ISSN 2085-4579

# Return on Invesment (ROI) of Software Product: A Systematic Literature Review

Rizqa Raaiqa Bintana[1], Putri Aisyiyah Rakhma Devi[2], Umi Laili Yuhana[3]
[1,2,3] Program Studi Informatika, Fakultas Teknologi Informasi, Surabaya, Indonesia
[1]rizqa.raaiqa.bintana@gmail.com, [2]putriaisyiyah@gmail.com, [3]yuhana@cs.its.ac.id

Diterima 08 Mei 2015
Disetujui 01 Juni 2015

***Abstract*—The quality of the software can be measured by its return on investment. Factors which may affect the return on investment (ROI) is the tangible factors (such as the cost) dan intangible factors (such as the impact of software to the users or stakeholder). The factor of the software itself are assessed through reviewing, testing, process audit, and performance of software. This paper discusses the consideration of return on investment (ROI) assessment criteria derived from the software and its users. These criteria indicate that the approach may support a rational consideration of all relevant criteria when evaluating software, and shows examples of actual return on investment models. Conducted an analysis of the assessment criteria that affect the return on investment if these criteria have a disproportionate effort that resulted in a return on investment of a software decreased.**

***Index Terms*—Assessment criteria, Quality assurance, Return on Investment, Software product**

## I. PENDAHULUAN

Perangkat lunak adalah istilah umum untuk data yang diformat, dan disimpan secara digital, termasuk program komputer, dokumentasinya, dan berbagai informasi yang bisa dibaca, dan ditulis oleh komputer. Pembuatan perangkat lunak itu sendiri memerlukan bahasa pemrograman yang ditulis oleh *programmer* untuk selanjutnya dikompilasi dengan aplikasi *compiler* sehingga menjadi kode yang bisa dikenali oleh mesin *hardware*. Produk perangkat lunak sendiri adalah sistem dari suatu rekayasa perangkat lunak yang terdiri dari *source code*, dokumentasi yang menjelaskan prosedur penyiapan dan penggunaan perangkat lunak tersebut.

Produk perangkat lunak setiap tahunnya akan terus mengalami perkembangan. Oleh karena itu, setiap produk perangkat lunak harus memiliki sebuah kualitas perangkat lunak. Namun dalam kenyataan yang ada di lapangan, proses penjaminan kualitas sebuah produk perangkat lunak bukanlah hal yang biasa dan mudah.

Perangkat lunak (sistem) dapat membantu suatu pekerjaan yang kompleks menjadi lebih ringan untuk diselesaikan. Seperti pekerjaan memasukkan, me-*retrieve*, mencari, mengirim data, dan berkomunikasi dengan pengguna atau *stakeholder* yang terlibat dalam pekerjaan. Faktor yang bersifat berwujud dan tidak berwujud harus masuk ke dalam evaluasi perangkat lunak sebagai bahan pertimbangan *return on investment* (ROI) untuk suatu perangkat lunak.

Peningkatan hasil dalam hal penghematan biaya dapat menjadi tujuan mengapa digunakannya suatu perangkat lunak dalam membantu pekerjaan [1]. Peningkatan hasil bukan dinilai berdasarkan biaya saja (faktor yang bersifat berwujud), tetapi juga dapat dinilai dari kinerja perangkat lunak (faktor yang bersifat tidak berwujud) yang dinilai oleh pengguna atau *stakeholder* lainnya. Untuk menentukan *return on investment* dari perangkat lunak dibutuhkan pertimbangan kuantitatif seperti peningkatan produktivitas, peningkatan *capture* biaya dan dokumentasi, dan pertimbangan kualitatif seperti manfaat penggunaan perangkat

lunak serta perannya dalam membantu pengambilan keputusan [1].

