

PENILAIAN RISIKO PADA PROYEK PERANGKAT LUNAK OUTSOURCING MENGGUNAKAN CMMI *for DEVELOPMENT* 1.3 STUDI KASUS DI PT. XYZ

Maya Eva Rahmmawati, Yeffry Handoko
Magister Sistem Informasi
Universitas Komputer Indonesia
Bandung

e-mail : <mailto:meeeva.me@gmail.com>, yeffryhp@unikom.ac.id

Abstrak—Proses pengembangan perangkat lunak dengan menggunakan *outsourcing* memiliki banyak kelebihan, diantaranya mengurangi dan mengendalikan biaya operasional serta meningkatkan fokus perusahaan pada kegiatan utama. Akan tetapi pada proses pengembangan ini, sering ditemukan banyak risiko yang berdampak pada kualitas produk yang dihasilkan. Misalnya waktu pengerjaan yang melebihi estimasi, permintaan yang berubah-ubah maupun kurangnya sumber daya manusia. Tujuan dari penulisan makalah ini adalah untuk mengetahui tingkat kapabilitas dan tingkat kematangan proses pengembangan perangkat lunak *outsourcing* pada PT. XYZ. Model yang digunakan untuk analisis yaitu menggunakan CMMI *for Development* 1.3. Sedangkan metode yang digunakan yaitu menggunakan representasi continuous dengan memilih praktik-praktik spesifik pada proses area CMMI level 2 dan level 3. Pengambilan data dilakukan dengan kuisioner pada responden yang merupakan karyawan PT. XYZ, yaitu manajer proyek dan programmer. Hasil penelitian ini dihasilkan sebuah saran-saran yang sesuai untuk meminimalkan risiko-risiko yang terjadi. Dengan memahami hasil dari penelitian ini, perusahaan dapat memiliki strategi pada setiap tahapan proses pengembangan perangkat lunak *outsourcing* yang berorientasi pada kualitas produk perangkat lunak yang dihasilkan.

Kata Kunci: Risiko, *Outsourcing*, Perangkat Lunak, CMMI.

I. PENDAHULUAN

Manajemen proyek memainkan peran penting dalam kedua proyek perangkat lunak yaitu *in-house* dan *outsourcing*. Hal ini dapat membantu sebuah proyek pengembangan perangkat lunak menghindari kegagalan, yaitu tetap beroperasi atau membatalkan proyek tersebut. Jadi, bagaimana cara mengidentifikasi, mengelola dan menghilangkan risiko ini sebelum menjadi ancaman bagi keberhasilan proyek perangkat

lunak adalah tujuan utama dari manajemen risiko perangkat lunak [4].

Berdasarkan *Standish Group 2015 Chaos Report*, hasil penelitian menunjukkan masih ada pekerjaan yang harus dilakukan untuk mencapai keberhasilan dari proyek pengembangan perangkat lunak. Tabel 1.1 ini merangkum hasil proyek selama lima tahun terakhir menggunakan definisi baru dari faktor keberhasilan (tepat waktu, anggaran dengan hasil yang memuaskan) [11].

Tabel 1.1 Keberhasilan proyek perangkat lunak dalam lima tahun terakhir

	2011	2012	2013	2014	2015
Successful	29%	27%	31%	28%	29
Challenged	49%	56%	50%	55%	52%
Failed	22%	17%	19%	17%	19%

Secara umum menentukan kualitas produk perangkat lunak dibagi menjadi dua pendekatan yaitu pendekatan proses produksi dan pendekatan hasil pengujian produk. Pendekatan proses produksi memegang peranan signifikan dalam menghasilkan suatu produk perangkat lunak berkualitas yang dilakukan oleh tim pengembang. Sangat penting bagi perusahaan memahami setiap proses area dalam pengembangan perangkat lunak sehingga dapat menggunakan *outsourcing* yang tetap berorientasi pada kualitas produk [3].

Terdapat beberapa model yang dirancang fokus pada evaluasi perbaikan proses, menilai level kematangan dan level kapabilitas tim pengembang perangkat lunak seperti *Six Sigma*, *ISO 9001*, *Bootstrap*, *Kipi*, dan *CMMI for Development*. *CMMI for development* merupakan model yang cukup populer digunakan pada perbaikan proses pengembangan perangkat lunak [1] [10]. Model *CMMI* dapat

digunakan untuk mengetahui level kapabilitas sekaligus level kematangan sebuah tim pengembang perangkat lunak yang disebut sebagai equivalent staging. Dengan demikian target proses area yang akan digunakan untuk evaluasi dimulai dari level 2 dan level 3 dengan metode representasi *continuous*. Pada level kematangan 2 merupakan tahap awal dimana seluruh proses pengembangan perangkat lunak oleh kontraktor seharusnya telah terencana, terstruktur, dan terkontrol dengan baik. Sedangkan proses area pada level 3, kontraktor seharusnya telah memiliki standarisasi prosedur, *tools*, dan metode kerja sehingga ada jaminan konsistensi dalam pelaksanaan proyek.

Pada penelitian ini, responden yang akan digunakan adalah karyawan pada PT. XYZ. Dengan memahami tahapan pada proses area pengembangan perangkat lunak, diharapkan perusahaan dapat memiliki strategi pada setiap tahapan proses pengembangan perangkat lunak skema *outsourcing* yang berorientasi pada kualitas produk.

