almeida et al., 2019), akan menjadi salah satu rujukan di dalam pelaksanaan demonstrasi,

2.11 *Theoretical Framework*

Gambar 2.4 menunjukkan *theoretical framework* pada penelitian ini untuk membangun kerangka kerja untuk *enterprise agile*. Studi literatur yang dilaksanakan secara intensif dari populer *scaling agile* methods seperti SAFe, LeSS, DAD, dan Nexus dipergunakan sebagai dasar untuk membangun *common ground*. Hasil dari studi literatur ini menghasilkan *command practice* dari *scaling agile methods* yang merupakan luaran dari RQ1.



Gambar 2.4 Kerangka Teoretis

2.12 Metodologi Penelitian *Design Science*

Salah satu pengertian mengenai metodologi penelitian adalah istilah yang menjelaskan strategi yang digunakan untuk menjawab pertanyaan penelitian tertentu (Recker, 2013). Creswell (2013) menjelaskan strategi tersebut sebagai rancangan kualitatif, kuantitatif dan *mixed-methods*. Recker (2013) menambahkan satu strategi lagi yaitu design science methods, yang didefinisikan sebagai *"Design Science Methods are procedures that feature methods to build and evaluate novel and innovative artefacts (such as new models, methods or systems) as the outcome of a research process and which are characterised by an emphasis on the construction of the artefact and the demonstration of its utility to an organisational problem"*. Pada beberapa literatur *design science method* ini sering juga disebut sebagai *design science research* (DSR).

Aktivitas pada design science dijelaskan sebagai: (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007)

Activity 1. Problem identification and motivation.

Activity 2. Define the objectives for a solution

Activity 3. Design and development.

Activity 4. Demonstration.

Activity 5. Evaluation.

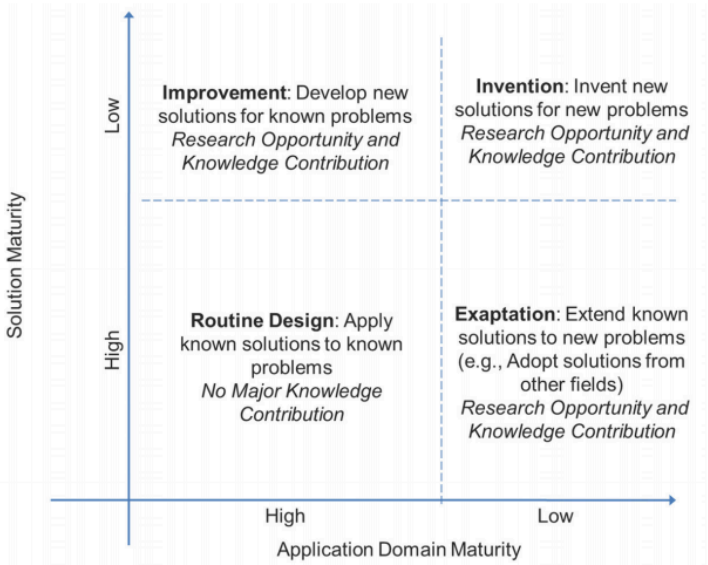
Activity 6. Communication

Kontribusi Penelitian Design Science

Kontribusi penelitian Design Science dapat dikategorikan pada *maturity* dari artefak yang dihasilkan (Gregor & Hevner, 2013). Gambar 2.5 dan 2.6 menunjukkan jenis kontribusi dari penelitian dan contoh dari artefak yang dihasilkan. Tipe kontribusi pada penelitian ini berada pada level 2, yaitu pembangunan artefak berupa kerangka kerja yang berisikan metode.

	Contribution Types	Example Artifacts
More abstract, complete, and mature knowledge ↕ ↕ ↕ ↕	Level 3. Well-developed design theory about embedded phenomena	Design theories (mid-range and grand theories)
	Level 2. Nascent design theory—knowledge as operational principles/architecture	Constructs, methods, models, design principles, technological rules.
More specific, limited, and less mature knowledge	Level 1. Situated implementation of artifact	Instantiations (software products or implemented processes)

Gambar 2.5 Tipe Kontribusi pada Penelitian Design Science (Gregor & Hevner, 2013)



Gambar 2.6 *DSR Knowledge Contribution Framework* (Gregor & Hevner, 2013)

Beberapa penelitian (Recker, 2013) menyebutkan bahwa DSR memiliki sifat generalisasi yang rendah. Argumen ini disampaikan dengan merujuk kepada alasan bahwa DSR memiliki domain problem yang bersifat local. Dengan merujukan kepada konsep kontribusi penelitian DSR (Gregor & Hevner, 2013), konsep generalisasi ini tergantung kepada tipe kontribusi dan kuadran kontribusi. Untuk sifat generalisasi yang rendah, ini masuk pada tipe kontribusi level

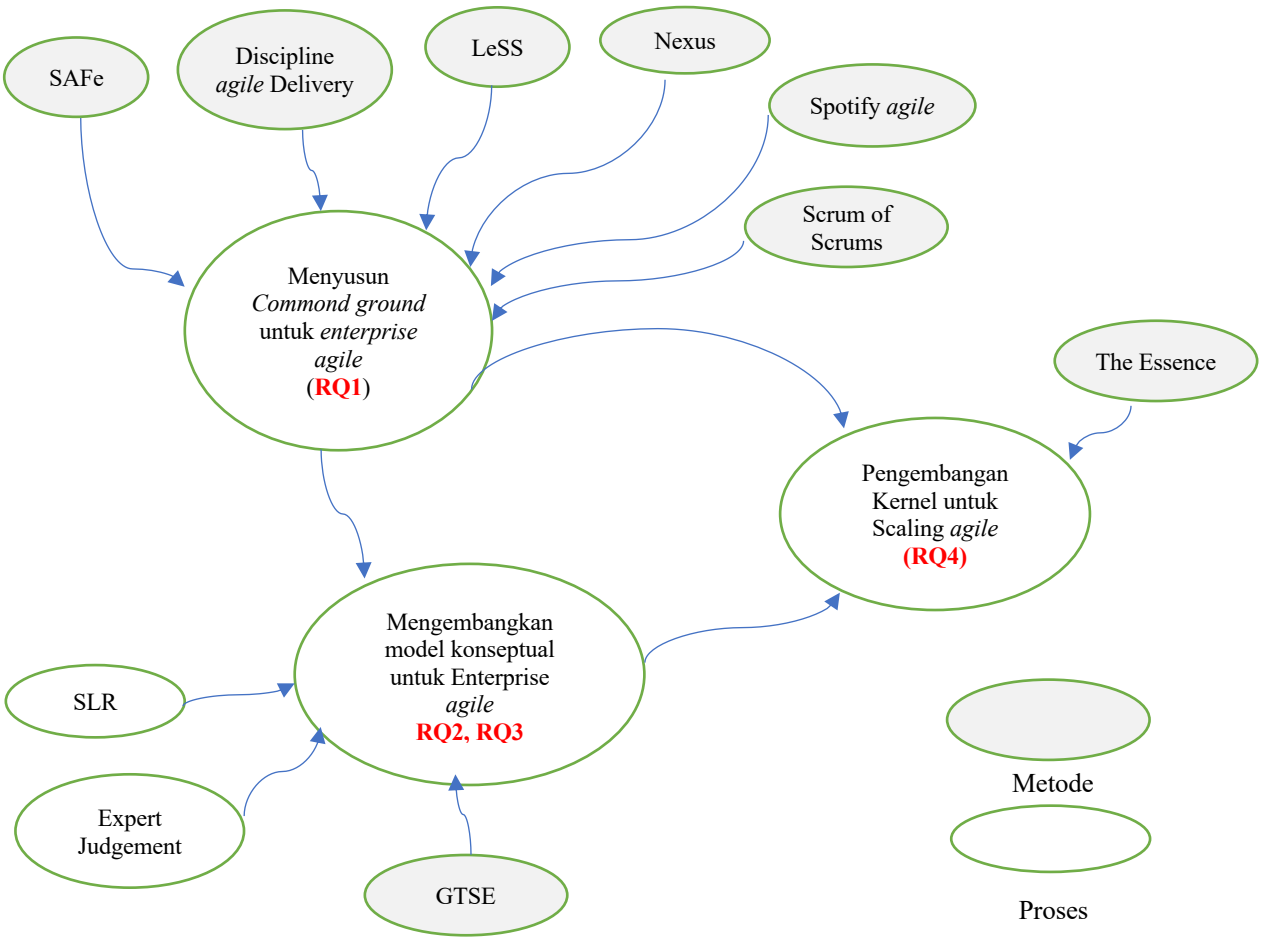
1 dan kontribusi pada kuadran Routine Design. Untuk kontribusi yang lebih general, dapat dikaitkan dengan jenis kontribusi level 2 atau 3.

BAB 3
METODOLOGI PENELITIAN

Bab ini menjelaskan berkaitan dengan metodologi penelitian yang berisi alur pikir penelitian, tahapan penelitian, desain penelitian, proses olah data, instrumen penelitian, dan jadwal peneltian.

3.1 Alur Pikir Penelitian

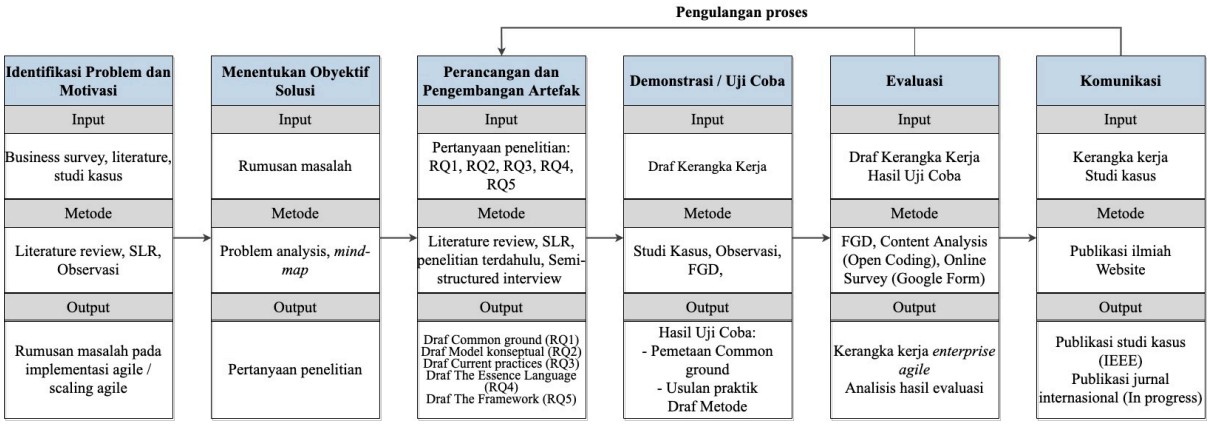
Seperti dijelaskan pada bab 2, penelitian ini disusun berdasarkan ide dari the *common ground of the software engineerring* (Jacobson et al., 2012). Alur pikir penelitin ini seperti ditunjukkan pada Gambar 3.1, pada tahap awal penelitian disusun the *common ground* untuk metode *scaling agile*. The *common ground* bersumber pada metode *scaling agile* yang sudah ada, yaitu: Scaled *agile* Framework (SAFe), Large Scaled Scrum (LeSS), Nexus, Discipline *agile*, Spotify *agile* dan Scrum of Scrums. The *common ground* pada metode *scaling agile* ini akan menjawab pertanyaan penelitian 1.



Gambar 3.1 Alur Pikir Penelitian

3.2 Tahapan Penelitian

Tahapan penelitian menggunakan metode Design Science yang merujuk pada (Peffers et al., 2007) dan (Johannesson & Perjons, 2014). Tahapan penelitian ditunjukkan pada Gambar 3.2.



Gambar 3.2 Tahapan Penelitian DSR

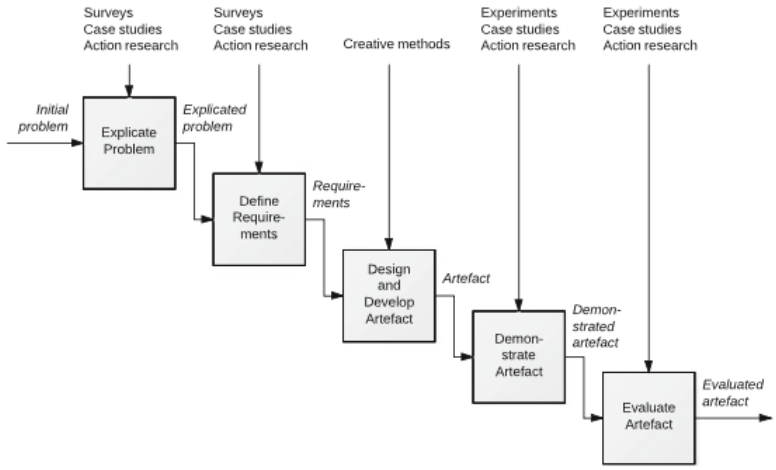
Penerapan DSR dilakukan dengan merujuk kepada metode DSR menurut (Peffers et al., 2007) dan (Johannesson & Perjons, 2014). Terdapat tahapan-tahapan utama di dalam pelaksanaan DSR, yaitu:

1. Identifikasi masalah
2. Menentukan tujuan pembuatan solusi
3. Desain dan pengembangan
4. Demonstrasi
5. Evaluasi
6. Komunikasi

Pada tahap demonstrasi *artefact* yang merupakan salah satu tahapan pada DSR, akan dilakukan kegiatan studi kasus berupa *proof of concept* untuk pembuatan kerangka kerja *enterprise agile* pada perusahaan media daring dan manufaktur farmasi di Indonesia. Setelah *proyek studi kasus* selesai, tahapan berikutnya adalah evaluasi *artefacts* yang juga merupakan salah satu tahapan penting pada DSR. Evaluasi DSR dilakukan dengan metode *semi-structure interview* pada pemangku kepentingan di *case study project* tersebut. *Feedback* yang dihasilkan akan dianalisis untuk ditindak lanjuti apakah berpengaruh pada desain rrtefact atau hanya pada tahapan demonstrasi artefacts. Proses demonstrasi dengan *proof of concept* ini mengambil

rujukan pada (Johannesson & Perjons, 2014), dan evaluasi artifacts dengan *case study* merujuk pada penelitian DSR sebelumnya (Peffers et al., 2007).

Aktivitas secara lebih rinci mengenai tahapan merujuk kepada (Johannesson & Perjons, 2014) sebagai: *"In large design science projects, it is common to use several research strategies and methods, because different design science activities may require different approaches"* seperti dicontohkan pada Gambar 3.3.