Tujuan dari tulisan ini adalah untuk mengetahui kriteria yang menjadi penilaian dalam mengevaluasi *return on investment* suatu perangkat lunak berdasarkan dari pembahasan kumpulan literatur terkait (*systematic literature review*). Kriteria tersebut berkaitan dengan faktor berwujud maupun tidak berwujud pada perangkat lunak. Pada bagian berikutnya dalam tulisan ini berisi tinjauan literatur yang terkait. Dilanjutkan dengan bagian penjelasan metode yang dilakukan dalam penelitian, penjelasan hasil dan pembahasan. Dan bagian terakhir diberikan kesimpulan.

## II. LITERATUR TERKAIT

*Return on investment* (ROI) adalah metrik yang sangat populer karena sifatnya yang fleksibilitas dan kesederhanaannya. Artinya, jika suatu perangkat lunak tidak memiliki *return on investment* positif, atau jika ada perangkat lunak lainnya yang memiliki *return on investment* yang lebih tinggi, maka investasi terhadap perangkat lunak tersebut tidak harus dilakukan, yang berarti tidak perlu untuk melanjutkan penggunaan dan pengembangan perangkat lunak tersebut dalam membantu pekerjaan. Hal yang bisa menjadi penilaian *return on investment* diantaranya total waktu, biaya, dan penilaian manfaat perangkat lunak oleh pengguna atau *stakeholder. Return on investment* adalah ukuran kinerja yang digunakan untuk mengevaluasi efisiensi investasi atau untuk membandingkan efisiensi dari sejumlah investasi yang berbeda terhadap suatu perangkat lunak. *Return on investment* diperoleh dengan cara membagi *return of an investment* dengan *cost of the investment*, dan hasilnya dinyatakan sebagai persentase atau rasio [2].

*Return on investment* telah diusulkan sebagai pengukuran yang efektif untuk mengevaluasi intervensi pelatihan. Namun demikian, ketika menerapkan *return on investment* dalam lingkungan produksi perangkat lunak, praktisi tidak mengambil pertimbangan (perhitungan) dampak-dampak dari variasi dalam proses produksi. Dengan perhitungan-perhitungan *return on investment*, biasanya permasalahan terjadi dimana faktor biaya dan sistem akuntansi suatu organisasi diketahui telah men-*tracking* perhitungan tersebut. Sebaliknya, kepentingan untuk mengidentifikasi lebih sulit dan biasanya perlu ada persetujuan diantara *stakeholder* yang terlibat dalam menganalisis hasil. Dalam intervensi pelatihan, peningkatan manfaat perangkat lunak harus datang dalam bentuk peningkatan kinerja tenaga kerja [3].

Jika *return on investment* telah diperkirakan sebelum penerapkan kualitas, hal ini akan membantu untuk memutuskan apakah akan melakukan suatu teknik tertentu atau tidak dan memilih diantara persaingan teknik kualitas perangkat lunak. *Return on investment* juga cocok untuk menginformasikan manajemen tentang potensi penghematan dari penerapan teknik kualitas perangkat lunak tertentu, dan memutuskan apakah akan meningkatkan, mengganti atau mengembangkan kualitas suatu perangkat lunak [4].

## III. METODE PENELITIAN

Untuk mencapai tujuan dari penelitian yang dilakukan, digunakan metode berdasarkan pendekatan Kitchenham dan Charters yaitu menggunakan metode *systematic literature review*. *Systematic literature review* adalah cara untuk mengidentifikasi, mengevaluasi dan menafsirkan semua penelitian yang tersedia dan yang relevan dengan pertanyaan tertentu pada suatu penelitian, atau bidang topik, atau fenomena yang menarik. Tahapan-tahapan dalam *systematic literature review* adalah menentukan *research question*, proses pencarian, *study selection*, *quality assessment*, dan proses ekstraksi data [5].