II. LANDASAN TEORI

Perbaikan proses pengembangan perangkat lunak mencakup serangkaian aktivitas menuju yang lebih baik, sebagai akibatnya kualitas perangkat lunak menjadi lebih tinggi. Kualitas perangkat lunak bukan hanya berdampak pada kepuasan pelanggan (user satisfaction) saja, tetapi dampaknya justru lebih besar ada pada tim pengembang. Dapat dianalogikan bahwa perangkat lunak adalah sebuah makhluk yang terus berevolusi sesuai dengan kebutuhan pengguna. Penambahan fitur, pengurangan fitur, perubahan proses bisnis yang akan berakibat pada reengineering. Apabila kualitas perangkat lunak tinggi, maka akan lebih mudah bagi para engineer untuk melakukan perubahan. Tujuan dari perbaikan proses pengembangan perangkat lunak adalah untuk memahami proses yang ada saat ini, memperkenalkan perubahan proses untuk meningkatkan kualitas, mengurangi biaya, atau mempercepat jadwal, tuntutan industri terhadap peningkatan kualitas produksinya, dan untuk mengurangi resiko cacat perangkat lunak yang diakibatkan dari kesalahan yang terjadi saat proses pengembangan.

A. Penyebab Risiko

1. *Quality Constraints* (Kendala dalam kualitas)

Kualitas adalah perhatian utama untuk produk-produk perangkat lunak. Vendor perangkat lunak menyadari bahwa jauh lebih murah untuk membuat produk software berkualitas baik dengan akhir biaya menjadi rendah daripada untuk menghasilkan produk software berkualitas buruk dan berakhir dengan biaya yang tinggi.

2. *Resource Unavailability* (Ketidakterediaan Sumber Daya)

Sumber daya yang tidak tersedia merupakan salah satu faktor risiko utama, sebagaimana pengembang perangkat lunak profesional yang diminati di seluruh dunia. Pengembang perangkat lunak profesional yang baik adalah kunci utama dalam pembuatan perangkat lunak.

3. *Disinterest* (Ketidaktertarikan)

Kurangnya minat adalah kekhawatiran yang perlu diatasi

oleh manajer proyek karena sangat berpengaruh terhadap produktivitas proyek. Dibutuhkan motivasi yang baik bagi individu yang kurang berminat, sehingga risiko dapat dikurangi.

4. *Attrition* (Gesekan)

Karena permintaan yang tinggi untuk perangkat lunak profesional, sebagian besar pengembang perangkat lunak profesional memiliki banyak pekerjaan lain pada waktu tertentu. Ketika mereka membutuhkan penghasilan yang lebih, maka mereka akan berhenti dari perusahaan untuk bergabung dengan perusahaan lain, sehingga meninggalkan proyek di tengah-tengah waktu pengerjaan. Hal ini tengah menjadi masalah besar di perusahaan, akan tetapi pada perusahaan besar telah memiliki program khusus untuk mengatasi itu.

5. *Scope Creep* (Perubahan yang Tidak Terkontrol)

Scope creep adalah salah satu risiko bahwa proyek-proyek perangkat lunak yang paling dan selalu berdampak kurang bagus. Permintaan terus berubah dan permintaan baru terus menumpuk bahkan setelah proyek telah masuk ketahap pengujian dan ketahap implementasi. Sebuah mekanisme perubahan manajemen yang baik dapat mengatasi ancaman ini secara efektif.

6. *Cost Constraints* (Kendala Biaya)

Setelah proyek disetujui untuk dimulai, anggaran dialokasikan untuk proyek tersebut. Namun karena alasan yang tidak dapat dihindari, anggaran dapat dibatasi. Dalam situasi seperti itu, proyek tidak dapat dilanjutkan karena sumber dana tidak mencukupi dan biaya proyek tidak dapat dipenuhi. Ada solusi untuk masalah ini, tetapi jika risiko ini diketahui di awal, maka proyek bisa dipotong dan risiko proyek dapat dihindari.

7. *Bad Negotiation* (Negosiasi yang Buruk)

Jika manajer proyek memiliki kemampuan negosiasi yang baik, maka ia bisa mendapatkan tambahan / modifikasi anggaran, dukungan, dan sumber daya, setiap kali diperlukan. Tapi kadang-kadang karena keterampilan negosiasi yang buruk atau karena kurangnya pandangan pada bagian dari pelanggan, jenis fasilitas ini tidak didapatkan.

8. *Unrealistic Estimate* (Estimasi yang Tidak Realistis)

Perkiraan yang tidak realistis merupakan resiko lain yang sangat umum di sebagian besar proyek. Hal ini juga merupakan fakta bahwa dampak dari perkiraan untuk proyek-proyek perangkat lunak sulit untuk membuat karena ketidakpastian yang terlibat. Jadi, diharapkan harus selalu waspada. Selalu lebih baik membuat perkiraan, untuk mengurus ketidakpastian.

9. *Human Error* (Kesalahan Manusia)

Otak manusia memiliki kekuatan pemrosesan yang tidak ada satupun komputer bisa mengimbangi, tetapi memiliki keterbatasan. Tidak dapat melakukan pekerjaan berulang-ulang tanpa membuat kesalahan. Kesalahan manusia disebabkan oleh gangguan otak karena otak kita terus memproses semua sinyal yang dikirim oleh alat indera kita terus menerus, dan dengan demikian, pekerjaan yang kita

lakukan kurang mendapat perhatian, dan dapat menghasilkan kesalahan dalam pekerjaan. Untuk mengatasi hal ini, kita harus memiliki proses review untuk pekerjaan yang dilakukan untuk menghilangkan kekurangan.