Gambar 3.3 Contoh Penerapan Berbagai Strategi Reserach pada DSM (Johannesson & Perjons, 2014)

Penulis menerapkan beberapa strategi research lainnya pada tahapan penelitian, seperti:

- *Literature review*: sebagai landasan teori
- SLR: mendapatkan problem, mendapatkan current practice
- *Thematic analysis* dengan *coding* menggunakan Nvivo: untuk membangun *common practice*
- *Semi-structure interview:* untuk proses evaluasi desain
- *Case study (proof of concept):* untuk demonstrasi *artefact*

BAB 4
HASIL PENELITIAN

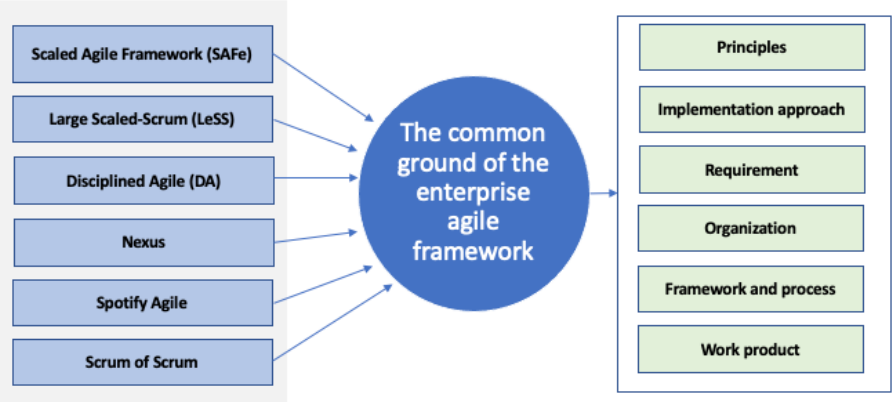
Bab ini menjelaskan hasil penelitian berdasarkan metode DSR. Materi yang dibahas meliputi pembuatan artefak berdasarkan pertanyaan penelitan RQ1, RQ2, RQ3, RQ4, dan RQ5. Hasil penelitian pada bab ini berdasarkan metode SLR bukanlah artefak akhir, tetapi merupakan draf artefak. Artefak akhir didapatkan melalui serangkaian proses demonstrasi atau uji coba, dan evaluasi. Hasil penelitian ini juga didasarkan pada tahapan penelitian seperti dijelaskan pada

4.1 Desain dan Pengembangan Artefak – RQ1

Subbab ini membahas desain dan pengembangan artefak pada langkah-langkah DSR, seperti ditunjukkan oleh kotak merah pada Gambar 4.1. Pengembangan artefak ini adalah pembuatan *common ground* untuk kerangka kerja *enterprise agile*.

The *common ground of the enterprise agile framework* dibangun menggunakan metode *literature review* dari metode *scaling agile* yang populer, yaitu:

- *Scaled agile Framework (SAFe)*
- *Large Scaled-Scrum (LeSS)*
- *Disciplined agile Delivery (DAD)*
- *Nexus*
- *Spotify agile*
- *Scrum of Scrum*



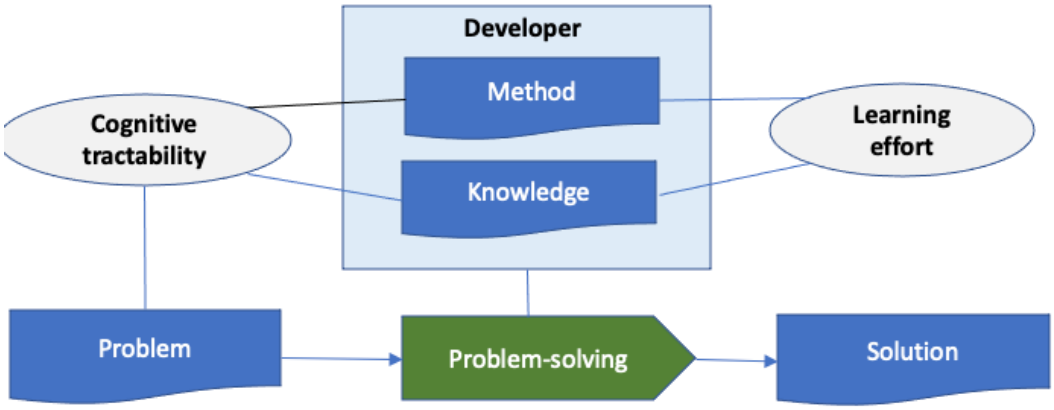
Gambar 4.1 Common ground untuk *scaling agile* framewrok

4.2 Desain dan Pengembangan Artefak – RQ2

Desain dan pengembangan artefak untuk RQ2 ini adalah model konseptual dari *enterprise agile framework.* Model konseptual ini merupakan pengembangan dari hasil RQ1, yaitu membuat hubungan antara komponen *common ground*.

Model Konseptual untuk Enterprise agile Framework

Beberapa penelitian mengungkapkan bahwa teori umum (*general theory*) memainkan peranan yang sangat penting di dalam pengembangan ilmu pengetahuan. Disiplin ilmu lainnya, mulai dari fisika hingga ekonomi semuanya memiliki landasan teori yang kuat. Pada area pengembangan perangakat lunak, teori Tarpit diusulkan sebagai teori umum yang dapat menjelaskan dan memperkirakan fenomena di dalam dunia perangkat lunak (Johnson & Ekstedt, 2016). Teori lainnya pada area pengembangan perangkat lunak yaitu *Software Engineering Body of Knowledge* (Bourque & Fairley, 2014) and *The Theory of Distance in Software Engineering* (Bjarnason et al., 2016). Teori Tarpit dibangun berdasarkan teori yang mapan, yaitu: bahasa dan *automata*, arsitektur kognitif, pemecahan masalah, dan stuktur organisasi. gambaran umum teori Tarpit ditunjukkan pada Gambar 4.2. Ada pengembang yang melakukan pekerjaan pemecahan masalah untuk menghasilkan solusi. Pengembangan memiliki metode dan pengetahuan yang memiliki kemampuan kognitif dan pembelajaran sebagai kemampuan dan ketrampilannya.



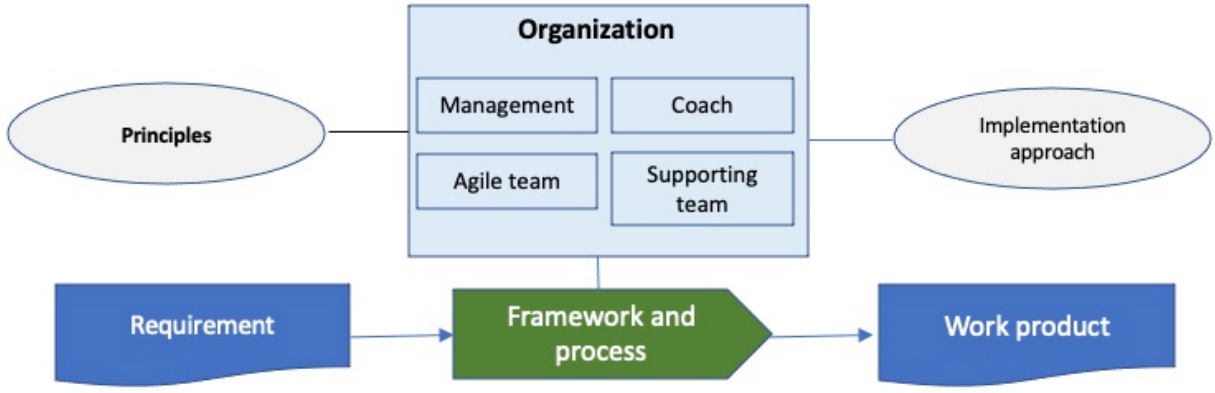
Gambar 4.2 Model konseptual pada Tarpit General Theory of Software Engineering (Johnson & Ekstedt, 2016)

Model konseptual pada penelitian ini yaitu model konseptual untuk *enterprise agile framework*, seperti ditunjukkan pada Gambar 4.8 diadopsi dari teori umum dari pengembangan

perangkat lunak (GTSE). Berdasarkan penelusuran *common ground* dari *enterprise agile framework* yang ada saat ini, terdapat enam komponen yang dapat dipetakan pada model yaitu:

1. Prinsip
2. Organisasi
3. Strategi implementasi
4. Kebutuhan
5. *Framework* dan proses
6. *Work product*

Gambar 4.3 menjelaskan hubungan antara enam komponen tersebut. Komponen prinsip memberikan panduan kepada organiasi di dalam melakukan implementasi *scaling agile*. Komponen organisasi terdiri dari stuktur tim baik internal mau pun eksternal yang berperan di dalam melakukan eksekusi atau dukungan di dalam implementasi *agile*. Strategi implementasi lebih ke arah perencanaan dan ruang lingkup yang akan dikerjakan. Tim *agile* yang berada di dalam organisasi melakukan implementasi kebutuhan pemakai melalui *framework* dan proses yang didefiniskan untuk menghasilan *deliverable* atau *work product*.



Gambar 4.3 Model Konseptual *Scaling Agile*

4.3 Desain dan Pengembangan Artefak – RQ3

Subbab ini menjelaskan mengenai pembentukan *common practices* melalui penggunaan SLR untuk mendapatkan praktik-praktik yang dilakukan oleh organisasi berdasarkan penelitian studi kasus sebelumnya. Praktik-praktik tersebut diharapkan dapat menerangkan secara lebih terperinci mengenai model dan *common ground* untuk *scaling agile*.

Tabel 4.1 *Current Practice* berdasarkan *literature*

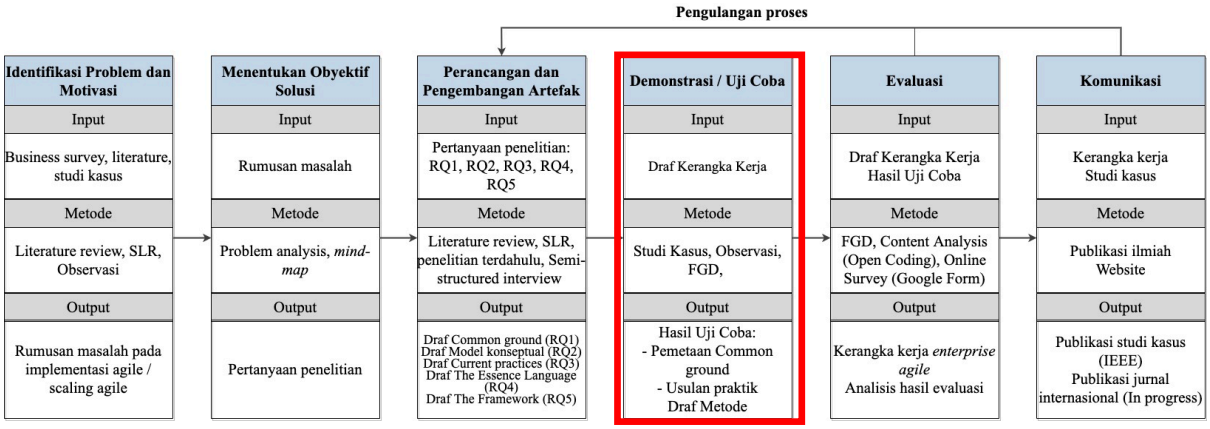
The common ground	***Current practices* berdasarkan literatur**
Principles	• *Identify the success factors, lesson learn, best practice, Lean approach*
	• *The need of management and sponsor support*
	• *Create internal principles*
Framework and Process	• *Following standard SAFe process*
	• *Conduct training*
	• *Develop a method by mapping the requirements of security standards into an agile process model*
	• *To utilize the integrated system architecture*
	• *Develop a tools for PI planning*
	• *Adapt Scrum-of-Scrum*
	• *The team is distributed to handle dedicated features*
	• *Program management need to coordinate the team*
	• *Provide community of practice*
	• *Aligning Requirements Engineering and System Testing*
	• *Two stage effort estimation can improve accuracy*
Organization	• *The use of external coach and change agent*
	• *Implement the extended team: Software architecture team, virtual maintenance team, managers*
	• *provide guild across the functions, agile guild, core guild, back end guild, web guild*
	• *Implement enterprise coach*
	• *local Product Owner at each site*
	• *The need of agile coach and agile champion for agile transition*
	• *Provide informal community of practice*
	• *Scaling Product Ownership Through Team Alignment and Optimization*
	• *Develop the agile Center of Excellence*
	• *The involvement of project director to manage the external relations*
Implementation approach	• *Combined LeSS with SAFe to have a superordinate portfolio level*
	• *Implement maturity model*
	• *Standardize the process*
	• *The implementation of SAFe at level team, program and portfolio level*
	• *Utilize SAFe and DAD to mitigate risks in GSD*
	• *Tailoring from various approach*
	• *Implement ScrumBut. Changes to the Scrum practices should be evaluated based on organizational consideration*
	• *define a framework for regulated Scrum practices*
	• *Develop owned practices and framework*
	• *Combine Design Thinking with agile practice*
	• *Using Scrum based for adopting COBIT 5*

4.4 Proses Evaluasi RQ1, RQ2, dan RQ3

Proses evaluasi untuk RQ1, RQ2, dan RQ3 dilakukan secara bersamaan karena ketiga RQ tersebut memiliki keterkaitan satu sama lain yaitu pada pembahasan *common ground*. Proses evaluasi bertujuan untuk mendapatkan umpan balik mengenai persetujuan model konseptual dan konfirmasi praktik yang ada serta mendapatkan pengayaan dari para pakar dan praktisi mengenai praktik yang mereka lakukan pada organisasi. Pada akhir evaluasi para pakar akan dimintakan pendapat mereka melalui skala Likert untuk kofirmasi persetujuan pada model konseptual (RQ2).

BAB 5
DEMONSTRASI ATAU UJI COBA PENELITIAN

Demonstrasi penelitian merupakan salah satu langkah dari metode DSR sebagai metodologi pada penelitian ini. Demonstrasi pada penelitian ini adalah uji coba artefak draf kerangka kerja pada beberapa perusahaan di Indonesia sebagai tempat studi kasus. Tahapan ini dilakukan setelah aktivitas desain artefak, ditunjukkan pada Gambar 5.1 di bagian kotak merah.



Gambar 5.1 Tahapan Penelitian DSR

5.1 Metode Demonstrasi atau Uji Coba yang Dipergunakan

Metode demonstrasi atau uji coba pada penelitian ini dibagi menjadi dua, yaitu:

- Uji coba secara *proof of concept*
- Uji coba penelitian studi kasus

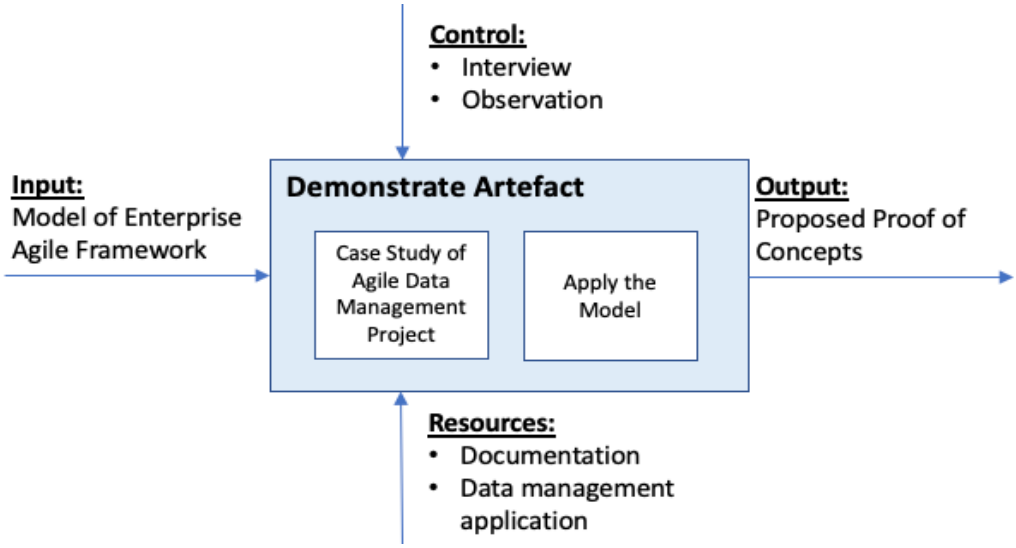
5.1.1 Uji Coba secara *Proof of Concept*

Uji coba dengan *proof of concepts* dilakukan menggunakan *template* dokumen yang disediakan pada penelitian ini (Lampiran 2), berupa berkas dalam bentuk Power Points atau pun dokumen Microsoft Word. Para praktisi dapat menggunakan secara langsung dokumen *template* tersebut di dalam melakukan proses:

- Analisis kondisi saat ini
- Pemetaan praktik saat ini pada *common ground*
- Pemilihan praktik
- Pembentukan prosedur
- Penggunaan *the essence language* (opsional)

5.1.2 Uji Coba Penelitian Studi Kasus

Demonstrasi pada penelitian ini merujuk kepada (Johannesson & Perjons, 2014) seperti ditunjukkan pada Gambar 5.2. Kerangka kerja yang dihasilkan pada penelitian ini akan menjadi masukan untuk proses demonstrasi artefak. Proses kontrol berupa wawancara, observasi atau pun *focus group discussion* (FGD) dilakukan sebagai masukan untuk menghasilkan luaran. Dokumen-dokumen tempat studi kasus, seperti prosedur atau proses lainnya juga digunakan sebagai pelengkap untuk menghasilkan luaran.