Metode penelitian yang dilakukan dalam penulisan yang berlandaskan *systematic literature review* (SLR) ini dimulai dengan perencanaan. Proses perencanaan merupakan proses untuk merumuskan batasan dan langkah kerja (metode) yang akan dilakukan. Perencanaan ini dilanjutkan dengan proses *review* yang berhubungan dengan

pencarian literatur, pemilihan literature tentang kualitas perangkat lunak, serta pencarian literatur pendukung. Metode terakhir pada penelitian ini adalah dokumentasi yang dilakukan untuk menuliskan hasil temuan dari proses sebelumnya. Tabel 1 menunjukkan tahapan yang dilakukan dalam penulisan yang berlandaskan SLR ini.

Tabel 1. Metode penelitian

| Perencanaan | Proses *Review* | Dokumentasi |
|---|---|---|
| Merumuskan batasan melalui *research question* | Mengidentifikasi hasil pencarian yang relevan | Menulis laporan |
| Merumuskan metode yang akan dilakukan | Memilih topik utama penelitian yang sesuai dengan masing-masing literatur | |
| Memvalidasi hasil perumusan metode | Mengukur tingkat kualitas hasil penelitian dari literatur yang telah ditemukan | Memvalidasi laporan |
| | Melakukan pengambilan data | |

*A. Kriteria Inklusi dan Eksklusi*

Tidak semua literatur tentang kualitas perangkat lunak dijadikan sumber literatur yang akan dianalisa. Ada beberapa kriteria mengenai literatur-literatur tersebut. Adapun kriteria literatur tersebut adalah sebagai berikut: literatur yang berupa jurnal, *conference* dan *workshop*; batas waktu publikasi dari tahun 2007 sampai tahun 2015; literatur yang membahas kualitas produk perangkat lunak; dan literatur yang membahas *return on investment*.

Selain itu, terdapat kriteria tertentu untuk literatur yang tidak digunakan sebagai sumber literatur. Adapun kriteria tersebut adalah sebagai berikut: literatur yang berbentuk editorial dan *resume*; literatur yang dipublikasi sebelum tahun 2007; literatur yang tidak berhubungan dengan kualitas produk perangkat lunak; dan literatur yang tidak berhubungan dengan *return on investment*.

*B. Sumber Data dan Strategi Pencarian*

Proses pengumpulan literatur hanya dilakukan pada pencarian dari media *online* yaitu sciencedirect, ieee, dan google scholar karena penulis menganggap sumber literatur dari ketiga media *online* tersebut terpercaya. Kata kunci yang digunakan dalam pencarian literatur pada media *online* tersebut yaitu sebagai berikut: *software quality*; *software quality assurance*; *software product*; *return on investment* (ROI); *software quality assurance AND return on investment*; dan *software quality assurance OR software product OR return on investment*.

Selain itu, tipe literatur yang dicari adalah berupa jurnal, *conference* maupun *workshop*. Dari hasil pencarian yang telah dilakukan, maka diperoleh literatur dengan rincian seperti yang ditunjukkan pada Tabel 2. Total literatur yang ditemukan dari hasil pencarian adalah 5874 literatur, dimana total literatur ini masih belum di-*filter* berdasarkan kriteria inklusi dan eksklusi yang telah ditetapkan.

Literatur-literatur tersebut kemudian di-*filter* berdasarkan kesesuaian topiknya. Jika topik yang dibahas dalam literatur sesuai, maka akan dianalisa lebih lanjut mengenai metode, hasil penelitian, dan pengembangan penelitiannya. Adapun langkah-langkah penyaringan literatur yang sesuai dengan topik pembahasan diilustrasikan pada Gambar 1.

Tabel 2. Hasil pencarian literatur pada media *online*

| | | | | |
|---|---|---|---|---|
| 1 | **ScienceDirect** | | | |
| | *Software Quality Assurance* | 2007- 2014 | Journal | 1187 |
| | *Software Product* | 2007- 2014 | Journal | 673 |
| | *Return on Investment Software Product* | 2007- 2014 | Journal | 178 |
| | **Total** | | | **2038** |
| | | | | |
| 2 | **leeexplore** | | | |
| | *Software Quality Assurance* | 2007- 2014 | Journal | 1263 |
| | *Software Product* | 2007- 2014 | Journal | 592 |
| | *Return on Investment Software Product* | 2007- 2014 | Journal | 236 |
| | **Total** | | | **2091** |
| | | | | |