10. *Poor Management* (Manajemen yang Buruk)

Manajemen yang buruk adalah faktor risiko dari manusia yang lain. Tidak semua manajer proyek secara alami berbakat. Banyak dari mereka belajar mengelola hal-hal ini dari pengalaman. Jika seorang manajer proyek tidak memiliki pengalaman dalam mengelola proyek, maka itu adalah kewajiban besar untuk proyek dan itu akan muncul dalam hasil proyek. Bahkan jika seorang manajer proyek memiliki pengalaman, sifat-sifat pribadi menentukan apakah dia dapat menangani proyek dengan baik atau tidak.

B. Kategori Risiko

Semua risiko yang disebutkan di bagian sebelumnya benar-benar dapat dikelompokkan menjadi [9]:

1. *Budget Risk* (Risiko Biaya)

Risiko yang berdampak pada anggaran proyek perlu menjadi pertimbangan utama, dan perlu dikendalikan. Jika untuk beberapa alasan anggaran berjalan di atas batas yang diperbolehkan, maka perusahaan harus melakukan sesuatu untuk mengontrolnya.

2. *Time/ Schedule Risk* (Risiko Jadwal)

Bisnis perlahan-lahan menyusut dalam lingkungan yang serba cepat dan berubah pada saat ini. Jadi, jika proyek terlihat semakin melebihi dari tanggal ditargetkan, maka akan menjadi kerugian peluang bisnis yang besar. Untuk alasan ini, proyek tidak boleh melebihi dari tanggal rilis yang ditargetkan. Namun demikian, karena beberapa keadaan, lebihnya tanggal proyek akan berpengaruh terhadap hal lainnya.

3. *Resource Risk* (Risiko Sumber Daya)

Anggota tim proyek adalah sumber daya yang paling mahal dalam proyek perangkat lunak. Di satu sisi, pihak manajemen perlu untuk menjaga biaya proyek di batas minimal, dan di sisi lain, harus membuat ketentuan untuk sumber daya yang dipesan proyek sebagai ketentuan untuk setiap risiko kehilangan anggota tim proyek setiap saat selama masa pelaksanaan proyek tersebut.

4. *Quality Risk* (Risiko Kualitas)

Industri perangkat lunak membutuhkan kehandalan yang solid sehingga selama proyek berlangsung, biaya dapat diminimalkan. Jika tidak, sebuah produk perangkat lunak berkualitas buruk justru akan membuat biaya semakin banyak. Kualitas produk dapat menjadi tidak berkualitas karena desain perangkat lunak yang buruk atau konstruksi perangkat lunak yang buruk.

5. *Technology Risk* (Risiko Teknologi)

Teknologi usang adalah fakta kehidupan. Dengan diperkenalkannya cepat produk baru ke pasar, produk yang lebih tua dengan cepat menjadi usang. Jadi, banyak proyek menghadapi prospek teknologi usang di mana produk

perangkat lunak yang sedang dibangun. Dalam kasus tersebut, produk perangkat lunak menjadi tidak dapat digunakan bahkan sebelum diimplementasikan.

C. Capability Maturity Model Integration

Menurut *CMMI Product Team*, *CMMI* merupakan suatu kumpulan praktik-praktik terbaik (*best practice's*) untuk membantu organisasi dalam meningkatkan proses pengembangan aplikasi mereka. Model ini dikembangkan oleh tim yang terdiri dari praktisi dunia industry, pemerintah dan *Software Engineering Institute* (SEI) (*CMMI Product Team*, 2010) [1].

Berdasarkan panduan *CMMI for Development* versi 1.3, terdapat dua macam representasi yang dapat disajikan menggunakan *CMMI-DEV*. Model *CMMI* memungkinkan dua pendekatan/representasi dari apa yang disebut *Quality Management*: representasi *continous* dan representasi *staged*. Pemilihan representasi dipengaruhi oleh tiga hal yaitu : faktor bisnis, faktor budaya, dan juga faktor pengalaman dalam penggunaan model-model sebelumnya. Pendekatan *continous* ditempuh bila upaya peningkatan fokus pada area proses tertentu saja, sedangkan pendekatan *staged* dirancang sebagai suatu tahapan-tahapan baku bagi peningkatan sekumpulan proses dalam sebuah organisasi. Pada penelitian ini akan menggunakan representasi *continuous*.

D. Standar Proses Evaluasi CMMI

Dalam penerapan *CMMI*, dilakukan pengukuran untuk mengevaluasi hasil pencapaian penerapannya. Proses evaluasi ini disebut dengan istilah *appraisal*. Metode standar yang digunakan dalam melakukan *appraisal*, yang dikeluarkan oleh SEI adalah *Standard CMMI Appraisal Method for Process Improvement*. SEI menggolongkan metode *appraisal* menjadi 3 kelas, masing-masing dengan dengan tingkat kelengkapan, besar usaha pelaksanaan dan requirement yang berbeda-beda. Kelas tersebut adalah class A, class B dan class C. Pada Tabel 2.1. berikut ini penjabaran detail mengenai perbandingan 3 kelas *appraisal method* [2].

Tabel 2.1 Perbandingan Kelas *Appraisal Method* (Sumber: Kneuper, 2009)

Metriks	Class A	Class B	Class C
Ukuran tim evaluasi	8-10	3-4	1-2
Waktu evaluasi	10 hari	3-4 hari	1-2 hari
Minimal jumlah sumber data	3	2	1
Keandalan dan keberhasilan	Tinggi	Sedang	Rendah
Usaha dan biaya yang diperlukan	Tinggi	Sedang	Rendah
Wawancara langsung	Ya	Ya	Tidak
SCAMPI	A	B	C

III. METODE PENELITIAN

Metodologi penelitian yang digunakan pada penelitian ini adalah sebagai berikut:

A. Tahap 1

Adalah kajian pustaka merupakan kegiatan untuk mempelajari literatur-literatur dan teori yang mendukung dalam melakukan penelitian ini.