Gambar 5.2 Model Uji Coba

Penerapan studi kasus secara utuh dilakukan melalui proses penelitian pada studi kasus:

- Perusahaan media daring sebanyak dua studi kasus
- Perusahaan manufaktur
- Perusahaan konsultan internasionalpada implementasi proyek pada pelanggannya
- Perusahaan asuransi swasta nasional
- Perusahaan perbankan nasional sebanyak 2 studi kasus
- Perusahaan *software house* nasional

Pada uji coba ini, penulis memberikan panduan kerangka kerja melalui pemaparan *literature* yang digunakan pada penelitian ini, yaitu:

- Konsep *the essence* yang disesuaikan dengan kondisi studi kasus

- Pemilihan praktik *agile* yang berdasarkan kebutuhan organisasi. Pemilihan praktik seperti yang dilakukan pada penelitian ini, yaitu dengan melakukan studi pustaka mengenai metode *agile* dan *scaling agile* yang ada saat ini
- Pembentukan prosedur
- Metode penelitian DSR
- Panduan implementasi

5.2 Tempat studi kasus

Tempat studi kasus yang ditentukan untuk uji coba atau demonstrasi kerangka kerja ini adalah:

- Perusahaan media daring sebanyak dua studi kasus
- Perusahaan manufaktur
- Perusahaan konsultan internasionalpada implementasi proyek pada pelanggannya
- Perusahaan asuransi swasta nasional
- Perusahaan perbankan nasional sebanyak 2 studi kasus
- Perusahaan *software house* nasional

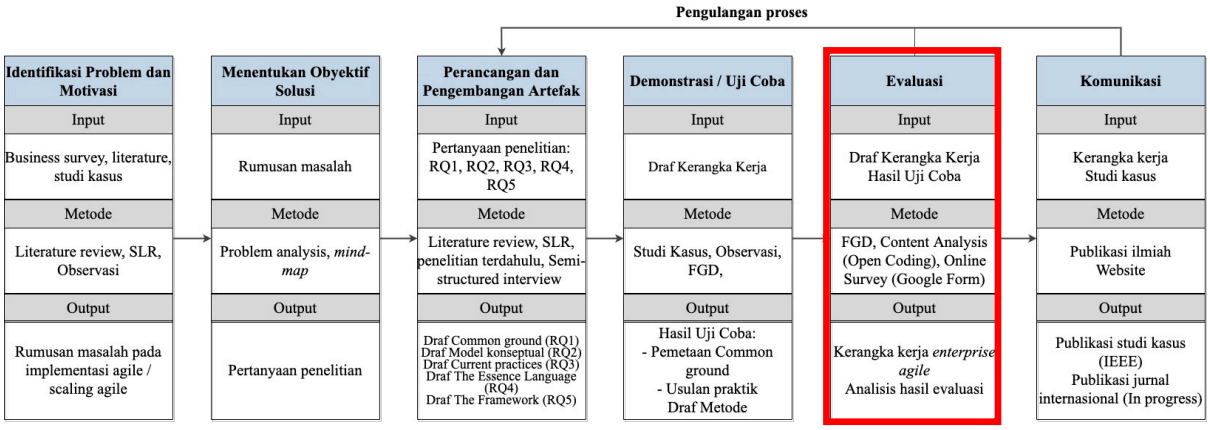
Proses demonstrasi atau uji coba dengan pendekapan *proof of concept* juga dilakukan pada beberapa perusahaan lainnya dengan uji coba langsung draf kerangka kerja yang telah dibuat untuk studi kasus tertentu. Draf kerangka kerja dapat didapatkan pada tautan berikut: https://bit.ly/Scaling*agile*Framework. Beberapa contoh uji coba diterapkan pada beberapa bagian organisasi, seperti:

- Industri perbankan
- Industri *e-commerce*
- Perusahaan *vendor*
- Perusahaan yang bergerak pada bidang energi
- Perusahaan lainnya dengan total 34 studi kasus

BAB 6
EVALUASI DAN ANALISIS HASIL PENELITIAN

Bab ini membahas mengenai evaluasi dan analisis hasil penelitian yang dilakukan pada bab sebelumnya. Evaluasi merupakan salah satu langkah dari metode DSR sebagai metodologi penelitian ini. Tahapan ini dilakukan setelah aktivitas desain demonstrasi atau uji coba dari artefak seperti ditunjukkan pada Gambar 6.1 di bagian kotak merah.

Bab ini juga melakukan analisis dengan membandingkan hasil penelitian dengan kerangka kerja yang sejenis. Proses validasi dengan menggunakan kriteria teori pada software engineering juga dilakukan untuk menguatkan justifikasi validitas penelitian ini.



Gambar 6.1 Penelitian *design science research*

6.1 Evaluasi Akhir Tahap Pertama RQ1, RQ2, RQ3, dan RQ4

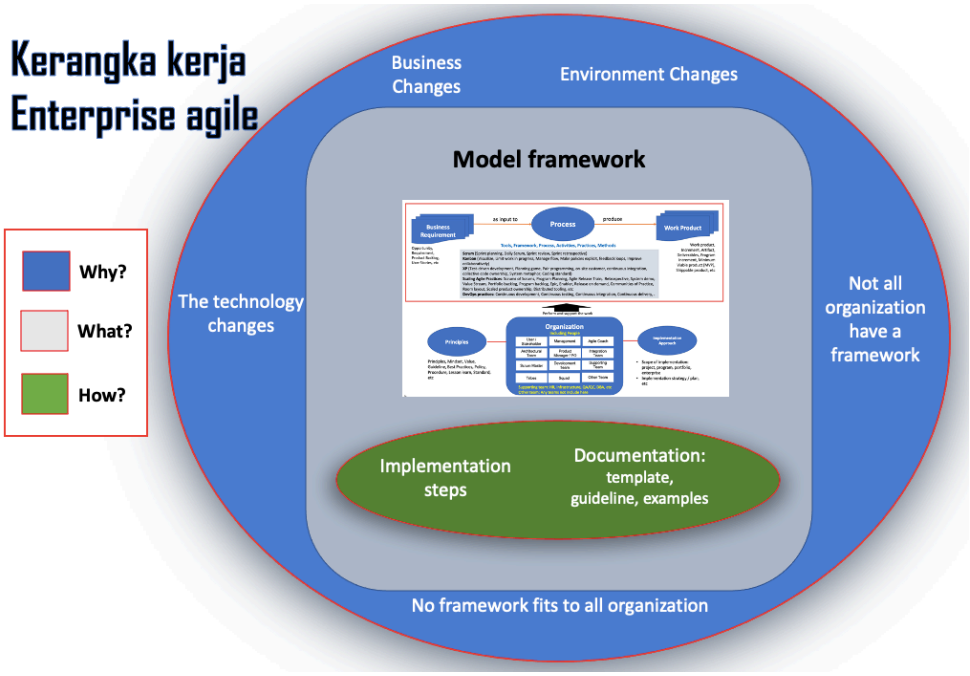
Proses evaluasi mengacu kepada langkah-langkah pada penelitian DSR. Proses ini dilakukan setelah tahapan proses demonstrasi. Tahapan wawancara ini dilakukan setelah pelaksanaan demonstrasi pada organisasi pada tanggal 28 Juni 2021 kepada para praktisi yang memiliki pengalaman pada proyek *agile*. Proses pembaruan pada artefak dilakukan sesuai dengan umpan balik pada proses evaluasi tahap sebelumnya.

6.2 *Enterprise Agile Framework*

Subbab ini menjelaskan mengenai makna kerangka kerja atau *framework* pada penelitian ini serta susunan kerangka kerja yang terdiri dari *common ground*, model kerangka kerja, rencana implementasi dan panduan pelaksanaan.

6.2.1 Makna *Framework* pada Penelitian

Framework pada penelitian ini merujuk pada pemahaman dan definisi *framework* seperti dijelaskan pada subbab 2.16. *Enterprise agile framework* (EAF) merupakan scaling *agile framework* sampai pada level *enterprise*. EAF ini merupakan komponen yang dibentuk untuk menjawab mengenai *why*, *what* dan *how*.



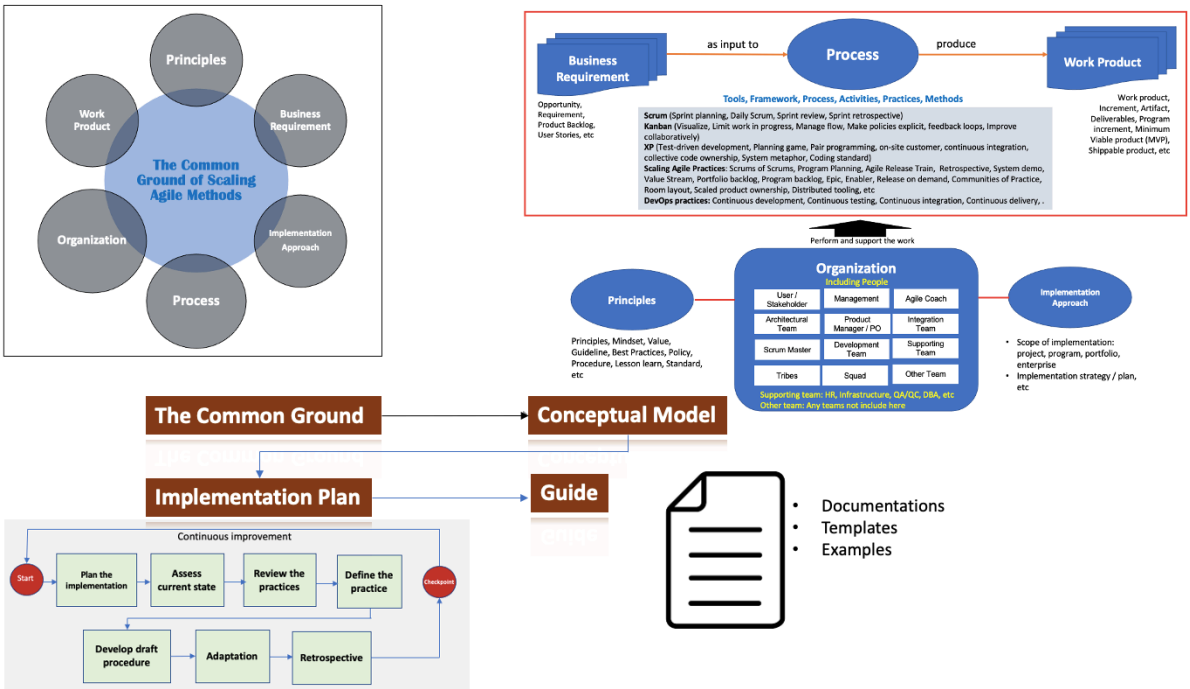
Gambar 6.2 Komponen *Enterprise Agile Framework*

Pertanyaan *why* digambarkan pada lingkaran berwarna biru, yaitu latar belakang diperlukannya kerangka kerja ini. Pertanyaan what dijawab dengan bentuk oval berwarna abu-abu, yaitu model kerangka kerja *enterprise agile* yang terdiri dari *common ground*. Implementasi dari kerangka kerja ini digambarkan oleh bentuk oval berwarna hijau, yaitu langkah-langkah implementasi, dokumentasi, *template*, *guideline* dan contoh-contoh. Penjelasan masing-masing komponen kerangka kerja dijabarkan lebih lanjut pada subbab di bawah ini.

6.2.2 Susunan Kerangka Kerja

Kerangka kerja *enterprise agile* seperti ditunjukkan pada Gambar 6.3 merupakan gabungan dari komponen kerangka kerja yang didefinisikan berdasarkan RQ dan dikembangkan menjadi model implementasi beserta *template*, yaitu:

- *Common ground* dari *enterprise agile*
- Model konseptual, yang dilengkapi dengan berbagai praktik yang didapatkan berdasarkan studi pustaka dan beberapa kali proses wawancara
- Model implementasi
- Dokumen *template* dan panduan



Gambar 6.3 Kerangka Kerja Akhir

6.2.3 *Common Ground* dari *Enterprise Agile*

Common ground merupakan bagian paling dasar didalam membuat pemetaan dari semua praktik yang ada pada *scaling agile*. Berdasarkan beberapa uji coba pada studi kasus, kerangka kerja ini juga dapat diimplementasikan pada proyek *agile*, dan *scaling agile*. Implementasi *hybrid* juga dapat menggunakan kerangka kerja ini. Proses pembentukan *common ground* dilakukan pada saa tahapan desain di DSR sampai dengan evaluasi berulang kali. Bentuk dasar *common ground* yang dijelaskan secara model konseptual ditunjukkan pada Gambar 6.4.

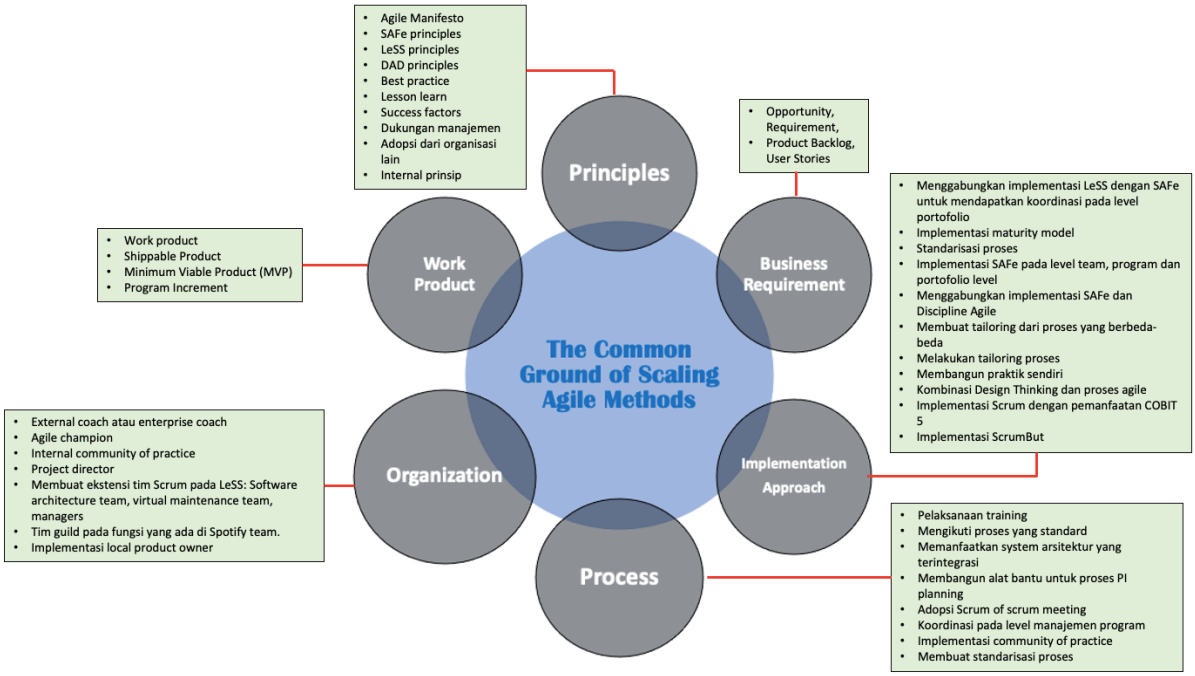
Gambar 6.4 *Common Ground* dari *Enterprise Agile*

Common ground juga dapat dilengkapi dengan beberapa praktik seperti ditunjukkan pada Gambar 6.5 yang didapatkan melalui proses SLR yang telah dijabarkan sebelumnya.

