I-STATEMENT
STIMIK ESQ | I-4
Volume 2 Nomor 1, Januari 2016

ISSN : 2442-8337

Penggunaan Pendekatan CMMI dalam Metodologi Agile Development

Anggar Riskinanto

Program Studi Sistem Informasi, STIMIK ESQ
Jl. TB Simatupang Kavling 1, Cilandak, Jakarta Selatan – 12560
Email: anggar.r@esqbs.ac.id

Abstract*: Developing software is actually different from creating software, because there are some rules that must be followed to produce high-quality solutions. CMMI tries help developers to achieve this. In this framework, there are processes that must be met in order for the solution's quality becomes higher. To achieve this, the documentation process is indispensable. This is somewhat different from the methodology of Agile Development, where it focused on how an application is made as early as possible, produce prototypes (prototype) without the need to create a document for each process.*

This writing seeks to discuss a staged CMMI approach in which it is used in one of the Agile Development methodology, namely Scrum. Problems that occur when combining the two is an error in use, the lack of accurate information, and the difficulty of terminology. Solutions to these problems can be solved through a mapping table of CMMI processes to Scrum activity.

Keywords*: Software Development, CMMI, Agile Development, Scrum, Mapping Table.*

Abstrak: Pengembangan *software* sejatinya berbeda dengan membuat *software*, karena ada beberapa kaidah yang harus diikuti untuk menghasilkan solusi berkualitas tinggi. CMMI mencoba membantu para developer untuk mencapai hal ini. Pada *framework* ini, terdapat proses-proses yang harus dipenuhi agar kualitas solusi yang dihasilkan menjadi tinggi. Untuk mencapai ini, proses dokumentasi sangat diperlukan. Hal ini agak berbeda dengan metodologi Agile Development, dimana ia menitik beratkan pada bagaimana sebuah aplikasi dibuat sedini mungkin, menghasilkan purwarupa (*prototype*) tanpa harus membuat dokumen pada tiap prosesnya.

Tulisan ini berusaha membahas sebuah pendekatan CMMI *staged* dimana ia digunakan ke dalam salah satu metodologi Agile Development, yaitu Scrum. Permasalahan yang terjadi ketika mengombinasikan keduanya adalah adanya kesalahan dalam penggunaan, kurangnya informasi yang akurat, dan kesulitan terminologi. Solusi terhadap permasalahan ini dapat dipecahkan melalui tabel *mapping* dari proses-proses CMMI ke aktivitas Scrum.

Kata Kunci: Pengembangan *Software*, CMMI, Agile Development, Scrum, Tabel *Mapping*

1. PENDAHULUAN

CMMI (Capability Maturity Model Integration) merupakan salah satu solusi yang berusaha membantu proses-proses yang ada dalam pengembangan *software* menjadi lebih teratur, *manageable*, bahkan dapat diukur. Hingga hasil akhirnya bisa diadakan suatu perbaikan dari proses-proses yang ada. Namun perlu disadari bahwa

CMMI hanyalah sebuah *framework* bagi proses pengembangan. Untuk melengkapi hal ini dibutuhkan metodologi untuk menjalankan semua proses pengembangan *software* dari awal hingga akhir.

Salah satu dari beberapa metodologi yang ada adalah Agile Development. Metodologi ini lebih menitik beratkan pada *release product* secara bertahap dan menggunakan iterasi yang lebih sering. Pengembangan ini sebenarnya juga dapat digunakan bersama-sama dengan pemodelan CMMI.

Penulisan ini mencoba memfokuskan pada salah satu pendekatan di dalam CMMI, yaitu Staged dan penggunaan Scrum sebagai salah satu metode dalam Agile Development.

Terdapat perbedaan yang cukup besar pada keduanya dalam hal menjalankan proses-proses yang ada. Penulisan ini juga berusaha untuk menjelaskan permasalahan-permasalahan yang ada ketika mengimplementasikan kedua cara ini. Kemudian dijabarkan solusi atau alternatif yang ada atas permasalahan-permasalahan tersebut.

1.1. CMMI

CMMI merupakan sebuah sekumpulan pengetahuan atau *best practices* untuk membantu organisasi dalam menjalankan proses-proses dalam pengembangan *software* menjadi lebih baik. Bertujuan agar performa organisasi menjadi lebih efektif dan efisien dalam aktifitas pengembangan *software*.

Proses- proses yang ada pada *framework* ini dinamakan sebagai PA (Process Area) yang merupakan sebuah kumpulan dari *practices* yang saling berhubungan, yang bila diaplikasikan secara kolektif akan memenuhi sebuah *goal* yang akhirnya akan membuat perbaikan pada suatu area. Setiap PA yang ada mempunyai beberapa *goal* atau sasaran yang harus dicapai. Dimana *goals* tersebut terbagi menjadi Specific Goals (SG) dan Generic Goals (GG). SG adalah *goals* yang harus dicapai hanya pada satu PA tertentu, sedangkan GG merupakan *goals* yang harus dicapai pada semua PA

CMMI berfokus pada proses dalam pengembangan *software* di sebuah organisasi. Sehingga *framework* ini dapat dianalogikan sebagai sebuah model yang dapat diubah atau bisa diimplementasikan sebagian oleh organisasi. Jika dibanding menjadi sebuah standar yang baku dan mewajibkan organisasi mengaplikasikan semua proses-proses yang ada.