B. Tahap 2

Adalah pengumpulan data dengan berdasarkan dokumen-dokumen yang telah diteliti sebelumnya.

C. Tahap 3

Analisis penilaian risiko adalah meneliti tentang implementasi CMMI pada pengembangan perangkat lunak *outsourcing*.

C.1. Memilih Representasi Continuous

Pada penelitian ini, penggunaan model CMMI menggunakan metode representasi continuous untuk memperoleh fleksibilitas dalam menentukan proses area yang akan dievaluasi. Pemilihan representasi ini dikarenakan lebih detail yaitu merepresentasikan setiap proses area dan disesuaikan dengan tingkat kemampuan organisasi [1].

C.2. Penentuan Sampel Proyek

Sampel proyek memberikan pandangan ragam cara kerja yang dilakukan agen kontraktor dalam menyelesaikan proyeknya. Dengan menentukan sampel proyek, maka akan diketahui jenis pekerjaan apakah *project development*, *project management*, atau *change request*. Pengambilan sampel proyek ini ditargetkan pada proyek yang terindikasi sering memunculkan risiko pada proses pengembangan perangkat

lunak, meskipun masih terdapat beberapa jenis proyek lain yang bisa memunculkan risiko yang sama.

C.3. Menentukan Proses Area Evaluasi

Pada siklus pengembangan perangkat lunak dengan pendekatan metode *agile* dengan menggunakan 10 proses area yang berada pada level 2 dan 3 (Mellon, 2010). Daftar proses area tersebut seperti yang terlihat pada Tabel 3.1. [1] [3] [7].

Tabel 3. 1. Domain Proses Area yang akan digunakan

Level	Kategori	Nama Proses Area	Singkatan
2	Support	Configuration Management	CM
	Project Management	Project Monitoring and Control	PMC
	Project Management	Project Planning	PP
	Support	Process and Product Quality Assurance	PPQA
	Engineering	Requirements Management	REQM
3	Engineering	Product Integration	PI
	Engineering	Requirements Development	RD
	Project Management	Risk Management	RSKM
	Engineering	Technical Solution	TS

C.4. Mengidentifikasi Praktik Spesifik yang Relevan pada Proses Area

Setelah menentukan proses area, pada tahapan ini menentukan praktik-praktik spesifik yang relevan diterapkan pada proses pengembangan perangkat lunak untuk digunakan sebagai indikator evaluasi. Kemudian indikator tersebut dikonversi kedalam kalimat tanya sebagai bahan pengambilan data kuisioner. Pada Tabel 3.2. rekapitulasi jumlah pertanyaan setiap proses area [1] [3] [7].

Tabel 3. 2. Jumlah Daftar Pertanyaan

Proses Area	Jumlah Pertanyaan
CM	2
PMC	2
PP	2
PPQA	3
REQM	2
PI	2
RD	1
RSKM	2
TS	1
VER	2
Total Pertanyaan	19

Tujuan dari proses area ini adalah untuk menjaga integritas produk dan mendukung frekuensi perubahan, frekuensi pengembangan yang didukung oleh personal, tim pengembang, ataupun even pair-programming. Pada Tabel 3.3 terdapat dua praktik spesifik yang akan digunakan sebagai proses evaluasi [1] [7].

C.5. Pengumpulan Data

Pengumpulan data dengan menggunakan kuisioner pada beberapa karyawan PT. XYZ, berdasarkan metode SCAMPI C responden yang dibutuhkan adalah berikut: 1. Karyawan dengan posisi sebagai manajer proyek dan programmer. 2. Telah memiliki nilai feedback baik. Pada Tabel 3.4. responden akan menilai setiap proses apakah diimplementasikan semua, sebagian besar, sebagian atau tidak diimplementasikan sama sekali. Kriteria penilaian *specific practice* (SP) terdapat pada tabel berikut ini [2] [7]:

Tabel 3. 3. Penilaian *Specific Practice*

Kode	Nama	Point Bobot
FI	Diimplementasikan penuh	3
LI	Diimplementasikan sebagian besar	2
PI	Diimplementasikan sebagian	1
NI	Tidak diimplementasikan	0

C.6. Identifikasi Peluang Perbaikan Proses Pengembangan Software

Pada tahapan ini dilakukan dengan mengidentifikasi dari jawaban kuisioner responden untuk menemukan celah proses yang harus diperbaiki. Celah proses diketahui dari hasil nilai

rata-rata responden tiap proses area dengan menggunakan rumus sebagai berikut:

$$\text{Nilai Pencapaian Area Proses} = \frac{\text{Jumlah Point Bobot Jawaban}}{\text{Jumlah Responden}}$$

Setelah diketahui angka nilai pencapaian tiap proses area, selanjutnya menentukan proses area tersebut tercapai atau tidak tercapai (S. U. Team, 2011). Pada Tabel 3.5 penjelasan detail dan rentang nilai yang digunakan [2] [7].

Tabel 3. 4. Penilaian *Specific Goals*

Kode	Nama	Point Bobot	Keterangan
S	Tercapai	2 - 3	Semua SP yang terkait dengan tujuan diimplementasikan penuh (FI) dan diimplementasikan sebagian besar (LI).
NS	Tidak tercapai	0 - 1.9	Agregasi kelemahan terkait dengan tujuan tidak memiliki dampak negatif yang signifikan terhadap pencapaian tujuan.

C.7. Rekomendasi Perbaikan Proses Pengembangan Perangkat Lunak

Pada tahapan ini, memberikan rekomendasi dan prioritas yang perbaikan pada pengembangan perangkat lunak outsourcing. Rekomendasi ini didasarkan pada hasil peluang perbaikan saat pengolahan data kuisioner. Rekomendasi berupa saran agar proses area tercapai, dan contoh artefak produk yang dapat mendukung pada proses area tersebut.