Pada komponen prinsip, terdapat beberapa praktik yang berhubungan, yaitu:

- *Agile Manifesto*
- *SAFe principles*
- *LeSS principles*
- *DAD principles*
- *Best practice*
- *Lesson learn*
- *Success factors*

- Dukungan manajemen
- Adopsi dari organisasi lain
- Prinsip internal



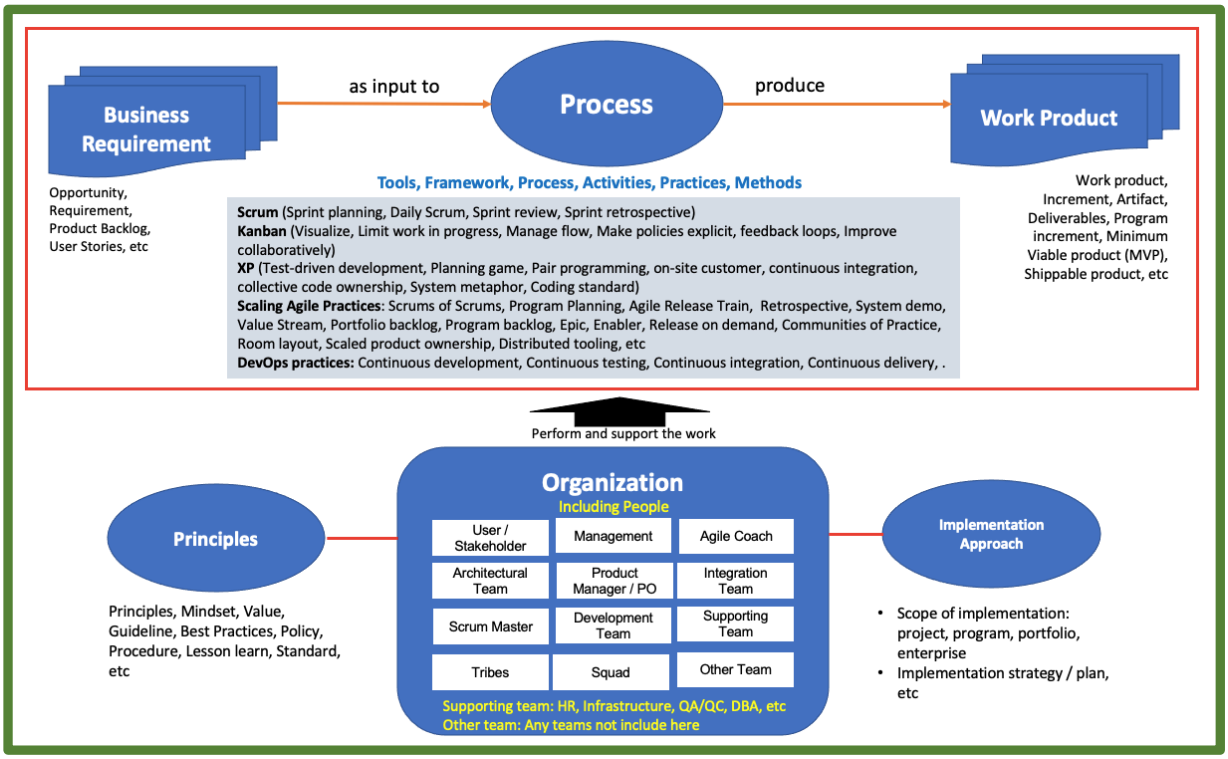
Gambar 6.5 Pemetaan Praktik Saat ini

6.2.4 Model Kerangka Kerja

Model konseptual kerangka kerja merupakan hubungan antara komponen *common ground*. Organisasi menggunakan prinsip sebagai panduan umum di dalam pelaksanaan pekerjaan mereka, sedangkan strategi implementasi merupakan panduan secara khusus di dalam melakukan arahan pelaksanaan. Secara proses, model kerangka kerja ini menggunakan aliran standar adanya input, proses, dan luaran. Input berupa business *requirement*, dan luaran beruapa *work product*. Model konseptual ini memang merupakan model yang umum dan organisasi dapat melakukan adopsi terhadap model ini, dan tidak harus menjalankan semua praktik yang dicantumkan. Beberapa catatan lain dari model ini adalah:

- Model dapat digunakan untuk proses iterasi. Proses iterasi ini dijelaskan melalui *implementation plan* yang merupakan bagian dari kerangka kerja

- Untuk implementasi pada level *enterprise*, maka dapat digunakan komponen *enterprise* lainnya yang merujuk kepada PMBOK



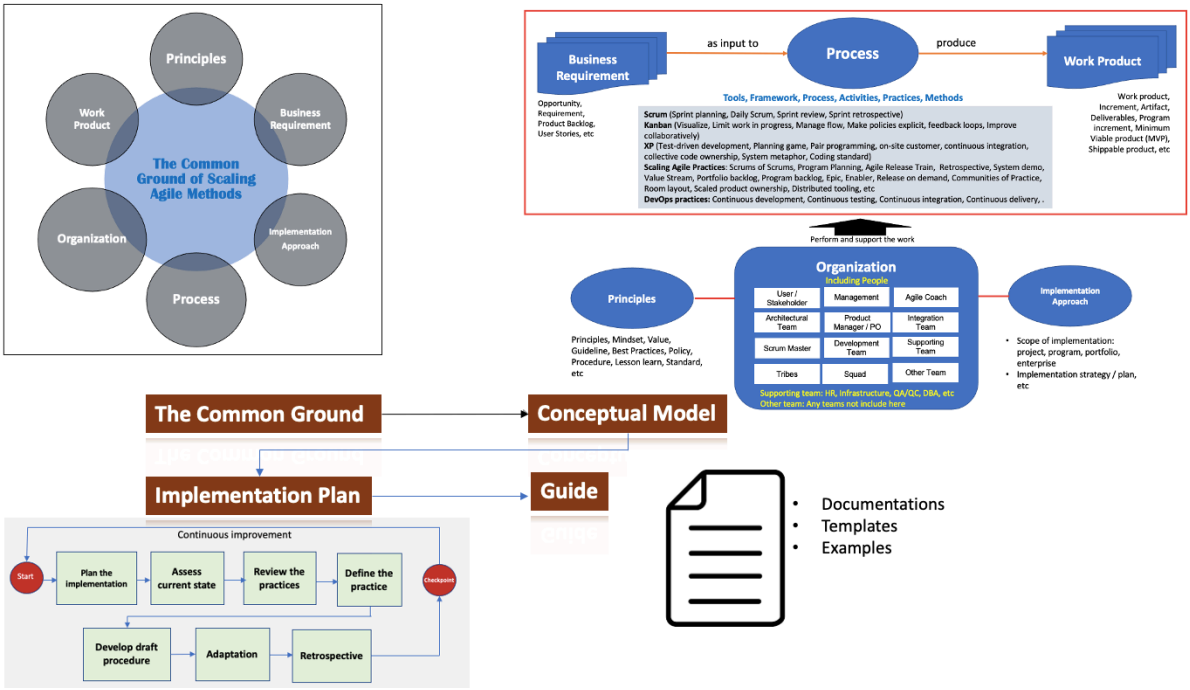
Gambar 6.6 Model Kerangka Kerja *Enterprise agile*

Gambar 6.6 menunjukkan model konseptual yang sudah diberikan daftar praktik yang dapat dipergunakan sebagai rujukan untuk digunakan pada organisasi. Praktik-praktik tersebut diambil dari proses studi pustaka dan evaluasi sebelumnya. Praktik yang sering digunakan adalah pada bagian proses dan organisasi. Pada bagian proses, terdapat daftar praktik dari Scrum, Kanban, XP, *scaling agile practice* dan DevOps. Tujuan dicantumkannya daftar praktik pada setiap komponen *common ground* dimaksudkan untuk memudahkan para praktisi untuk mempelajari secara lebih jauh praktik tersebut dan melakukan *assessement* apakah praktik tersebut dapat diimplementasikan pada proyek *agile* mereka.

6.2.5 Perbandingan Kerangka Kerja ini dengan Kerangka Kerja *Scaling Agile*

Kerangka kerja ini dibangun berdasarkan common ground dari *scaling agile* yang sudah ada sebelumnya, seperti SAFe, LeSS, DA, Scrum of Scrum, dan Nexus. *Common ground* ini sudah dijelaskan sebelumnya. Kerangka kerja ini dibentuk dari praktik-praktik yang ada pada kerangka kerja scaling agile tersebut. Pada subbab ini, penulis berusaha menjelaskan

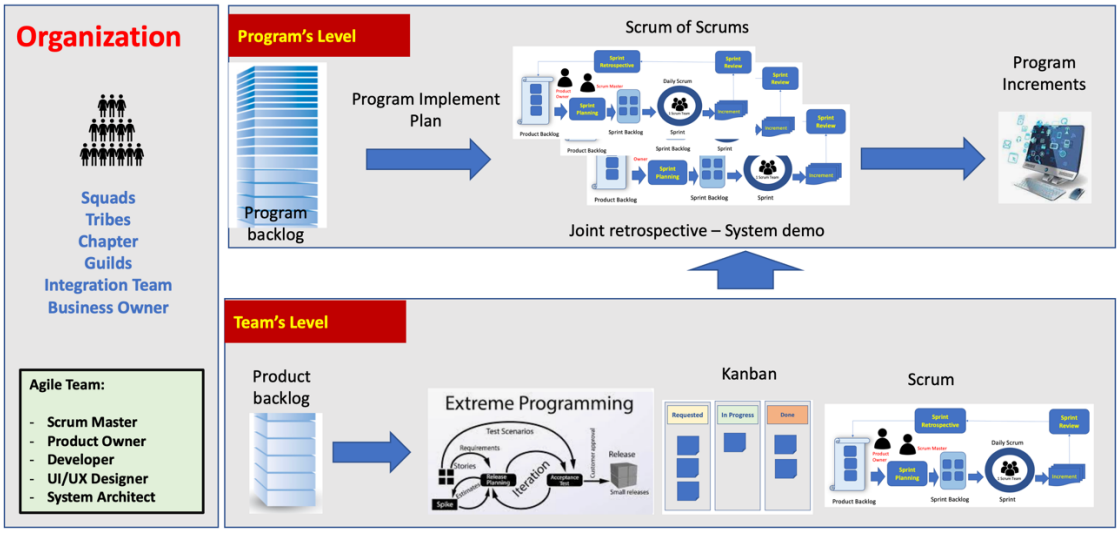
bagaimana contoh penerapan kerangka kerja ini menggunakan praktik-praktik yang ada pada beberapa komponen kerangka kerja scaling agile. Kerangka kerja pada penelitian ini ditunjukkan pada Gambar 6.7, yang terdiri dari *common ground*, model*, implementation plan*, dan dokumentasi.



Gambar 6.7 Kerangka Kerja *Enterprise Agile*

Dari kerangka kerja yang dihasilkan, organisasi dapat membangun metode pengembangan sendiri sesuai dengan kebutuhan yang ingin dicapai. Proses pembangunan metode bagi organisasi tersebut dijelaskan secara rinci pada subbab sebelumnya. Gambar 6.8 berikut merupakan contoh penggambaran kerangka kerja yang digunakan oleh organisasi menggunakan konsep kerangka kerja pada penelitian ini. Contoh tersebut menerapkan pemilihan praktik pada beberapa metode agile dan scaling agile yang sudah ada, yaitu:

- Scrum
- Kanban
- Extreme programming
- SAFe: Program implement planning, program increment, system demo
- LeSS: Joint retrospective
- Spotify agile: Squads, Tribes, Chapter, Guilds
- Scrum of Scrums



Gambar 6.8 Implementasi Kerangka Kerja

Pada contoh kerangka kerja pada penerapan organisasi tersebut dapat terlihat, adanya level tim dan level program. Level tim merupakan pengembangun produk yang dikerjakan untuk setiap tim, sedangkan level program merupakan integrasi dari level tim. Pada level tim, terdapat product backlog yang dikerjakan oleh agile team menggunakan praktik-praktik pada Extreme programming, Kanban, dan Scrum. Sedangkan pada level program, terdapat program backlog yang merupakan kumpulan dari product backlog. Pada level program ini, dilakukan pemilihan praktik beruapa program increment plan, yaitu rapat gabungan seluruh tim pada saat awal implementasi. Pada level implementasi, secara koordinasi digunakan praktik scrum of scrums dengan joint restrospective dan system demo. Pada akhir proyek yang ditentukan, maka dihasilkan sebuah program increment.

6.3 Analisis Kontribusi Penelitian

Pada subbab sebelumnya sudah dipaparkan mengenai kontribusi penelitian ini terhadap SWEBOK. Pada subbab ini penulis mencoba melakukan analisis hasil penelitian terhadap kontribusi yang telah dipaparkan sebelumnya. Kontribusi utama penelitian ini adalah menyediakan suatu kerangka kerja yang dapat dipergunakan oleh organisasi di dalam membangun cara kerja atau prosedur kerja *agile* mereka. Kerangka kerja ini menyediakan panduan yang lengkap yang menjawab pertanyaan *why, how* dan *what*. Dokumen panduan penggunaan kerangka kerja serta dokumen *template* yang dapat dipergunakan oleh organisasi sudah disediakan pada spesifikasi kerangka kerja ini.

6.4 Kontribusi pada Aspek Teori

Kerangka kerja ini berkontribusi secara teori di dalam pengembangan *general theory* yang lebih luas pada bidang *scaling agile*. Hal ini dapat dijelaskan karena kerangka kerja ini dikembangkan dari konsep *general theory of software engineering*. *Common ground* yang dihasilkan dapat digunakan sebagai referensi teori di dalam pengembangan proyek *agile*. Pada subbab 6.6. sudah dipaparkan bagaimana kerangka kerja ini diuji coba berdasarkan kriteria *software engineering* seperti pada penelitian sebelumnya. *Path* yang digunakana dalam pengembangan kerangka kerja ini menjadi *general theory* adalah berasal dari SEMAT / The Essence dan GTSE dikembangkan menjadi *common ground* dan *general theory.*

Penelitian ini juga berkontribusi di dalam implementasi *essence language* yang merupakan *kernel* yang dapat menjelaskan secara lebih rinci mengenai *common ground* pada *scaling agile*. Pembuatan *essence language* diturunkan dari pemetaan *kernel alpha* pada essece pada *scaling agile* yang merupakan hasil RQ4 dari penelitian ini.