| | | | | |
|---|---|---|---|---|
| 3 | **Google Scholar** | | | |
| | *Software Quality Assurance* | 2007- 2014 | Journal | 1089 |
| | *Software Product* | 2007- 2014 | Journal | 502 |
| | *Return on Investment Software Product* | 2007- 2014 | Journal | 154 |
| | **Total** | | | **1745** |
| | | | | |
| | **Total Keseluruhan** | | | **5874** |

Gambar 1. Tahapan evaluasi literatur (*n* adalah jumlah literatur yang dievaluasi pada proses tahapan)

*C. Keputusan Inklusi dan Eksklusi*

Total literatur yang terkumpul pada proses pencarian awal adalah 5874 literatur. Setelah itu, literatur tersebut diidentifikasi berdasarkan judulnya. Jika judulnya sesuai dengan topik penjamin kualitas perangkat lunak, maka literatur tersebut akan dipisahkan untuk dilakukan identifikasi pada bagian abstraknya. Total literatur yang telah diidentifikasi berdasarkan judul adalah sebanyak 2937 literatur. Setelah itu, dilakukan proses penyaringan literatur dengan membaca abstrak dari literatur-literatur tersebut untuk mengidentifikasi kesesuaian literatur dengan topik pembahasan dalam penelitian ini. Pada hasil identifikasi, terkumpul 85 literatur. Setelah dilakukan proses identifikasi terhadap keseluruhan isi literatur (isi, metode, dan hasil penelitian dalam masing-masing literatur), maka diperoleh hanya 6 literatur yang dinilai sesuai, yang benar-benar membahas tentang kualitas produk perangkat lunak dengan penilaian *Return on Investment* (ROI)-nya.

## IV. HASIL DAN PEMBAHASAN

Terdapat 3 *research question* (RQ) yang menjadi acuan pembahasan dalam studi *systematic literature review* (SLR) ini.

*A. Kriteria Penilaian Return on Investment*

RQ1. Apa saja kriteria dalam penilaian *Return on Investment* pada produk perangkat lunak?

1. *Review*

*Review* dianggap sebagai kegiatan penjaminan kualitas perangkat lunak yang paling efektif dan merupakan praktek industri terbaik menurut banyak temuan penelitian. *Review* mempunyai manfaat yang maksimal dimana *review* dapat mencegah kerusakan atau kecacatan (*defects*) suatu perangkat lunak dan menghemat waktu dalam pekerjaan kedepannya. Kegiatan yang dapat dilakukan melalui *review* diantaranya menemukan cacat spesifikasi, bagian yang hilang, dan pengembang *blind spot* lebih mudah ditemukan melalui *review* [6].

2. *Testing*

*Testing* berfungsi sebagai penanganan cara kerja sistem yang sebenarnya, dimana *review* tidak mampu melakukannya melalui *imaging*. Meskipun kedua kegiatan penjaminan kualitas perangkat lunak tersebut saling tumpang tindih (*overlapping*) dalam menemukan beberapa jenis kerusakan (*defects*) yang umum, misalnya kesalahan pemrograman dan cacat logika, keduanya cukup saling melengkapi dalam menemukan berbagai jenis kerusakan (kecacatan). Cacat dalam hal perkiraan numerik dan dinamika program sulit untuk ditemukan melalui *review* karena *reviewer* tidak bisa menjelaskan seberapa cepat dan handal sistem, seberapa *user-friendly* suatu sistem dengan hanya me-*review* dan menyimpulkan dari kebutuhan abstrak, rancangan dokumen atau *code* [6].

3. Proses Audit

Proses audit dapat membantu tim pengembang perangkat lunak untuk bekerja pada arahan yang benar, menggunakan dan menyesuaikan metode yang paling efektif untuk konteks yang spesifik pada sebuah proyek perangkat lunak, meningkatkan keefektifan proses yang pada akhirnya akan meningkatkan kualitas perangkat lunak [6].