CMMI terdiri dua representasi atau pendekatan, yaitu Staged dan Continous[3]. Pendekatan Staged memungkinkan sebuah organisasi untuk mencapai tingkat kematangan (*maturity level*) yang tinggi. Sedangkan pendekatan Continous memungkinkan organisasi memfokuskan pada PA tertentu untuk mencapai tingkat kemampuan (*capability level*) yang tinggi.

1.2. Pendekatan Staged

Pada pendekatan ini organisasi diharuskan melewati beberapa tahapan atau tingkatan untuk mencapai tingkat kemapanan yang tinggi. Tingkatan yang dicapai oleh organisasi bersifat kumulatif, dimana tingkatan yang di atas harus juga memenuhi setiap *goals* pada level di bawahnya.

Dikarenakan tingkat kematangan yang harus dipenuhi oleh organisasi bertahap, maka dikenal akan adanya Maturity Level (ML). Pendekatan ini mengenal 5 ML [3], yaitu:

- ML 1: Initial
- ML 2: Managed
- ML3: Defined
- ML 4: Quantitatively Managed
- ML 5: Optimizing

Semua ML diatas menjelaskan tahapan-tahapan yang harus dilalui oleh sebuah organisasi untuk mendapatkan suatu proses pengembangan *software* yang lebih efektif dan efisien. Daftar diatas juga menjelaskan mengenai evolusi proses-proses yang ada dari keadaan *chaos* atau kacau menjadi sangat terorganisir bahkan teroptimasi.

1.3. Agile Development

Agile Development adalah suatu metodologi pengembangan *software* yang menekankan pada proses yang iteratif dan inkremental. Dimana *requirement* dan solusi yang ada merupakan hasil kolaborasi antara *developer* dan *stakeholder*.

Metodologi ini mempunyai pedoman yang dinamakan Agile Manifesto, yang terdiri dari:

- *Individuals and interactions over processes and tools.*
- *Working software over comprehensive documentation.*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*

Dari manifesto di atas, bisa terlihat bahwa metodologi ini berfokus pada interaksi sesama pihak yang terlibat. Ia juga menekankan pada pengembangan *software* itu sendiri dibanding pembuatan dokumentasi yang mendalam. Selain juga merespon pada perubahan terhadap perencanaan yang telah dibuat.

Beberapa penerapan metodologi Agile Development yang cukup dikenal luas adalah sebagai berikut:

- Agile Modelling
- Extreme Programming (XP)
- Scrum

Perbedaan pada setiap metode diatas terletak pada praktek atau penerapan yang dilakukan dalam mengembangkan *software*.

1.4. Scrum

Scrum merupakan salah satu penerapan atau metode dalam Agile Development. Ciri khas dari metode ini adalah pada *roles* dan *meeting* selama proses pengembangan *software*. Roles merupakan representasi dari pihak-pihak yang terlibat didalam proses pengembangan. Beberapa *roles* yang terdapat dalam Scrum adalah:

- Product Owner: pihak yang mewakili *stakeholder* dan memastikan *software* memenuhi keinginan klien.
- Scrum Master: pihak yang memastikan proses pengembangan *software* berjalan semestinya.
- Team Member: pihak yang menjalankan proses pengembangan *software*.

Pihak-pihak di atas berperan secara langsung dalam mengembangkan *software*. Untuk itu perlu diadakan *meeting* rutin di antara semua pihak. *Meeting* yang diadakan biasanya berdurasi pendek, sekitar 15 menit.

Selain itu Scrum juga mengenal istilah Sprint dan Backlogs dalam prosesnya. Sprint adalah proses pengembangan itu sendiri yang biasanya berjarak sekitar 2 minggu hingga 1 bulan. Sedangkan Backlog adalah hasil dari setiap proses pengembangan yang terjadi pada satu iterasi.

1.5. CMMI dan Agile Development

Penggunaan CMMI ke dalam metodologi Agile Development, terutama Scrum sangat dimungkinkan. Bahkan penggunaan keduanya dapat meningkatkan kinerja dan menghasilkan produk yang lebih berkualitas. Pada saat yang sama tetap memenuhi kesesuaian terhadap proses-proses area pada CMMI [7].

2. PERMASALAHAN DAN PEMECAHAN

Meski penggunaan CMMI dan Agile Development dimungkinkan, namun pada prakteknya tetap terjadi beberapa permasalahan dalam implementasi keduanya. Berikut adalah beberapa contohnya [2]:

- Kesalahan dalam penggunaan.