6.5 Analisis Aspek Teori dan Praktik

Pada penelitian ini, aspek teori dan praktik sangat berkaitan dan seimbang. Hal ini didasari dengan rujukan teori yang sangat memadai. Teori-teori tersebut pada dasarnya juga paduan antara keseimbangan antara teori dan praktik, yaitu:

- *Theory of software engineering*
- *The essence of software engineering*
- *Design science research (DSR)*

Aspek teori pada penelitian ini dikaitkan dengan referensi *theory of software engineering* pada pembuatan *common ground* serta uji validasi yang menggunakan kriteria pada teori *software engineering*. Kontribusi penelitian secara teori adalah pengembangan kerangka kerja ini menjadi *general theory* yang dapat dikembangkan secara lebih mendalam. Aspek praktik adalah penggunaan kerangka kerja yang berguna bagi organiasi di dalam mengambangkan metode atau prosedur kerja mereka di dalam pelaksanaan proyek *agile* dan *scaling agile*.

Pada tahapan penelitian, terlihat keseimbangan antara aspek teori dan praktik. Penggunaan DSR sebagai metode penelitian sendiri menyeimbangan antara kedua aspek ini. Pembuatan

kerangka kerja ini lebih banyak ditekan pada metode penelitian yang bersifat teoritis, seperti penggunaan *general theory of software engineering* sebagai salah satu dasar pembuatan kerangka kerja. Pada subbb 6.5 dan 6.6 sudah dijelaskan secara rinci mengenai analisis pengelitian ini berdasarkan referensi *general theory of software engineering* dan evaluasi hasil penelitian terhadap teori *software engineering*. Aspek praktik sangat terlihat jelas pada saat melakukan uji coba yang melibatkan studi kasus pada banyak organisasi di berbagai industri, seperti *e-commerce,* pemerintahan, perbankan, dan perusahaan rintisan atau *startup*.

6.6 Analisis Uji Coba

Uji coba dilakukan pada para praktisi yang sedang menjalankan proyek *agile*. Lingkup uji coba tergantung dari jenis perusahaan:

1. Perusahaan *startup*
2. Perusahaan menengah dan *enterprise*

Ruang lingkup uji coba saat ini masih mengarah pada level proyek pada satu department yang terdiri dari team *agile* dan *scaling agile*. Saat awal, para praktisi akan melakukan *assesmen* terlebih dahulu, apakah sudah terdapat SOP atau belum. Secara umum, terdapat keseragaman implementasi berdasarkan *common ground*, dengan menggunakan template yang disediakan, tetapi pada pengembangannya, setiap perusahaan akan mengadopsi praktik yang berbeda disesuikan dengan kondisi perusahaan.

Uji coba lainnya juga dilakukan pada perusahaan rintisan. Perusahaan tersebut memiliki kapasitas pegawai di bawah seratus orang. Keseragaman uji coba umumnya sama dengan perusahaan menengah seperti perusahaan daring, perusahaan asuransi nasional dan perusahaan manafaktur farmasi di Indonesia. Perbedaannya pada level pemilihan praktik yang cenderung pada level proyek. Sedangkan uji coba pada perusahaan yang lebih besar, maka pemilihan praktik cenderung menggunakan metode *scaling agile* karena level implementasi yang lebih besar, misalnya pada level departemen yang terdiri dari beberapa tim *agile*.

Uji coba kerangka kerja ini juga pernah dilakukan pada perusahaan yang sudah memiliki prosedur *agile* yang sudah cukup matang, dan relatif tidak memiliki masalah yang dihadapi pada level pengerjaan proyek, seperti:

- Proyek aplikasi pengembangan *mobile banking*
- Pengembangan aplikasi pada perusahaan *automotive* di Indonesia

Pengembangan metode pada kedua perusahaan tersebut tidak begitu menghasilkan kontribusi yang berarti, tetapi penggunaan essence *language* dapat digunakan untuk pengembangan lebih terperinci. Hal ini pernah dilakukan penjabaran essence *language* untuk studi kasus proyek Jenius secara *proof of concepts.*

Analisis uji coba juga pernah dilakukan untuk studi kasus organisasi yang menerapkan *hybrid*, yaitu perpaduan antara pengembangan *agile* dan *waterfall*. Uji coba ini diterapkan pada perusahaan asuransi terkemuka di Indonesia. Referensi yang digunakan adalah mengenai konsep pemilihan praktik dan prosedur di dalam pembuatan praktik. Untuk studi kasus ini, kerangka kerja ini masih tetap dapat dipakai dengan menggunakan referensi tambahan untuk *waterfall* yang dapat diambil dari salah satunya perpaduan antara PMBOK dan *agile practice guide.*

6.7 Keseragaman Pola Uji Coba

Berdasarkan uji coba yang dilakukan pada berbagi studi kasus, penulis melihat adanya keseragam pola serta karakteristik pada setiap sudi kasus. Keseragaman uji coba tentunya sesuai dengan panduan yang didapat pada kerangka kerja ini melalui pemilihan praktik dan proses implementasinya. Perbedaan yang terlihat pada pola uji coba terdapat pada jenis studi kasus serta kebutuhan pada setiap organisasi, misalnya:

- Untuk perusahaan *startup* dan perusahaan kecil, maka pemilihan praktik cukup mencakup pada metode *agile* pada level proyek, misalnya Scrum, Kanban, dan Extreme Programming
- Pada perusahaan besar, misalnya studi kasus pada 2 perusahaan perbankan di Indonesia, maka pemilihan praktik mencakup *agile* sampai level *scaling agile* karena pada studi kasus tersebut memang dibutuhkan metode *agile* yang lebih kompleks. Pemilihan praktik bisa didapatkan dari beberapa praktik dari SAFe, LeSS, Nexus, dan Spotify Agile.

6.8 Masukan dan Luaran dari Hasil Uji Coba

Penulis mencoba melakukan analisis mengenai luaran hasil yang dilakukan setelah proses uji coba dilakukan. Gambar 6.9 menjelaskan ringkasan hasil uji coba yang diterangkan pada subbab sebelumnya. Secara ringkas, uji coba ini memiliki beberapa perspektif:

- Jenis perusahaan
- Kesiapan organisasi
- *Size* perusahaan
- Pola keseragaman
- Metode saat ini yang dipakai, yang akan dibahas pada subbab ini

Gambar 6.8 Ringkasan Hasil Uji Coba

Berdasarkan perspektif metode saat ini yang digunakan oleh organisasi, maka didapat beberapa analisis, yaitu:

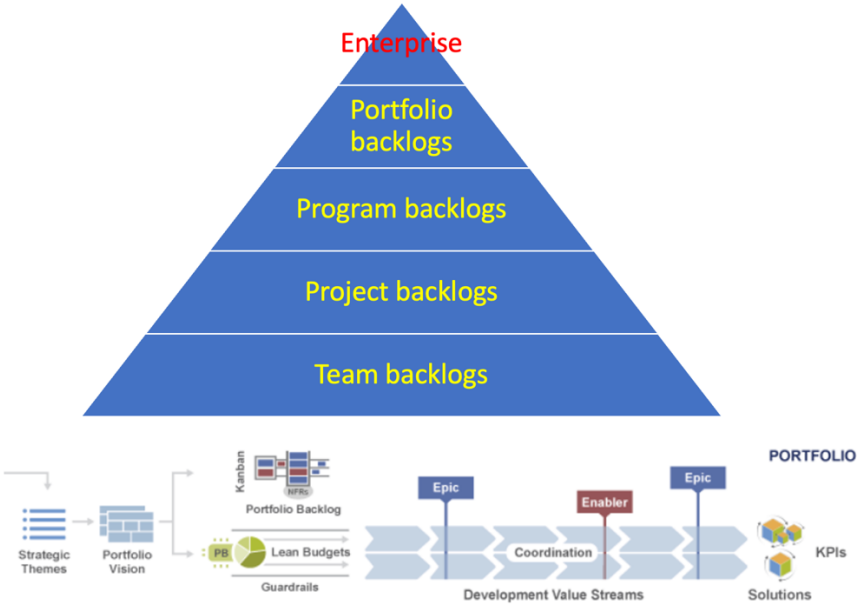
- Studi kasus yang belum menerapkan *agile*, atau dengan kata lain masih melakukan proses secara *waterfall*. Organisasi ingin melakukan transformasi ke proses *agile* dengan pertimbangan kebutuhan proyek. Dengan menerapkan kerangka kerja ini, luaran yang dihasilkan tentunya memang suatu metode pengembangan yang berbasis *agile.* Ini terjadi pada studi kasus Kalbe Farma yang menerapkan Kanban dan Scrum sebagai metode utamanya.
- Studi kasus yang sudah menerapkan agile di dalam proses pengembangannya, tetapi praktik-praktik tersebut tidak dijalankan secara benar, konsisten, dan terdokumentasi. Kerangka kerja ini dan implementasi pada studi kasus dapat dijadikan rujukan bagaimana membangun metode pengembangan yang sesuai. Hasil akhir dari metode yang didapatkan tentunya berupa metode agile. Contoh studi kasus pada poin ini adalah implementasi pembuatan metode pengembangan agile pada detikcom
- Studi kasus yang sudah menerapkan agile tetapi masih dibutuhkan beberapa penyempurnaan di dalam proses pengembangan metodenya, maka luaran dari proses

ini adalah metode pengembangan berbasis *agile*. Hal ini terjadi pada studi kasus, misalnya Bank BRI, perusahaan konsultan internasional, dan pada perusahaan *software house*

- Studi kasus yang menerapkan proses *hybrid* yaitu penggabungan antara praktik *agile* dan *waterfall*. Kerangka kerja ini dapat digunakan dengan menambahkan praktik-praktik pada proses *waterfall*, misalnya yang merujuk pada PMBOK dan PRINCE2. Hasil luaran metode adalah metode *hybrid*. Kondisi ini terjadi pada studi kasus perusahaan asuransi nasional di Indonesia
- Pada penerapan studi kasus, tidak ditemukan hasil luaran dari proses yang awalnya dijalankan secara *agile* lalu berubah hasilnya menjadi *waterfall* setelah penerapan kerangka kerja ini, walaupun secara teoritis ini dimungkinankan jika merujuk pada panduan implementasi di mana studi kasus harusnya memang sesuai menggunakan metodologi waterfall. Justifikasi ini bisa merujuk pada landasan pemilihan metode yang dijelaskan pada sebelumnya.

6.9Analisis Kecocokan Uji Coba pada *Requirement Enterprise*

Pengertian *enterprise* seperti dijelaskan pada subbab sebelumnya yang merujuk pada (D.Leffingwell, 2020) pada level *enterprise*, terdapat *portfolio backlogs* yang merupakan penjabaran dari *strategic theme.* Hal ini dijabarkan pada Gambar 6.10 di bawah ini. Dibutuhkan *scaling agile* karena *agile framework* pada level tim (Scrum, Kanban, XP, dll) hanya cocok untuk level tim dan *project backlogs*. Untuk level di atasnya dibutuhkan *scaling agile framework*.



Gambar 6.10 Model Enterprise Agile

Pada uji coba yang melibatkan lebih dari satu tim agile, maka dapat menggunakan praktik-praktik pada *enterprise agile* yang merujuk pada *scaling agile*. Contoh-contoh praktik-praktik tersebut dapat diambil, misalnya adopsi dari SAFe, seperti:

- *Portfolio backlog*
- *Epic*
- *Program backlog*
- *Agile release traing*
- *Epic owner*
- *Enterprise architect*
- Dan lainnya

BAB 7
KESIMPULAN DAN SARAN

Bab ini membahas kesimpulan dan saran yang diperoleh melalui hasil penelitian, analisis, demonstrasi atau uji coba kerangka kerja berdasarkan metodologi *design science research* (DSR). Kesimpulan ini akan menjawab pertanyaan penelitian yang dipaparkan pada bab 1, sedangkan saran merupakan tindak lanjut yang dapat dilakukan untuk penelitian selanjutnya. Kontribusi dan keterbatan penelitian juga disampaikan pada akhir pembahasan. Beberapa bahasan yang juga merupakan ringkasan dari penelitian juga dipaparkan yaitu kontribusi penelitian, keterbatasan penelitian dan komunikasi hasil penelitian.

7.1 Kesimpulan

Kesimpulan penelitian ini untuk menjawab pertanyaan penelitian yang sebelumnya dibahas yaitu:

1. Bagaimana *common ground* untuk *enterprise agile* pada organisasi berdasarkan pemikiran *the essence of software engineering* dan *general theory of software engineering*?
2. Bagaimana model konseptual untuk *enterprise agile* yang dikembangkan dari komponen *common ground*?
3. Bagaimana *current practice*s berdasarkan *systematic literature review (SLR)* yang sesuai menurut *common ground*?
4. Bagaimana pemetaan *The Essence SEMAT kernel* pada *common ground* dari kerangka kerja *enterprise agile*?
5. Bagaimana susunan akhir kerangka kerja dari *enterprise agile?*

7.1.1 RQ1: *Common Ground* dari *Enterprise Agile*

Common ground dari *enterprise agile* dibahas pada subbab 4.1 dan ditunjukkan pada Gambar 7.1 yaitu:

- Prinsip
- Organisasi
- *Business requirement*

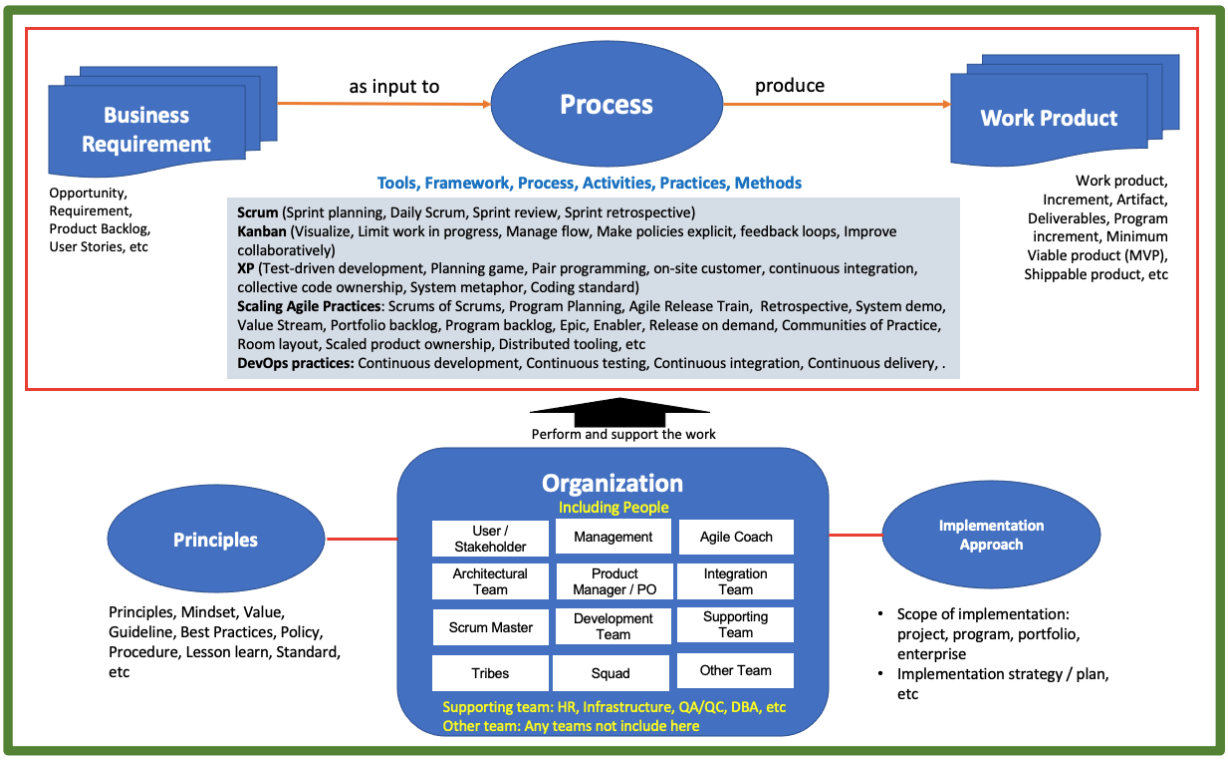
- Proses
- *Work product*
- Strategi implementasi

Common ground ini dihasilkan melalui beberapa tahapan iterasi dan evalusi melalui proses pengujian pakar, wawancara dan FGD. Beberapa uji coba yang dilakukan pada bab 5 telah berhasil dilakukan pemetaan pada praktik-praktik pada organisasi pada komponen *common ground* ini.