4. Penilaian kinerja perangkat lunak oleh pengguna

Sebelum memulai proyek pembuatan ataupun pengembangan perangkat lunak, konsep *return on investment* digunakan dalam mengembangkan

model simulasi *spreadsheet* untuk memperkirakan kelayakan proyek perangkat lunak, dimana seluruhnya didasarkan pada modul manajemen persediaan, yang merupakan aplikasi bisnis. Perkiraan tersebut memprediksi penghematan yang cukup besar dari proyek tersebut yang dapat dilakukan, yang hanya didasarkan pada parameter yang jelas [1].

Kriteria ineksplisit yang dapat dimasukkan ke dalam model ini meliputi ketersediaan data untuk pengambilan dan analisis, rata-rata *delay* data yang tersedia hingga dapat mengambil keputusan, tingkat kesalahan dari entri data yang berkurang, *delay* yang mampu dikurangi dan biaya pengoreksian kesalahan, pengurangan jumlah *callback* untuk memperbaiki permintaan resmi yang disebabkan *code* produk yang hilang atau tidak terbaca atau terdeskripsi, penghapusan untuk perintah yang duplikat oleh pengguna, waktu yang dibutuhkan untuk rutinitas pemrosesan berkurang.

Kuantitatif *return on investment* dapat digunakan untuk mengevaluasi suatu perangkat lunak. Penilaian dari pengguna terhadap perangkat lunak atau aplikasi jauh lebih mungkin untuk dapat memberikan kontribusi yang besar. Evaluasi untuk setiap perangkat lunak hampir sama, hanya tergantung pada ukuran kontribusi dari karakteristik kuantitatif ataupun kualitatif perangkat lunak yang dipertimbangkan.

### *B. Mengidentifikasi Alokasi Sebaran Kegiatan Penjaminan Kualitas*

RQ2. Bagaimana mengidentifikasi kegiatan penjaminan kualitas yang mana yang mungkin tidak seimbang sementara yang lain mungkin *over*?

Beberapa cara yang dapat dilakukan untuk mengidentifikasi alokasi sebaran kegiatan penjaminan kualitas, yaitu melalui *regression diagnostic model*, menghitung *cost of finding and removing defects*, dan menganalisa sebaran jenis kerusakan (kecacatan) perangkat lunak yang ditemukan melalui *testing*.

Dalam *regression diagnostic model*, pengujian adalah kegiatan yang paling padat dalam kegiatan penjaminan kualitas dan *effort* memposisikan *review* dan proses audit menjadi tidak seimbang sebaran kegiatannya, hal ini mungkin memberikan manajer kualitas beberapa proses perbaikan implikasi yang menempatkan lebih banyak *effort* pada *review* dan proses audit untuk menghemat waktu pengujian, dan akhirnya memperpendek jadwal pengujian dan menyimpan seluruh biaya pengujian. Selain itu, karena *return on investment* yang menambahkan lebih banyak *effort* pada proses audit lebih besar daripada yang menambahkan lebih banyak *effort* pada *review*, hal tersebut memberikan beberapa implikasi bahwa untuk organisasi perangkat lunak, meningkatkan kegiatan proses audit mungkin bisa menjadikan lebih hemat biaya.

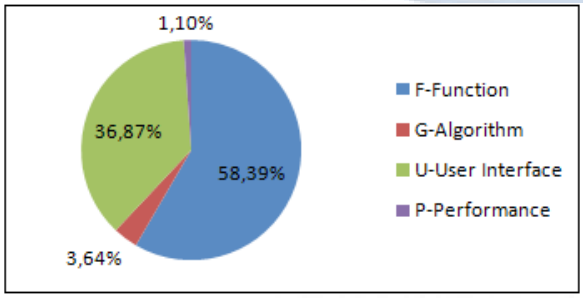
Untuk membandingkan *cost of finding and removing defects* pada setiap fase, dapat digunakan Persamaan 1. [6] melakukan perbandingan dengan menghitung *cost of finding and removing defects* terlebih dulu dengan data yang digunakan untuk perhitungan seperti yang ditunjukkan pada tabel oleh [6] beserta hasilnya. Mereka melakukan perbandingan terhadap *requirement*, desain, *cost of finding and removing defects* pada tahap pengujian, dan *cost of finding and removing defects Non-Conformances*.