  Kesalahan dalam penggunaan terutama terjadi pada organisasi yang ingin menerapkan model CMMI. Mereka menganggap model ini sebagai suatu standar baku yang tidak bisa berubah dibanding dengan sebuah model yang fleksibel. Salah satu kesalahan yang terjadi adalah organisasi yang keliru menggunakan *rating appraisal* sebagai bentuk pengukuran kinerja bisnis. Hal yang terjadi kemudian adalah salah penerapan model menjadi sebuah standar ketika organisasi ingin membangun suatu *software* atau solusi.

- Kurangnya informasi yang akurat.
  Kurangnya informasi yang akurat juga menjadi masalah lain bagi organisasi yang akan menerapkan model CMMI ke dalam Agile Development. Hal ini terutama terjadi pada Agile Development itu sendiri, dimana literatur mengenai metodologi ini lebih sedikit jumlahnya dibanding dengan literatur mengenai pemodelan CMMI. Hal ini menyebabkan tidak banyaknya organisasi yang menggunakan kedua kombinasi ini.
- Kesulitan terminologi.
  Terminologi-terminologi yang ada berperan serta menghambat organisasi untuk mengimplementasikan keduanya. Terdapat banyak istilah yang sama penamaannya, tetapi mempunyai arti berbeda. Selain itu terdapat juga istilah-istilah yang berbeda, yang sebenarnya mempunyai makna yang sama.

Terhadap permasalahan yang ada, maka terdapat alternatif atau solusi sebagai berikut:

- *Mapping* CMMI dengan Agile Development
  Meski terdapat perbedaan istilah diantara keduanya, namun tetap dimungkinkan adanya *mapping* atau pemetaan terhadap proses-proses yang ada pada CMMI ke dalam aktivitas pada metode Scrum. Berikut adalah contoh tabel *mapping* pada PA Requirement Management (REQM) dengan Scrum [6]:

Tabel 1. Mapping PA Requirement Management (REQM) dengan Scrum.

REQM	CMMI Practice	Scrum Practice
SP 1.1	Develop an understanding with the requirements providers on the meaning of the requirements.	• Review of Product Backlog (requirements) with Product Owner and team.
SP 1.2	Obtain commitment to the requirements from the project participants.	• Release Planning and Sprint Planning sessions that seek team member commitment.
SP 1.3	Manage changes to the requirements as they evolve during the project.	• Add requirements changes to the Product Backlog. • Manage changes in the next Sprint Planning meeting.
SP 1.5	Identify inconsistencies between the project plans and work products and the requirements.	• Daily Standup Meeting to identify issues. • Release planning and Sprint Planning sessions to address inconsistencies. • Sprint Burndown chart that tracks effort remaining. • Release Burndown chart that tracks story points that have been completed. This shows how much of the product functionality is left to complete.

- Memanfaatkan *practises* CMMI hingga Maturity Level 3

  Meski pemodelan CMMI dapat digabungkan dengan metode Scrum, namun hasil terbaik yang dapat menghasilkan produk berkualitas tinggi adalah memanfaatkan *practices* yang ada pada CMMI hanya hingga ML 3 [7]. Hal ini bisa terjadi karena beberapa *practices* yang ada pada CMMI tidak mempunyai padanan pada metode Scrum [4,6]. Mengimplementasikan *practices* pada Level 4 ke atas hanya akan menghambat proses pengembangan *software* pada metode Scrum. Karena pada level-level ini dibutuhkan sumber daya yang lebih untuk memenuhi *goals* yang ada, yang pada akhirnya akan membuat metode Scrum tidak “lincah” lagi.

3. KESIMPULAN

Akhirnya, bisa disimpulkan bahwa dengan menerapkan beberapa solusi, organisasi pengembang *software* dapat memadukan kedua *framework* dan metode ini untuk menghasilkan produk yang berkualitas tinggi, yang memenuhi kaidah *on-budget, on-schedule,* dan *on-scope*.

DAFTAR PUSTAKA

[1] R. Coffin and D. Lane, “A Practical Guide to Seven Agile Methodologies”, http://www.devx.com/architect/Article/32761, 2006

[2] H. Glazer, J. Dalton, D. Anderson, M. Konrad, and S. Shrum, "CMMI® or Agile: Why Not Embrace Both!", 2008

[3] R. Kneuper, *CMMI:Improving Software and Systems Development Processes Using Capability Maturity Model Integration (CMMI-Dev)*, Rocky Nook, 2009

[4] A. S. C. Marçal, B. C. C. de Freitas, F. S. F. Soares, and A. D. Belchior, “Mapping CMMI Project Management Process Areas to SCRUM Practices”, 2007

[5] M. Pikkarainen and A. Mäntyniemi, “An Approach for Using CMMI in Agile Software Development Assessments: Experiences from Three Case Studies”, 2006

[6] N. Potter, “Comparing Scrum And CMMI How Can They Work Together”, The Process Group, 2010

[7] J. Sutherland, C. R. Jakobsen, and K. Johnson, “Scrum and CMMI Level 5: The Magic Potion for Code Warriors”, 2008

This page intentionally left blank