IV. HASIL DAN PEMBAHASAN

Hasil penelitian merupakan tahap dimana menyesuaikan dan menyusun seluruh dokumen berdasarkan literatur, penelitian terkait dengan proyek software *outsourcing*. Penyusunan dokumen literatur pada penelitian ini bersumber pada dokumen CMMI dan untuk penyusunan observasi juga didasarkan atas dokumen literatur yang menyusun pertanyaan yang akan digunakan saat observasi.

A. Identifikasi Risiko

Berdasarkan observasi terhadap fenomena yang terjadi pada proses pengembangan perangkat lunak *outsourcing* pada PT. XYZ terjadi berbagai masalah yang menghambat proses pengembangan tersebut dari sisi sumber daya manusia. Masalah yang terjadi pada proses pengembangan perangkat lunak *outsourcing* pada PT. XYZ dapat dilihat pada tabel 4.1 [5] [9].

Tabel 4. 1. Identifikasi Risiko

Kategori Risiko	Penyebab Risiko	Risiko
<i>Time/Schedule Risk</i> / Risiko Jadwal	<i>Unrealistic Estimate</i> / Estimasi yang Tidak Realistis	- Estimasi pengerjaan proyek sering tidak sesuai dengan perencanaan awal.
<i>Resource Risk</i> /Risiko Sumber Daya	<i>Human Error</i> / Kesalahan Manusia	- Kode yang belum diverifikasi maupun dites memungkinkan akan terjadinya error maupun terdapat <i>bug</i> , akan tetapi langsung di deploy ke staging bahkan di <i>live server</i> . Aplikasi seperti <i>redmine</i> dan <i>trello</i> yang berfungsi mencatat hal-hal yang berhubungan dengan proyek yang sedang dikembangkan. Seperti daftar fitur, <i>bug</i> , dokumentasi tidak diupdate apabila pekerjaan sudah selesai, progress ataupun yang lain.
<i>Resource Risk</i> /Risiko Sumber Daya	<i>Resource Unavailability</i> / Ketidakterseediaan Sumber Daya	- Berkelanjutan dengan bertambahnya waktu pengerjaan, hal ini dikarenakan dengan tidak adanya tenaga profesional untuk pengembangan sebuah perangkat lunak. Jadi tim pengembang perangkat lunak masih awam dengan bahasa pemrograman yang dipakai serta etos kerja dalam organisasi. Kurangnya tenaga profesional dibidang pengembangan perangkat lunak. Dikarenakan tenaga profesional tersebut sudah diposisikan pada proyek lain, sehingga harus menggunakan tenaga cadangan yaitu tenaga pengembang yang masih baru.

<i>Resource Risk</i> /Risiko Sumber Daya	<i>Poor Management</i> /Manajemen yang Buruk	Tidak semua manajer proyek sadar akan pentingnya sebuah tools untuk memantau perkembangan pekerjaan tim, dikarenakan kelalaian manajer proyek bertanggung jawab lebih dari satu proyek dalam waktu yang bersamaan.
<i>Resource Risk</i> /Risiko Sumber Daya	<i>Bad Negotiation</i> / Negosiasi yang Buruk	Tidak semua manajer proyek maupun konsultan proyek mampu untuk negosiasi.
<i>Resource Risk</i> /Risiko Sumber Daya	<i>Scope Creep</i> / Perubahan yang Tidak Terkontrol	Terkadang terdapat klien yang ingin mengubah alur sistem pada saat masa pengembangan.

B. Metode Kuisioner Penelitian

Berdasarkan identifikasi risiko dan pemetaan proses area, kemudian indikator tersebut dikonversi kedalam kalimat tanya sebagai bahan pengambilan data kuisioner. Dalam penelitian ini akan menggunakan cara kuisioner dengan beberapa pihak yang terkait dalam pengembangan proyek perangkat lunak *outsourcing*. Tahap ini ditujukan pada tahap pengembang atau proyek dengan penekanan pada bagian siklus hidup proyek secara keseluruhan dari daftar identifikasi risiko yang ditemukan maka daftar identifikasi risiko tersebut akan dipertimbangkan, dikaji dan disusun menjadi daftar pertanyaan-pertanyaan yang praktis sebagai acuan penelitian ini.

Tabel 4. 2. Daftar Kuisioner

No.	Level	Proses Area	Kode	Praktik Spesifik	Pertanyaan
1.	2	CM	CM-SP1.1	Mengidentifikasi konfigurasi produk	Apakah manajer proyek menerapkan standarisasi penggunaan tools, design interface, dan penjelasan proses bisnis software?
2.	2	CM	CM-SP2.1	Melacak perubahan permintaan	Apabila terjadi perubahan permintaan, apakah terdapat kesepakatan antara tim dengan pelanggan tentang estimasi waktu, prioritas dan risiko dampak perubahan?
3.	2	PMC	PMC-SP1.2	Mengawasi komitmen	Apakah manajer proyek melakukan pengawasan estimasi waktu pelaksanaan proyek, jadwal testing, dan rilis produk?
4.	2	PMC	PMC-SP1.4	Mengawasi manajemen data	Apakah manajer proyek selalu memantau perkembangan progres melalui dashboard seperti <i>atlantis jira</i> , <i>redmine</i> , atau <i>trello</i> ?
5.	2	PP	PP-SP2.3	Merencanakan data manajemen	Apakah selalu ada pembagian tugas dan rincian detail yang harus dikerjakan oleh manajer proyek kepada anggota tim development?
6.	2	PP	PP-SP3.2	Merekonsiliasi tingkat pekerjaan dan sumber daya	Apakah dalam menerima proyek selalu mempertimbangkan kompleksitas, keahlian anggota tim yang dibutuhkan, dan jadwal pengerjaan?
7.	2	PPQA	PPQA-SP1.2	Mengevaluasi hasil kerja dan layanan secara objektif	Apakah selalu ada pengujian input, proses, dan output produk perangkat lunak sebelum diberikan ke client?
8.	2	PPQA	PPQA-SP2.1	Mengkomunikasikan dan memastikan resolusi persoalan ketidakpatuhan	Apabila menemukan masalah dalam proses pengembangan apakah selalu didiskusikan dan dikomunikasikan internal antar anggota tim dengan manajer proyek?
9.	2	PPQA	PPQA-SP2.2	Membangun rekaman	Apakah selalu mencatat setiap pengaduan masalah produk perangkat lunak dari pelanggan?