*Cost of finding and removing defects = Hours spent on reviewing or testing in the phase / numbers of defects found and removed in the phase* (1)

Bisa dilihat [6] bahwa waktu yang dihabiskan untuk menemukan dan menghapus kecacatan (*defects*) pada perangkat lunak meningkat dari fase *requirement* hingga pengujian, ini artinya bahwa biaya untuk menemukan dan menghapus kecacatan meningkat ketika proyek perangkat lunak berlangsung. Hal ini tidak hanya ditemukan pada organisasi atau perusahaan di cina tetapi juga IBM, TRW, maupun GTE [6]. Waktu yang dihabiskan untuk menemukan dan menghapus kecacatan *Non-Conformances* melalui proses audit juga jauh lebih kecil daripada waktu yang dihabiskan untuk menemukan dan menghapus kecacatan melalui pengujian. Analisa ini mampu mendukung proses audit dan *reviewing* yang lebih

efektif dan efisien untuk mendeteksi kecacatan perangkat lunak daripada pengujian. Selain itu, penambahan *effort* yang lebih pada awal *review*, temuan kecacatan dan proses audit akan menjadikan lebih hemat biaya.

[6] juga mengumpulkan kecacatan (*defects*) yang ditemukan dalam pengujian pada suatu perangkat lunak dengan tiga versi (v3.0, v3.1 dan v3.1.1) di dalam penelitiannya. Mereka mengklasifikasikan kecacatan yang ditemukan sesuai dengan jenisnya, yaitu *F-Function*, *G-Algorithm*, *U-User Interface*, dan *P-Performance* seperti yang ditunjukkan pada Gambar 2. Ditemukan bahwa kecacatan yang berkaitan dengan *function* berpengaruh lebih besar dan hal ini mungkin menyiratkan bahwa perlu upaya untuk menyeimbangkan *review* di awal terutama menyeimbangkan *code walkthrough* sebelum pengujian. Kecacatan yang berkaitan dengan *user interface* dan *performance* sulit untuk dideteksi melalui *review* tapi bisa lebih mudah dideteksi melalui pengujian.




Gambar 2. Sebaran jenis kerusakan (kecacatan) perangkat lunak yang ditemukan melalui *testing* [6]

*C. Menyeimbangkan Alokasi Kriteria-kriteria yang ada dalam Penilaian Return on Investment*

RQ3. Bagaimana cara menyeimbangkan kriteria-kriteria yang berkaitan dengan penilaian suatu *Return on Investment*?

Tujuan: Meningkatkan *return on investment* pada kegiatan penjaminan kualitas.

Pertanyaan-1 (Q1): Apa dasar alokasi upaya (*effort*) pada kegiatan penjaminan kualitas perangkat lunak dibawah konteks organisasi tertentu? dengan Metrik-1 (M1): Deskripsi statistik: *mean*, median, standar deviasi pada persentase sebaran upaya (*effort*)

Pertanyaan-2 (Q2): Bagaimana meningkatkan *review* jika tidak seimbang? dengan Metrik-2 (M2): *Filter Effectiveness*

Pertanyaan-3 (Q3): Bagaimana meningkatkan proses audit jika tidak seimbang? dengan Metrik-3 (M3): Sebaran NCs (*Not Compatibles*)