Gambar 7.1 *Common Ground* dari *Enterprise agile Framework*

7.1.2 RQ2: Model konseptual dari *enterprise agile*

Model konseptual secara garis besar dibahas pada subbab 4.3 dan dijelaskan pada Gambar 7.2. Model konseptual ini merupakan hubungan antara *common ground* pada yang didefinisikan pada RQ1. Model konseptual ini merupakan model tahap akhir yang dilakukan penyempurnaan melalui tahapan-tahapan evaluasi dan uji coba yang dilakukan. Uji coba pada organisasi, yang dijelaskan pada bab 5 berhasil menjadi model konseptual ini sebagai panduan bagi organisasi di dalam mendefinisikan proses atau prosedur mereka.



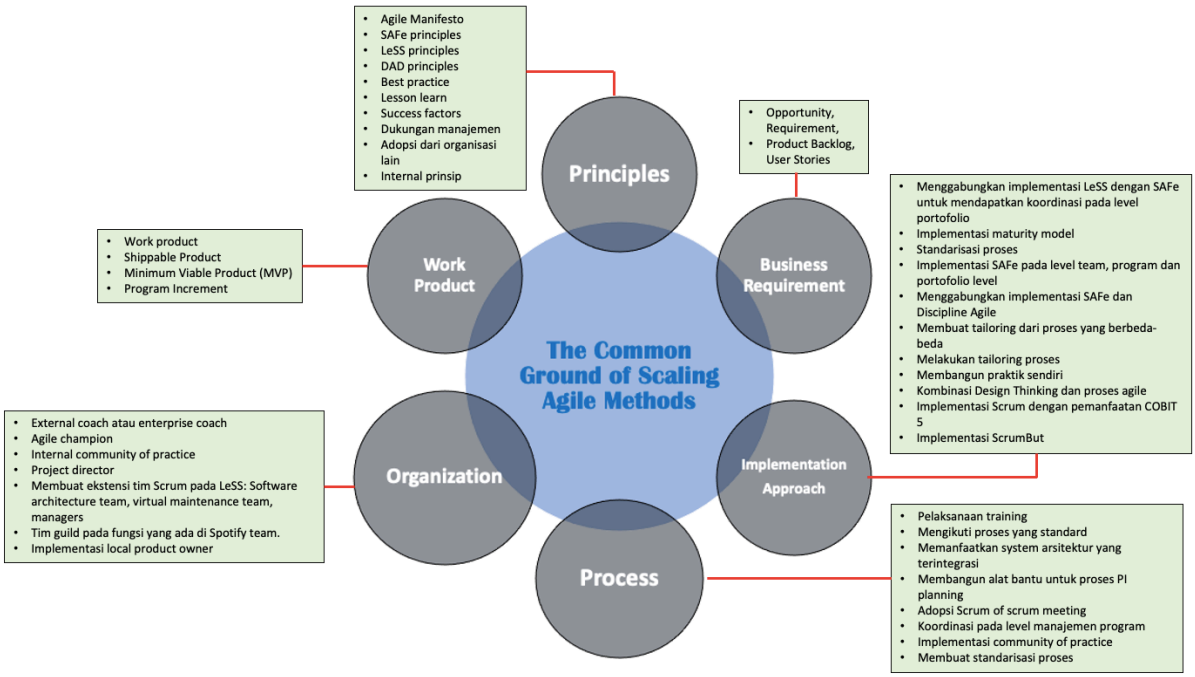
Gambar 7.2 Model Konseptual *Enterprise agile Framework*

Model konseptual ini seperti dipaparkan pada pertanyaan penelitian diadopsi dari *general theory of software engineering* (Johnson & Ekstedt, 2016). Hubungan antara komponen pada model konseptual tersebut dijelaskan sebagai berikut:

- Prinsip memandu organisasi di dalam pelaksanaan implementasi proyek *agile*
- Organisasi terdiri dari struktur tim dan pemangku kepentingan yang terlibat di dalam proyek
- Pendekatan implementasi merupakan strategi dan perencanaan organisasi
- Tim dan orang-orang yang terlibat pada organisasi akan menjalankan proses berdasarakan masukan dari *business requirement*, dan akan menghasilkan *work product* sebagai luaran dari implementasi

7.1.3 RQ3: *Current Practices* untuk *Enterprise Agile*

Current practices atau praktik saat ini untuk *enterprise agile* dilakukan melalui proses *systematic literature review* (SLR). Penjelasan terperinci mengenai pelaksanaan SLR ini dijelaskan pada subbab 4.3. Hasil dari SLR dipetakan menurut komponen common gound seperti dijelaskan pada Gambar 7.3. Pada setiap lingkaran *common ground*, penulis memberikan keterangan mengenai praktik yang dapat dilakukan.

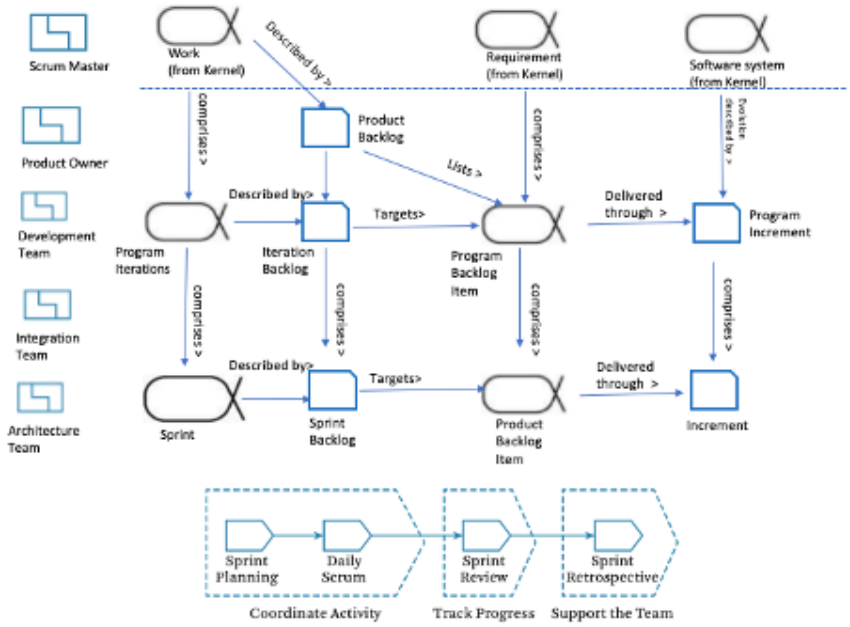


Gambar 7.3 Praktik Saat ini Berdasarkan SLR

Praktik saat ini yang didapatkan melalui proses SLR digunakan sebagai salah satu komponen kerangka kerja yang dapat digunakan sebagai referensi organisasi di dalam mendefinikan praktik yang dapat mereka lakukan. Penjelasan lebih rinci mengenai praktik saat ini yang diperoleh dari literatur, dijelaskan pada subbab 6.4.

7.1.4 Pemetaan *The Essence SEMAT Kernel* untuk *Enterprise Agile Framework* (RQ4)

Common ground yang dibantuk pada penelitian ini, dapat dilakukan pemetaan pada komponen essence *language*. Penjelasan lebih terperinci mengenai pemetaan ini, dapat dilihat pada subbab 4.7. Berdasarkan panduan dari *The Essence*, kami melakukan proses pemetaan dari setiap elemen di *kernel alpha* ke elemen model (yaitu, *requirement*, organisasi, prinsip, kerangka kerja, dan proses, dan produk kerja).



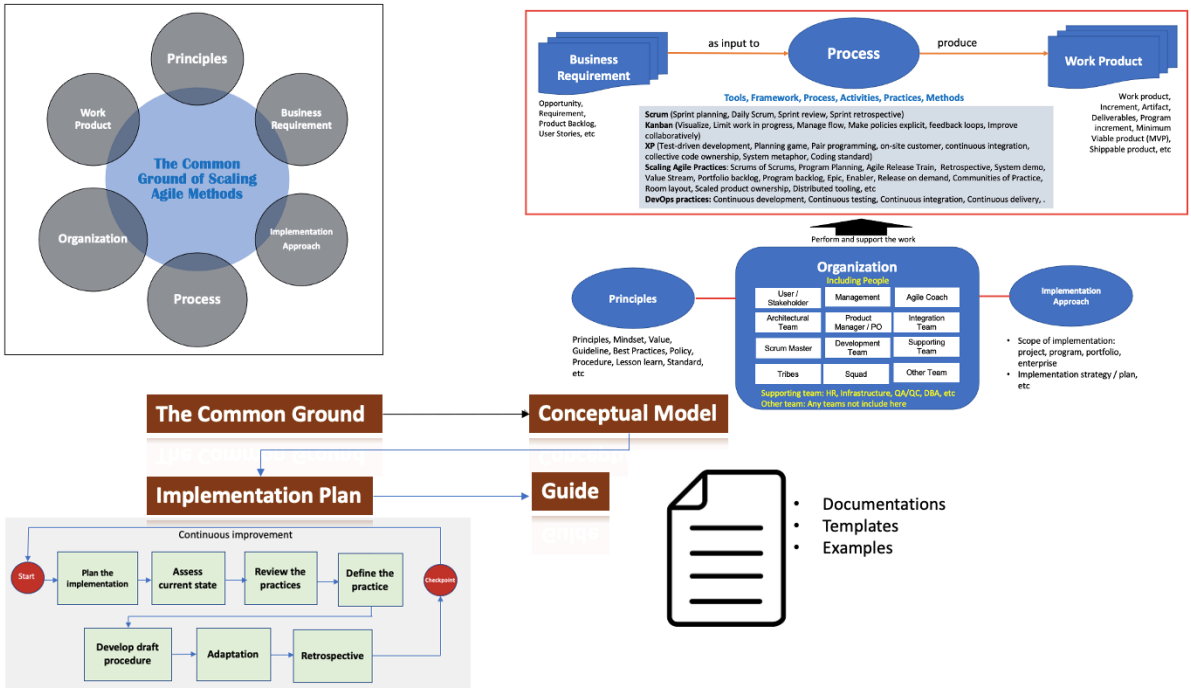
Gambar 7.4 Contoh Implementasi *Scaling Scrum* pada *Essence Language*

Gambar 7.4 menunjukkan hasil pemetaan *common ground* pada Scrum. Setiap organisasi dapat mengembangkan *essence language* yang berbeda-beda berdasarkan prosedur yang mereka buat. Beberapa studi kasus memberikan contoh implementasi penggunaan *the essence language* berdasarkan kondisi prosedur yang mereka hasilkan. Penelitian ini memberikan petunjuk berupa pemetaan dan contoh untuk *scaling Scrum* untuk implementasi *the essence language*.

7.1.5 Kerangka Kerja *Enterprise agile* (RQ5)

Kerangka kerja akhir ditunjukkan pada Gambar 7.5. Penjelasan lebih mendalam dijelaskan pada subbab 6.4. Kerangka kerja akhir ini didapatkan melalui proses dan tahapan DSR dengan tahapan iterasi untuk evaluasi yang berulang-ulang. Ini merupakan gabungan dari komponen kerangka kerja yang didefinisikan berdasarkan RQ dan dikembangkan menjadi model implementasi beserta *template*, yaitu:

- *Common ground* dari *enterprise agile*
- Model konseptual, yang dilengkapi dengan berbagai praktik yang didapatkan berdasarkan studi pustaka dan beberapa kali proses wawancara
- Model implementasi
- Dokumen *template* dan panduan



Gambar 7.5 Kerangka Kerja Akhir

7.2 Saran-saran

Subbab ini memaparkan saran-saran untuk penelitian selanjutnya berdasarkan hasil analisis dan tahapan uji coba yang dilakukan sebelumnya.

7.2.1 Penelitian Studi Kasus

Penelitian studi kasus dengan melakukan pengembangan metodologi dapat dilakukan dengan menggunakan kerangka kerja ini. Demonstrasi atau uji coba kerangka kerja yang dibahas pada subbab 5 dapat dijadikan acuan dan panduan di dalam pembangun metodologi. Contoh pembangunan metodologi yang sudah dilakukan dapat mengacu pada https://bit.ly/Scaling*agile*Framework.

Salah satu penelitian studi kasus yang sudah dilakukan publikasi adalah penerapan kerangka kerja ini pada salah satu perusahaan farmasi di Indonesia (A. N. Fitriani, Raharjo, Hardian, & Prasetyo, 2021). Penelitian ini menggunakan pendekatan yang sesuai dengan panduan kerangka kerja, yaitu penggunaan the essence *requirement* (Jacobson et al., 2012) dan design science research (Peffers et al., 2007) di dalam pembentukan prosedur kerja menggunakan Lean dan Kanban.

7.2.2 Membangun *Software Engineering Theory*

Penelitian ini dapat dilakukan dasar sebagai pengembangan teori rekayasa perangkat lunak. Pembahasan lebih terperinci dapat dilihat pada subbab 6.9 mengenai rencana pengembangan penelitian menjadi teori rekayasa perangkat lunak. Model konseptual yang telah disusun dan dilakukan uji coba dapat menjadi acuan yang layak untuk dapat dikembangkan menjadi sebuah teori karena model konseptual yang dibangun merujuk kepada *general theory of software engineering* (Johnson & Ekstedt, 2016). Kerangka kerja ini sendiri sudah dilakukan validasi berdasarkan konsep theory of software engineering seperti dijelaskan pada subbab 6.6.