10.	2	REQM	REQM-SP1.1	Memperoleh pemahaman terhadap persyaratan	Apakah tim programmer memahami detail persyaratan produk perangkat lunak diawal sebelum memulai koding?
11.	2	REQM	REQM-SP1.2	Memperoleh komitmen terhadap persyaratan	Apakah selalu konsisten antara perencanaan persyaratan awal dengan hasil akhir produk perangkat lunak?
12.	3	PI	PI-SP1.3	Membangun prosedur dan kriteria integrasi produk	Apakah tim kolaborasi selalu menerapkan prosedur, tools, dan pencatatan perubahan dalam mengintegrasikan kode program dari berbagai programmer?
13.	3	PI	PI-SP2.2	Menelola desain antarmuka	Apakah ada anggota tim yang mengelola desain antarmuka produk perangkat lunak dan memastikan sudah sesuai dengan yang diharapkan konsumen?
14.	3	RD	RD-SP3.5	Memvalidasi persyaratan pelanggan	Apakah analis sistem sudah mentransformasikan semua kebutuhan klien kedalam detail persyaratan produk?
15.	3	RSKM	RSKM-SP2.1	Mengidentifikasi risiko	Apakah risiko-risiko dalam pengembangan produk perangkat lunak sudah diketahui diawal dan terdapat strategi pencegahan?
16.	3	RSKM	RSKM-SP2.2	Mengevaluasi, mengkategorikan dan memprioritaskan risiko	Apakah terdapat pembagian kategori dan skala prioritas penanganan risiko oleh manajer proyek?
17.	3	TS	TS-SP3.2	Mengembangkan dokumentasi pendukung produk	Apakah dalam mengembangkan produk perangkat lunak dibuatkan manual dokumentasi untuk pelanggan berupa petunjuk operasional, spesifikasi detail, kebutuhan hardware, dan tampilan antarmuka produk?
18.	3	VER	VER-SP3.1	Melaksanakan verifikasi	Apakah manajer proyek melakukan verifikasi produk perangkat lunak antara model desain awal dengan hasil akhir setelah rilis?
19.	3	VER	VER-SP3.2	Menganalisis hasil verifikasi	Apakah produk perangkat lunak dilakukan pengujian input data, proses, dan hasil output sudah sesuai dengan flow proses yang diharapkan?

C. Pengambilan Data Responden

Data responden diambil dari perusahaan yang bergerak dibidang perangkat lunak *outsourcing*, yaitu PT. XYZ. Semua responden kuisioner (RK) yang dipilih tersebut telah memiliki nilai (rating) dan feedback yang baik dari klien, yaitu manager proyek dan programmer pada masing-masing sampel proyek yang diambil [2] [7].

D. Penentuan Sampel Proyek

Pada penelitian ini, sampel proyek dilakukan dengan menganalisa data proyek yang telah selesai dilakukan dan relevan untuk dievaluasi proses pengembangannya menggunakan CMMI. Pada Tabel 4.3 menampilkan 5 contoh sampel proyek yang akan digunakan sebagai dasar pengisian kuisioner [2] [3] [7].

Tabel 4. 3. Tabel Sampel Proyek

RK	Contoh Proyek
RK1	Proyek pengembangan perangkat lunak <i>Ruby on Rails</i>
RK2	Proyek lanjutan dan <i>vendor</i> lain
RK3	Proyek aplikasi mobile iOS & Android
RK4	Proyek aplikasi berbasis desktop
RK5	Maintain proyek yang sedang <i>ongoing</i>

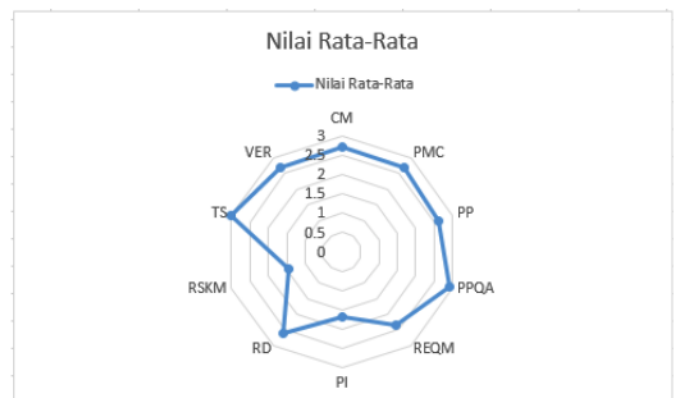
E. Perhitungan Peluang Perbaikan Perangkat Lunak

Telah didapatkan hasil perhitungan jawaban dari 10 responden terhadap kuisioner yang telah disebarkan yaitu dalam tabel 4.4.