7.2.3 Pengembangan Obyek Penelitian Lainnya

Kerangka kerja ini dibangun berdasarkan konsep *the Essense of software engineering*. Penelitian selanjutkan juga dapat membuat penelitian berdasarkan konsep yang dilakukan pada penelitian ini yaitu:

- Acuan penelitian berdasarkan *konsep the Essence of software engineering* (Jacobson et al., 2012)
- Penggunaan metodologi DSR untuk membangun artefak (Peffers et al., 2007)
- Kontribusi penelitian berdasarkan DSR
- Ekstensi *kernel* untuk *The Essence Language*

Berdasarkan spesifikasi the Essence, obyek penelitian lain dapat dibangun, seperti:

- *Product line engineering*
- Pengembangan metodologi DevOps, atau DevSecOps

7.2.4 Penerapan *the Essence Language*

Penelitian studi kasus dapat dilakukan dengan melakukan implementasi praktik *scaling agile* menggunakan *Essence Language*. Untuk *Essence Alpha* yang standar dapat mengacu kepada kartu dan *state* yang ada, sedangkan untuk pengembangan *Kernel Alpha* yang lain, dapat melakukan ekstensi dari kartu-kartu tersebut. Hasil dari RQ4 sebagai contoh penggunaan essence *language* pada *scaling* Scrum dapat dilihat pada subbab 4.7 dapat digunakan sebagai referensi penggunaan the essence *language*. Beberapa studi kasus pada penelitian ini, misalnya pada salah satu perusahaan *digital banking*, dapat dilihat pada subbab 5.7 berusaha membuat *proof of concept* untuk penerapan the *essence language.*

7.2.5 Pengembangan Penelitian ini Menjadi Teori *Software Engineering*

Penelitian ini menghasil suatu kerangka kerja *agile* dan *scaling agile* yang sudah dilakukan uji coba pada bebeberapa studi kasus. Model konseptual dapat dikembangkan menjadi suatu teori software engineering dengan alasan berikut:

- Kerangka kerja yang dibuat berdasarkan solid teori yang ada, yaitu: General Theory of software engineering dan The Essence of Software Engineering
- Metode pengembangan menggunakan DSR yang sesuai untuk pengembangan suatu teori
- Evaluasi kerangka kerja sudah dilakukan berdasarkan kriteria kualitas yang dipakai oleh teori software engineering (Johnson & Ekstedt, 2016)
- Kerangka kerja ini dapat menjawab beberapa fenomena di yang terjadi di dalam proses pengambangan perangkat lunak, seperti kerangka kerja ini dapat memberikan panduan pemetaan Essence Language untuk *scaling agile*

7.3 Kontribusi Penelitian

Berdasarkan paparan pada bab 1 mengenai kontribusi penelitian, maka dapat dijabarkan:
Kontribusi penelitian terhadap bidang akademik:

- Model konseptual kerangka kerja *agile* dapat memberikan referensi bagi akademisi di dalam menetapkan model implementasi proyek *agile*
- Model konseptual kerangka kerja ini dapat dikembangkan dan diformalkan menjaadi teori software engineering, sesuai dengan paparan pada subbab 6.9.

Kontribusi penelitian terhadap bidang industri:

- Kerangka kerja dapat digunakan sebagai panduan dan solusi di dalam pembangunan prosedur pengembangan proyek *agile*. Hal ini sudah dibuktikan dari uji coba dan demonstasi pada beberapa studi kasus. Subbab 5.4 sampai 5.16 menjelaskan implementasi dari uji coba ini.

7.4 Keterbatasan Penelitian

Sebagai sebuah kerangka kerja, tentunya masih banyak keterbatasan dibandingkan dengan kerangka kerja lain yang sudah terbukti, seperti *Scrum Guide, agileSHIFT, The Essence*

Framework, SAFe, dan lainnya. Spesifikasi yang dihasilkan belum memberikan panduan secara menyeluruh. Untuk impelementasi kerangka kerja ini, peneliti masih perlu memberikan panduan, walau pun secara ringkas kepada para praktisi yang akan menggunakannya. Beberapa uji coba pada banyak perusahaan belum mencakup satu tahapan implementasi yang utuh, walau pun secara garis besar, kerangka kerja ini sudah dapat digunakan.

Untuk mendapatkan taraf penyempurnaan, tentunya dibutuhkan waktu yang lebih lama, sosialisasi dan uji coba yang lebih mendalam serta beberapa kali penelitian lanjutan. Penulis berharap hasil karya ini dapat terus berkembang. Saat ini, sudah ada beberapa usulan proposal mengenai penelitian studi kasus menggunakan panduan kerangka kerja ini.

7.5 Komunikasi Hasil Penelitian

Komunikasi hasil penelitian merupakan salah satu tahapan akhir dari DSR (Peffers et al., 2007). Komunikasi pada penelitian ini adalah publikasi hasil penelitian pada jurnal internasional. Rencana lain dari komunikasi penelitian ini adalah pada penerbitan buku teks dan juga publikasi pada laman penelitian. Dokumentasi penelitian ini yang dijelaskan secara ringkas pada Lampiran 2 akan diterbitkan pada laman cs.ui.ac.id untuk konsumsi masyarakat umum. Dokumentasi kerangka kerja, langkah-langkah penerapannya, serta dokumen *template* akan dibuat semudah dan sederhana mungkin untuk memudahkan masyarakat menggunakan kerangka kerja ini untuk menyusun metode pengembangan proyek *agile* mereka.

DAFTAR PUSTAKA

Adolph, S., & Kruchten, P. (2013). Generating a useful theory of software engineering. *2013 2nd SEMAT Workshop on a General Theory of Software Engineering, GTSE 2013 - Proceedings*, 47–50. https://doi.org/10.1109/GTSE.2013.6613870

Agilemanifesto.org. (2001). *Manifesto for agile Software Development*. Retrieved from https://*agile*manifesto.org/

Almeida, R., Teixeira, J. M., Mira da Silva, M., & Faroleiro, P. (2019). A conceptual model for *enterprise* risk management. *Journal of Enterprise Information Management*, *32*(5), 843–868. https://doi.org/10.1108/JEIM-05-2018-0097

Alqudah, M., & Razali, R. (2016). A review of *scaling agile* methods in large software development. *International Journal on Advanced Science, Engineering and Information Technology*, *6*(6), 828–837. https://doi.org/10.18517/ijaseit.6.6.1374

Amorim, A. C., Mira da Silva, M., Pereira, R., & Gonçalves, M. (2020). Using *agile* methodologies for adopting COBIT. *Information Systems*, (xxxx), 101496. https://doi.org/10.1016/j.is.2020.101496

Asad, F., & Pinnington, A. H. (2014). ScienceDirect Exploring the value of project management : Linking Project Management Performance and Project Success. *JPMA*, *32*(2), 202–217. https://doi.org/10.1016/j.ijproman.2013.05.012

Axelos. (2018a). *A Guide to agileSHIFT*. TSO (The Stationery Office), part of Williams Lea Tag.

Axelos. (2018b). *Prince2 agile*. United Kingdom for The Stationery Office: The Stationery Office Ltd. Kindle Edition. Retrieved from axelos.com

AXELOS. (2017). *Managing Successful Projects with PRINCE2* (6th ed.). London: The Stationery Office.

Bass, J. M. (2014). Scrum master activities: Process tailoring in large *enterprise* projects. *Proceedings - 2014 IEEE 9th International Conference on Global Software Engineering, ICGSE 2014*, 6–15. https://doi.org/10.1109/ICGSE.2014.24

Bass, J. M. (2016). Artefacts and *agile* method tailoring in large-scale offshore software development programmes. *Information and Software Technology*, *75*, 1–16. https://doi.org/10.1016/j.infsof.2016.03.001

Beecham, S., Clear, T., Lal, R., & Noll, J. (2021). Do *scaling agile requirement*s address global software development risks? An empirical study. *Journal of Systems and Software*, *171*,

110823. https://doi.org/10.1016/j.jss.2020.110823

Bittner, K., Kong, P., & West, D. (2017). *Nexus Framework for Scaling Scrum*. Addison-Wesley Professional.

Bjarnason, E., Smolander, K., Engström, E., & Runeson, P. (2016). A theory of distances in software engineering. *Information and Software Technology*, *70*, 204–219. https://doi.org/10.1016/j.infsof.2015.05.004

Bourque, P., & Fairley, R. (2014). *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)*.

Brenner, R., & Wunder, S. (2015). Presentation and Real World Example. *2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 1–2.

Brown, A. W., Ambler, S., & Royce, W. (2013). Agility at scale: Economic governance, measured improvement, and disciplined delivery. *Proceedings - International Conference on Software Engineering*, 873–881. https://doi.org/10.1109/ICSE.2013.6606636

Buzan. (2005). *Mind Map Handbook The Ultimated Thinking Tool*. Australia: HarperCollins Publishers Pty. Ltd.

Canty, D. (2015). *agile for Project Managers*. (G. Levin, Ed.). Boca Raton New York: CRC Press Taylor & Francis Group. Retrieved from www.crcpress.com

Chan, F. K. Y., & Thong, J. Y. L. (2009). Acceptance of *agile* methodologies : A critical review and conceptual *requirement*. *Decision Support Systems*, *46*(4), 803–814. https://doi.org/10.1016/j.dss.2008.11.009

Chikhale, M. M., & Mansouri, M. (2015). An *agile* and Collaborative Framework for Effective Governance to Enhance Management in Large-Scale Enterprise Business Systems: The Case of Apple Inc. *Global Journal of Flexible Systems Management*, *16*(3), 283–293. https://doi.org/10.1007/s40171-015-0098-9

Chow, T., & Cao, D. (2008). A survey study of critical success factors in *agile* software projects, *81*, 961–971. https://doi.org/10.1016/j.jss.2007.08.020

Clarke, P., O'Connor, R.V., "The situational factors that affect the software development process: towards a comprehensive reference framework," Journal of Information and Software Technology, Vol. 54, Issue 5, Elsevier, pp. 433-447, 2012.

Conboy, K., & Carroll, N. (2019). Implementing Large-Scale *agile* Frameworks: Challenges and Recommendations. *IEEE Software*, *36*(2), 44–50. https://doi.org/10.1109/MS.2018.2884865

Cooper, D., & Schindler, P. (2014). *Business research methods*.

Council, T. G. A. (2015). *GridWise Transactive Energy Framework Version 1.*

Creswell, J. W. (2013). Research Design_ Qualitative, Quantitative, and Mixed Method Approaches. SAGE Publications.

Crowder, J. A., & Friess, S. (2015). *agile Project Management : Managing for Success*.

D.I. Sjøberg, T. Dybå, B.C. Anda, J. E. H. (2008). *Building theories in software engi- neering, in: Guide to Advanced Empirical Software Engineering*. Springer International Publishing.

Darrell K. Rigby, Sutherland, J., & Noble, A. (2018). *agile* at Scale. *Harard Business Review*. Retrieved from https://hbr.org/2018/05/*agile*-at-scale

Davis, B. (2013). *agile Practices for Waterfall Projects Shifting Processes for Competitive Advantage*. J.Ross Publishing.

Deloitte. (2019). Agility at scale. The *agile* way of working has arrived in banking, (September). Retrieved from https://www2.deloitte.com/content/dam/Deloitte/lu/Documents/technology/lu-*agile*-banking.pdf

Deloitte. (2020). *Combating COVID-19 with an agile change management approach*.

Denning, S. (2018). *The Age of agile. How Smart Companies Are Transforming they Way Works Gets Done*. AMACOM. ISBN: 9780814439104.

Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale *agile* transformations: A systematic literature review. *Journal of Systems and Software*, *119*, 87–108. https://doi.org/10.1016/j.jss.2016.06.013

Dingsøyr, T., Dybå, T., Gjertsen, M., Jacobsen, A. O., Mathisen, T. E., Nordfjord, J. O., … Strand, K. (2019). Key lessons from tailoring *agile* methods for large-scale software development. *IT Professional*, *Year: 2019*, 34–41.

Dingsøyr, T., Rolland, K., Moe, N. B., & Seim, E. A. (2017). Coordination in multi-team programmes: An investigation of the group mode in large-scale *agile* software development. *Procedia Computer Science*, *121*, 123–128. https://doi.org/10.1016/j.procs.2017.11.017

Dittrich, Y. (2016). What does it mean to use a method? Towards a practice theory for software engineering. *Information and Software Technology*, *70*, 220–231. https://doi.org/10.1016/j.infsof.2015.07.001

Dow. (2012). *The Tactical Guide for Building a PMO Take Your PMO Into the Future*. United State of America: William Dow Publishing.

DSDM. (2014). *The DSDM agile Project Framework Handbook*.

Durisic, D., & Berenyi, A. (2019). *agile* System Architecture in Large Organizations: An Experience Report from Volvo Cars. *Proceedings - 2019 IEEE International Conference on Software Architecture - Companion, ICSA-C 2019*, 33–36. https://doi.org/10.1109/ICSA-C.2019.00014

Dyba, T., & Dingsoyr, T. (2015). *agile* Project Management: From Self-Managing Teams to Large-Scale Development. *Proceedings - International Conference on Software Engineering*, *2*, 945–946. https://doi.org/10.1109/ICSE.2015.299

Eickhoff, F. L., McGrath, M. L., Mayer, C., Bieswanger, A., & Wojciak, P. A. (2018). Large-scale application of IBM Design Thinking and *agile* development for IBM z14. *IBM Journal of Research and Development*, *62*(2–3), 11–19. https://doi.org/10.1147/JRD.2018.2795879

Ekstedt, M. (2013). An empirical approach to a general theory of software (engineering). *2013 2nd SEMAT Workshop on a General Theory of Software Engineering, GTSE 2013 - Proceedings*, 23–26. https://doi.org/10.1109/GTSE.2013.6613866

Fitriani, A. N., Raharjo, T., Hardian, B., & Prasetyo, A. (2021). IT infrastructure *agile* adoption for SD-WAN project implementation in pharmaceutical industry: Case study of an Indonesian Company. *2021 IEEE International IOT, Electronics and Mechatronics Conference, IEMTRONICS 2021 - Proceedings*. https://doi.org/10.1109/IEMTRONICS52119.2021.9422650

Fitriani, W. R., Rahayu, P., & Sensuse, D. I. (2016). Challenges in *agile* Software Development : A Systematic Literature Review. *2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 155–164. https://doi.org/10.1109/ICACSIS.2016.7872736

Foo, D., Cruz, J. Dela, Sekar, S., & Sharma, A. (2020). Automating Continuous Planning in SAFe, 504–504. https://doi.org/10.1145/3387940.3391536

Gandomani, T. J., & Nafchi, M. Z. (2015). An empirically-developed *requirement* for *agile* transition and adoption : A Grounded Theory approach. *The Journal of Systems & Software*, *107*, 204–219. https://doi.org/10.1016/j.jss.2015.06.006

Girma, M., Garcia, N. M., & Kifle, M. (2019). *agile* scrum *scaling* practices for large scale software development. *Proceedings - 2019 4th International Conference on Information Systems Engineering, ICISE 2019*, (Ld), 34–38. https://doi.org/10.1109/ICISE.2019.00014

Goodpasture, J. (2016). *Project Management the agile Way Making it Work in the Enterprise SECOND EDITION*. J. Ross Publishing.

Graziotin, D. (2012). A Web-based modeling tool for the SEMAT Essence theory of Software Engineering. *66*, עלון הנוטע(1), 39–37.

Gregor, S. (2006). The nature of theory in Information Systems. *MIS Quarterly: Management Information Systems*, *30*(3), 611–642. https://doi.org/10.2307/25148742

Gregor, S., & Hevner, A. R. (2013). Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly*, *37*(2), 337–355. https://doi.org/10.2753/MIS0742-1222240302

Gupta, M., George, J. F., & Xia, W. (2019). Relationships between IT department culture and *agile* software development practices : An empirical investigation. *International Journal of Information Management*, *44*(June 2018), 13–24. https://doi.org/10.1016/j.ijinfomgt.2018.09.006

Hastie, S., & Wojewoda, S. (2015a). Standish Group 2015 Chaos Report. *InfoQ*, 1–9. Retrieved from https://www.infoq.com/articles/standish-chaos-2015.

Hastie, & Wojewoda. (2015b). *Standish Group 2015 Chaos Report - Q&A with Jennifer Lynch*.

Helingo, M., Purwandari, B., Satria, R., & Solichah, I. (2017). The Use of Analytic Hierarchy Process for Software Development Method Selection: A Perspective of e-Government in Indonesia. In 4th Information Systems International Conference 2017, ISICO 2017, 6-8 November 2017, Bali, Indonesia

Heikkila, V. T., Paasivaara, M., & Lassenius, C. (2013). ScrumBut, but does it matter? A mixed-method study of the planning process of a multi-team scrum organization. *International Symposium on Empirical Software Engineering and Measurement*, 85–94. https://doi.org/10.1109/ESEM.2013.27

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly: Management Information Systems*, *28*(1), 75–105. https://doi.org/10.2307/25148625

Ieee. (2017). International Standard Systems and software engineering — Vocabulary, *Second Edi*.

Inayat, I., Salwah, S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). Computers in Human Behavior A systematic literature review on *agile requirement*s engineering practices and challenges, *51*, 915–929. https://doi.org/10.1016/j.chb.2014.10.046

Iso.org. (2017). Systems and Software Engineering Vocabulary. Retrieved from https://www.iso.org/standard/71952.html

Ivarsson, A., & Sunden, J. (2013). *agile* at scale at Spotify. Retrieved from https://www.*agile*alliance.org/resources/sessions/*agile*-at-scale-at-spotify/

Jacobson, I., Lawson, H., Ng, P.-W., McMahon, P., & Goedicke, M. (2019). *The Essentials of Modern Software Engineering: Free the Practices from the Method Prisons!* ACM Books. https://doi.org/10.16309/j.cnki.issn.1007-1776.2003.03.004

Jacobson, I., Ng, P. W., McMahon, P. E., Spence, I., & Lidman, S. (2012). The essence of software engineering: The SEMAT *kernel*. *Queue*, *10*(10), 40–51. https://doi.org/10.1145/2381996.2389616

Johannesson, P., & Perjons, E. (2014). *An Introduction to Design Science*. Heidelberg New York Dordrecht London: Springer International Publishing Switzerland.

Johnson, P., & Ekstedt, M. (2016). The Tarpit - A general theory of software engineering. *Information and Software Technology*, *70*, 181–203. https://doi.org/10.1016/j.infsof.2015.06.001

Johnson, P., Ekstedt, M., & Jacobson, I. (2012). Where's the theory for software engineering? *IEEE Software*, *29*(5), 94–95. https://doi.org/10.1109/MS.2012.127

Jovanovic, M., Mas, A., Mesquida, A., & Lali, B. (2017). Transition of organizational roles in *agile* transformation process : A grounded theory approach. *The Journal of Systems and Software*, *133*, 174–194. https://doi.org/10.1016/j.jss.2017.07.008

Kalenda, M., Hyna, P., & Rossi, B. (2018). Scaling *agile* in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, *30*(10). https://doi.org/10.1002/smr.1954

Karkukly, W. (2015). *Managing the PMO Lifecycle A Step by step to PMO set-up, Build Out and Sustainability Second Edition*. Canada: Friesen Press.

Kasauli, R., Knauss, E., Horkoff, J., Liebel, G., & de Oliveira Neto, F. G. (2020). Requirements engineering challenges and practices in large-scale *agile* system development. *Journal of Systems and Software*, 110851. https://doi.org/10.1016/j.jss.2020.110851

Kendall, G. (2003). *Advanced Project Portfolio Management and the PMO: Multiplying ROI at Warp Speed.* Boca Raton, Florida, United State of America: International Institute for Learning, Inc. and J. Ross Publishing, Inc.

Kerzner, H. (2006). *Project Management A Systems Approach to Planning, Scheduling, and Controlling*. Hoboken, New Jersey, United State of America: John Wiley & Sons.

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, *33*(TR/SE-0401), 28. https://doi.org/10.1.1.122.3308

Klopper, R., Grunner.S., Kourie. D.G. (2007). Assessment of a framework to compare software development methodologies

Laanti, M., & Kangas, M. (2015). Is *agile* Portfolio Management Following the Principles of

Large-Scale *agile*? Case Study in Finnish Broadcasting Company Yle. *Proceedings - 2015 agile Conference, agile 2015*, 92–96. https://doi.org/10.1109/*agile*.2015.9

Larman, C., & Vodde, B. (2016). *Large-Scaled Scrum More with LESS*. Addison-Wesley Professional.

Leffingwell, D. (2020). Scaled *agile* Framework. Retrieved from https://www.scaled*agileframework*.com

Leffingwell, D. L. (2007). *Scaling Software Agility: Best Practices for Large Enterprises (agile Software Development)*. Addison-Wesley Professional.

Maranzato, R. P., Neubert, M., & Herculano, P. (2012). Scaling scrum step by step: "The mega *framework*." *Proceedings - 2012 agile Conference, agile 2012*, 79–85. https://doi.org/10.1109/*agile*.2012.22

Marchewka, J. (2015). *Information Technology Project Management Providing Measurable Organizational Value Fifth Edition*. Danvers, United States: John Wiley & Sons, Inc.

Marchewka, J. (2016). *Information Technology Project Management*.

McKinsey&Company. (2018). *agile* compendium, (October).

Mendeley. (2019). About Mendeley. Mendeley. Retrieved from https://www.elsevier.com/solutions/mendeley

Miller, G. (2013). *agile* problems, challenges, & failures. Paper presented at PMI® Global Congress 2013—North America, New Orleans, LA. Newtown Square. PA: Project Management Institute. Retrieved from https://www.pmi.org/learning/library/*agile*-problems-challenges-failures-5869

Okoli, C., & Schabram, K. (2010). Working Papers on Information Systems A Guide to Conducting a Systematic Literature Review of Information Systems Research, *10*(2010).

Olszewska, M., Heidenberg, J., Weijola, M., Mikkonen, K., & Porres, I. (2016). Quantitatively measuring a large-scale *agile* transformation. *The Journal of Systems and Software*, *117*, 258–273. https://doi.org/10.1016/j.jss.2016.03.029

OMG. (2018). Kernel and Language for Software Engineering Methods (Essence). *Object Management Group*, (Versión 1.2), 300. Retrieved from https://www.omg.org/spec/Essence/1.2

Özcan-top, Ö., & Demirors, O. (2019). Computer Standards & Interfaces Application of a software agility assessment model – AgilityMod in the fi eld. *Computer Standards & Interfaces*, *62*(October 2017), 1–16. https://doi.org/10.1016/j.csi.2018.07.002

Paasivaara, M. (2017). Adopting SAFe to scale *agile* in a globally distributed organization. *Proceedings - 2017 IEEE 12th International Conference on Global Software Engineering,*

ICGSE 2017, 36–40. https://doi.org/10.1109/ICGSE.2017.15

Paasivaara, M., Heikkilä, V. T., & Lassenius, C. (2012). Experiences in *scaling* the Product Owner role in large-scale globally distributed Scrum. *Proceedings - 2012 IEEE 7th International Conference on Global Software Engineering, ICGSE 2012*, 174–178. https://doi.org/10.1109/ICGSE.2012.41

Paasivaara, M., & Lassenius, C. (2014). Deepening our understanding of communities of practice in large-scale *agile* development. *Proceedings - 2014 agile Conference, AGILE 2014*, 37–40. https://doi.org/10.1109/AGILE.2014.18

Paasivaara, M., & Lassenius, C. (2016). Scaling scrum in a large globally distributed organization: A case study. *Proceedings - 11th IEEE International Conference on Global Software Engineering, ICGSE 2016*, 74–83. https://doi.org/10.1109/ICGSE.2016.34

Padalkar, M., & Gopinath, S. (2016). ScienceDirect Six decades of project management research : Thematic trends and future opportunities. *JPMA*, *34*(7), 1305–1321. https://doi.org/10.1016/j.ijproman.2016.06.006

Pandya, A., Mani, V. S., & Pattanayak, A. (2020). Expanding the responsibility of an offshore team and sustainably increasing business value using SAFe, 1–5. https://doi.org/10.1145/3372787.3390441

Papadopoulos, G. (2015). Moving from traditional to *agile* software development methodologies also on large , distributed projects . *Procedia - Social and Behavioral Sciences*, *175*, 455–463. https://doi.org/10.1016/j.sbspro.2015.01.1223

Parizi, R. M., Gandomani, T. J., & Nafchi, M. Z. (2014). Hidden facilitators of *agile* transition: *agile* coaches and *agile* champions. *2014 8th Malaysian Software Engineering Conference, MySEC 2014*, 246–250. https://doi.org/10.1109/MySec.2014.6986022

Park, J. S., McMahon, P. E., & Myburgh, B. (2016). Scrum Powered by Essence. ACM SIGSOFT Software Engineering Notes.

Patanakul, P., & Rufo-mccarron, R. (2018). Transitioning to *agile* software development : Lessons learned from a government-contracted program. *Journal of High Technology Management Research*, *29*(2), 181–192. https://doi.org/10.1016/j.hitech.2018.10.002

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*. https://doi.org/10.2753/MIS0742-1222240302

Phillips, J. (2019). *PMI-ACP agile Certified Practitioner All-in-One Exam Guide*. McGraw-Hill Education.

PMI. (2017a). *A Guide to the PROJECT MANAGEMENT BODY OF KNOWLEDGE*. Project

Management Institute, Inc.

PMI. (2017b). *agile Practice Guide*. Project Management Institute, Inc.

PMI. (2017c). *PMI's PULSE of the PROFESION 9TH Global Project Management Survey*.

PMI. (2019). Discipline *agile*.

Pries-Heje, J., & Krohn, M. M. (2017). The SAFe way to the *agile* organization. *ACM International Conference Proceeding Series*, *Part F1299*, 19–21. https://doi.org/10.1145/3120459.3120478

Raharjo, T., & Purwandari, B. (2020a). *agile* Project Management Challenges and Mapping Solutions-A Systematic Literature Review-V10b-TRACKED_REVISED. International Conference on Software Engineering and Information System.

Raharjo, T., & Purwandari, B. (2020b). *agile* Project Management Challenges and Mapping Solutions: A Systematic Literature Review. *International Conference on Software Engineering and Information Management*.

Rational. (1998). Rational Unified Process Best Practices for Software Development Teams.

Razzak, M. A., Richardson, I., Noll, J., Canna, C. N., & Beecham, S. (2018). Scaling *agile* across the Global Organization: An Early Stage Industrial SAFe Self-Assessment. *2018 IEEE/ACM 13th International Conference on Global Software Engineering (ICGSE)*, 121–130. https://doi.org/10.1145/3196369.3196373

Recker, J. (2013). *Scientific Research in Information Systems - A Beginner's Guide*. Heidelberg New York Dordrecht London: Springer International Publishing -Verlag Berlin Heidelberg.

Rigby, D., Sutherland, J., & Noble, A. (2018). *Change management - agile at Scale*. Retrieved from https://hbr.org/2018/05/*agile*-at-scale

Rose, D. (2018). *Enterprise Agility*. Jon Willey & Sons, Inc.

Salameh, A., & Bass, J. M. (2020). Heterogeneous Tailoring Approach Using the Spotify Model. *ACM International Conference Proceeding Series*, 293–298. https://doi.org/10.1145/3383219.3383251

Sánchez-morcilio, R., Rico, P., & Campus, R. P. (2016). Issues in Information Systems, *17*(Iii), 187–198.

Schwaber, K. (2004). *agile Project Management with Scrum*. Microsoft.

Scrum.org. (2018). Online Nexus Guide. Retrieved from https://www.scrum.org/resources/online-nexus-guide

Scrum.org. (2019). Scrum Master Trends.

Scrumguide.org. (2017). *The Scrum Guide*. Retrieved from

https://www.scrumguides.org/scrum-guide.html

SEI. (2006). CMMI ® for Development Version 1 . 2 Improving processes for better products, (August).

Sekaran, U. R. B. (2013). *Research Methods for Business: A Skill-Building Approach 6th Edition*.

Semat.org. (2020). Software Engineering Methods and Theory. Retrieved from semat.org

Shameem, M., Khan, A. A., Gulzarul Hasan, M., & Akbar, M. A. (2020). Analytic hierarchy process based prioritisation and taxonomy of success factors for *scaling agile* methods in global software development. *IET Software*, *14*(4), 389–401. https://doi.org/10.1049/iet-sen.2019.0196

Smite, D., Moe, N. B., Levinta, G., & Floryan, M. (2019). Spotify Guilds: How to Succeed with Knowledge Sharing in Large-Scale *agile* Organizations. *IEEE Software*, *36*(2), 51–56. https://doi.org/10.1109/MS.2018.2886178

Smolander, K., & Paivarinta, T. (2013). Forming theories of practices for software engineering. *2013 2nd SEMAT Workshop on a General Theory of Software Engineering, GTSE 2013 - Proceedings*, 27–34. https://doi.org/10.1109/GTSE.2013.6613867

Stojanov, I., Turetken, O., & Trienekens, J. J. M. (2015). A Maturity Model for Scaling *agile* Development. *Proceedings - 41st Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2015*, 446–453. https://doi.org/10.1109/SEAA.2015.29

Stol, K. J., & Fitzgerald, B. (2013). Uncovering theories in software engineering. *2013 2nd SEMAT Workshop on a General Theory of Software Engineering, GTSE 2013 - Proceedings*, 5–14. https://doi.org/10.1109/GTSE.2013.6613863

Sugiyono. (2015). *Metode Penelitian Manajemen Pendekatan Kuantitatif Kualitatif Kombinasi Penelitian Tindakan dan Penelitian Evaluasi*. Bandung, Indonesia: Penerbit Alfabeta.

Tjahjana, L. (2009). *The Program Management Office Advantage A Powerful and Centralized Way for Organizations to Manage Projects.* New York, United State of America: American Management Association.

Uludag, O., Kleehaus, M., Caprano, C., & Matthes, F. (2018). Identifying and structuring challenges in large-scale *agile* development based on a structured literature review. *Proceedings - 2018 IEEE 22nd International Enterprise Distributed Object Computing Conference, EDOC 2018*, 191–197. https://doi.org/10.1109/EDOC.2018.00032

Uludag, O., Kleehaus, M., Dreymann, N., Kabelin, C., & Matthes, F. (2019). Investigating the Adoption and Application of Large-Scale Scrum at a German Automobile Manufacturer. *Proceedings - 2019 ACM/IEEE 14th International Conference on Global Software*

Engineering, ICGSE 2019, 22–29. https://doi.org/10.1109/ICGSE.2019.00019

VersionOne. (2016). *State of agile Report 11th Annual*.

VersionOne. (2019). The 13th annual STATE OF AGILE Report - 2018. *CollabNe t | VersionOne*, *13*, 16.

Wahyuni, S. (2012). *Qualitative Research Method Theory and Parctice*. Penerbit Salemba Empat.

zachman.com. (2019). Zachman Framework. Retrieved from https://www.zachman.com/about-the-zachman-*framework*