Tabel 4. 4. Rangkuman Hasil Jawaban Responden

Level	Proses Area	Nilai Rata-Rata	Tercapai
2	CM	2.7	S
2	PMC	2.7	S
2	PP	2.6	S
2	PPQA	2.9	S
2	REQM	2.35	S
3	PI	1.7	NS
3	RD	2.6	S
3	RSKM	1.45	NS
3	TS	3	S
3	VER	2.7	S

Berdasarkan table 4.4, maka hasil nilai rata-rata dikonversikan kedalam diagram kiviati atau radar sesuai dengan gambar 4.4. Dari diagram tersebut terlihat jelas bahwa hasil dari proses area PI dan RSKM yang belum tercapai tujuannya.



Gambar 4. 1. Diagram Radar untuk Nilai Rata-Rata Hasil Kuisioner

F. Analisis Risiko

Singkatnya, risiko proyek adalah dinamis. Risiko dapat terjadi pada setiap tahap proyek. Jadi, matriks risiko proyek yang telah terdaftar risiko dan dampaknya serta probabilitas masih perlu direvisi secara berkala dan risiko yang mungkin terjadi pada saat itu dalam waktu dekat perlu dinilai dan tindakan perbaikan harus diambil untuk meminimalkan terjadinya risiko yang mungkin lebih besar, yaitu diberhentikannya proyek tersebut. Pada bagian ini, penulis hanya mencantumkan 2 peringkat teratas untuk Matrix Risiko Berdasarkan Probabilitas dan Dampak.

Tabel 4. 5. Matrix Risiko Berdasarkan Probabilitas dan Dampak

No.	Kategori Risiko	Proses Area	Penyebab Risiko	Jenis Risiko	Risiko	Probabilitas	Dampak
1.	Time/ Schedule Risk /Risiko Jadwal	PP	Unrealistic Estimate / Estimasi yang Tidak Realistis	Internal	Estimasi pengerjaan proyek sering tidak sesuai dengan perencanaan awal. Yaitu bertambahnya waktu, sehingga penyelesaian proyek akan bertambah lama.	Tinggi	Tinggi Apabila pada tahap perencanaan tidak benar-benar dihitung estimasi pengerjaannya, maka kemungkinan besar proyek akan berjalan lebih lama dari estimasi. Sehingga akan berdampak buruk dalam kelangsungan proyek tersebut.
2.	Time/ Schedule Risk /Risiko Jadwal	RSKM	Unrealistic Estimate / Estimasi yang Tidak Realistis	Internal	Estimasi pengerjaan proyek sering tidak sesuai dengan perencanaan awal. Yaitu bertambahnya waktu, sehingga penyelesaian	Tinggi	Tinggi Terdapat beberapa proyek yang belum bisa menyelesaikan <i>feature</i> yang telah disepakati diawal kontrak ketika waktu sudah mendekati

G. Pembahasan

Berdasarkan hasil dari nilai rata-rata proses area 10 responden kontraktor, diketahui bahwa masih terdapat 2 proses area yang belum tercapai antara lain *Product Integration* dan *Risk Management*. Sebagian besar proses area yang belum tercapai ada pada level 3. Dengan demikian perlu adanya strategi meningkatkan proses area tersebut.

G.1. Strategi Meningkatkan Proses Area Product Integration

Proses area ini masuk pada bagian kategori engineering. Pada proses kolaborasi pengembangan perangkat lunak setiap saat ada penambahan kode, bahkan satu data file dikerjakan lebih dari satu programmer. Penambahan fungsionalitas, penghapusan bagian kode tertentu, penataan ulang kode program (*refactoring*) selalu ada pada saat bekerja dengan tim kolaborasi. Proses area ini membantu dalam upaya assembly beberapa file kode menjadi satu produk yang siap dikirim ke client dengan beberapa strategi sebagai berikut:

a. Membuat perencanaan integrasi perangkat lunak langkah awal pada proses area ini adalah membuat rencana integrasi produk. Menentukan repositori atau server yang akan digunakan untuk integrasi, perangkat lunak yang akan digunakan, prosedur pengiriman kode masing- masing anggota tim.

b. Memastikan kompatibilitas sistem dalam mengintegrasikan kode harus dipastikan setiap perubahan yang dikirim oleh anggota tim programmer tidak mengganggu sistem yang telah ada. Dalam praktik pengembangan perangkat lunak biasanya menggunakan unit testing yang dipakai sebagai validasi bahwa kode pemrograman aman untuk diintegrasikan tanpa mengganggu fungsionalitas yang lain. Tim programmer direkomendasikan menggunakan tools unit testing. Unit testing digunakan untuk menguji penambahan, pengubahan, dan modifikasi fungsi.

Assembly dan *packing* produk pada tahap akhir, file kode dari beberapa anggota tim programmer diintegrasikan (*merger*), diverifikasi, dan divalidasi. Verifikasi dan validasi digunakan untuk memastikan kinerja, kesesuaian, dan kesiapan perangkat lunak untuk dipacking kemudian dikirim ke client. Pada

proses verifikasi dan validasi biasanya dilakukan oleh seorang *quality assurance* (QA) untuk melakukan pengujian fungsionalitas dan kualitas perangkat lunak.

G.2. Strategi Meningkatkan Proses Area Risk Management

Resiko biasanya terjadi diluar perencanaan pengembangan perangkat lunak. Kegagalan pengembangan perangkat lunak bisa terjadi karena perubahan persyaratan, anggota tim resign mendadak, dan sebagainya. Proses area ini ditujukan untuk mengidentifikasi potensi masalah sebelum terjadi dan meminimalkan dampak resiko. Strategi yang dapat dilakukan pada proses area ini antara lain:

a. Mengidentifikasi sumber risiko

pada proses pengembangan perangkat lunak sumber resiko bisa berasal dari beberapa faktor seperti:

1) Faktor waktu: proyek tidak selesai sesuai estimasi waktu pada perencanaan.

2) Faktor kualitas dan fungsionalitas: Proyek tidak sesuai dengan fitur yang telah direncanakan, atau kualitas rendah saat rilis produk.

3) Faktor biaya: Anggaran biaya proyek jauh lebih tinggi dari anggaran yang ditetapkan.

b. Melakukan penilaian risiko

Setelah ditemukan sumber risiko dan dikelompokkan kedalam faktor risiko, tahapan selanjutnya dilakukan penilaian risiko. Penilaian risiko digunakan untuk menghasilkan peringkat yang akan digunakan sebagai bahan pengambilan keputusan prioritas pencegahan resiko yang akan dilakukan. Semakin tinggi tingkat risiko yang akan berpengaruh terhadap kegagalan proyek pengembangan perangkat lunak, maka semakin diprioritaskan mitigasi risikonya.

c. Pengendalian risiko dapat meminimalisir potensi ancaman kegagalan proyek dengan cara melakukan pengawasan selama proses pengembangan perangkat lunak berlangsung. Pengawasan progres selama proses pengembangan pada dapat dilakukan dengan menggunakan tools seperti redmine, atau trello, seperti pada saat ini sehingga selalu ada komunikasi antara project manager dengan anggota tim.

Berdasarkan penjelasan diatas, maka tabel 4.6. berikut ini merepresentasikan jenis risiko pada proses area RSKM serta cara pengendalian risikonya:

Tabel 4. 6. Risiko pada Proses Area RSKM Serta Cara Pengendalian risikonya

No	Kategori Risiko	Risiko
1.	<i>Time / Schedule Risk / Risiko Jadwal</i>	Estimasi pengerjaan proyek sering tidak sesuai dengan perencanaan awal. Yaitu bertambahnya waktu, sehingga penyelesaian proyek akan bertambah lama.
2.	<i>Resource Risk / Risiko Sumber Daya</i>	Berkelanjutan dengan bertambahnya waktu pengerjaan, hal ini dikarenakan dengan tidak adanya tenaga professional untuk pengembangan sebuah perangkat lunak. Jadi tim pengembang perangkat lunak masih awam dengan Bahasa pemrograman yang dipakai serta etos kerja dalam organisasi.
3.	<i>Resource Risk / Risiko Sumber Daya</i>	Keterbatasan akan pengetahuan atau kompleksitas sebuah produk yang akan dibangun sering kali menjadi salah satu penyebab. Sehingga masih terjadi miskomunikasi antara tim dan klien.

V.KESIMPULAN DAN SARAN

A.Kesimpulan

Berdasarkan hasil analisis yang telah dilakukan dan mengacu kepada tujuan penyusunan tesis ini, maka dapat ditarik kesimpulan sebagai berikut:

1. Hasil yang didapatkan dari penelitian ini merupakan rencana strategis yang digunakan PT. XYZ untuk meminimalkan risiko dalam proses pengembangan perangkat lunak.
2. Rencana dan saran yang dihasilkan dapat membantu memberikan solusi alternative untuk pengambilan keputusan mengenai proses pengembangan perangkat lunak dengan risiko yang telah dianalisis.

B.Saran

Berdasarkan hasil analisis yang telah dilakukan dan mengacu kepada tujuan penyusunan tesis ini, maka terdapat saran yang bisa disampaikan untuk lebih memperbaiki hasil yang didapatkan. Untuk penelitian selanjutnya dari hasil penelitian yang sudah ada bisa dikembangkan sampai tahap implementasi dengan cara menambahkan penentuan area

proses *Validation* pada level 3 CMMI, mengidentifikasi kembali jenis risiko, menganalisis dan memprioritaskan kembali jenis-jenis risiko.

VI.

DAFTAR PUSTAKA

- [1] C. M. University, "CMMI® for Development, Version 1.3," CMMI Product Team, Hanscom, 2010.
- [2] C. M. University, Standard CMMI® Appraisal Method for Process Improvement (SCAMPISM) A, Version 1.3: Method Definition Document, Hanscom: SCAMPI Upgrade Team, 2011.
- [3] Z. U. ISLAM and X. ZHOU, "Software Process Improvement Framework for Software Outsourcing Based On CMMI," *University of Gothenburg*, August 2011.
- [4] S. Atluri and P. K. Nalli, "Software Development in an Outsourcing Environment," *Ume'a University*, June 2006.
- [5] C.-J. Jager, S. Vos, M. Borgers, F. Harmsen, S. Brinkkemper and L. v. d. Wijngaert, "Controlling risk prior to offshore application development," *Utrecht University*.
- [6] A. Yalaho, Managing Offshore Outsourcing of Software Development Using the ICT-supported Unified Process Model, University of Jyväskylä, 2009.
- [7] W. Widodo, "EVALUASI PROSES PENGEMBANGAN PERANGKAT LUNAK PADA VIRTUAL TEAM DEVELOPMENT MENGGUNAKAN CMMI Versi 1.3," *JURNAL INFORMATIKA*, vol. 10, January 2016.
- [8] J. Miler, "A Method of Software Project Risk Identification and Analysis," 2005.
- [9] A. Ahmed, Software Project Management – A Process Driven Approach, Parkway: Taylor & Francis Group, LLC, 2012.
- [10] C. M. University, "Software Engineering Institute," U.S. Department of Defense (DoD), 2016. [Online]. Available: <http://www.sei.cmu.edu/>.