1. Pertanyaan-1 dan Metrik-1

Jika ingin menjaga *maturity* suatu proses perangkat lunak sehingga mencapai tujuan kualitas suatu perangkat lunak, *stakeholder* harus berupaya dalam kegiatan penjaminan kualitas. Deskripsi statistik untuk sebaran upaya (*effort*) penjaminan kualitas memberikan dasar persentase upaya penjaminan kualitas yang diperlukan untuk tujuan kualitas yang dibutuhkan. [6] telah melakukan analisa terhadap deskripsi statistik (*mean*, median, dan standar deviasi) pada suatu perangkat lunak dengan lima versi dari proyek pengembangannya (versi 2.5, 2.6, 3.0, 3.1, dan 3.1.1). Analisa dilakukan untuk *effort* pada *review*, *testing*, proses audit dan *quality assurance* di masing-masing versi perangkat lunak tersebut. Dan hasilnya, dibutuhkan *effort* sekitar 48,21% pada kegiatan penjaminan kualitas (*quality assurance*). Dimana *effort* pada kegiatan penjaminan kualitas tersebut terbagi untuk tiga *effort* lainnya, yaitu *effort* untuk *review* 4,30%, *effort* untuk pengujian (*testing*) 34,47%, dan *effort* untuk proses audit 1,70%.

Nilai standar deviasi terkecil ada pada *effort* untuk proses audit, dengan nilai mendekati nol, 0,63. Nilai ini menunjukkan bahwa variasi sebaran *effort* yang dilakukan untuk proses audit di masing-masing versi perangkat lunak tersebut adalah mendekati sama, yang dibuktikan dengan nilai *effort* 1,78%; 2,51%; 1,49%; 1,00%; dan NA. Sedangkan nilai standar deviasi pada *effort* untuk *review* dan *testing* adalah 2,66 dan 13,33, yang berarti bahwa variasi sebaran *effort* yang dilakukan untuk proses *review* dan *testing* di masing-masing versi perangkat lunak adalah sangat bervariasi seperti yang telah ditunjukkan oleh [6].

2. Pertanyaan-2 dan Metrik-2

Pengembang perangkat lunak tidak akan menempatkan *effort* yang sama pada *review* kebutuhan, *review* rancangan dan *code*. Pengembang akan menempatkan upaya (*effort*) ekstra pada bagian terlemah yang akan lebih hemat biaya. Hasil analisis akan membantu manajer kualitas untuk mengidentifikasi *review* yang mana yang paling lemah menurut konteks suatu organisasi.

*Review* berfungsi sebagai *filter*, menghapus maupun mengurangi persentase kerusakan (kecacatan) perangkat lunak dan melanjutkan pengembangan perangkat lunak ke tahap berikutnya. Definisi *Filter Effectiveness* (FE) dari setiap tahap sebagai metrik dalam peningkatan *review* adalah [6]:

*Filter Effectiveness = defects found in this phase / defects introduced in this phase* ...(2)

atau bisa ditulis menjadi,

*Filter Effectiveness = defects found in this phase / (defects found in this phase + defects introduced in this phase and found by testing)* (3)

Pada tabel yang ditunjukkan [6], disajikan nilai *filter effectiveness* dari *reviewing requirement*, *reviewing design*, dan *reviewing code* untuk setiap tahap pengembangan perangkat lunak. Rata-rata nilai *filter effectiveness* dari *reviewing requirement* untuk keseluruhan versi perangkat lunak adalah 0,69. Sedangkan *filter effectiveness* dari *reviewing design* yaitu 0,42 dan *filter effectiveness* dari *reviewing code* adalah 0,10.

Berdasarkan hasil yang ditunjukkan [6], disimpulkan bahwa *filter effectiveness* menurun ketika proyek berlangsung, terutama di tahap *coding*, dengan nilai *filter effectiveness* 0,01 untuk perangkat lunak versi 3.0. Tim pengembang perangkat lunak harus lebih berupaya pada *code walkthrough* dan unit *testing* sebelum mereka merilis *code* ke tim penguji.

3. Pertanyaan-3 dan Metrik-3

Akan lebih bermanfaat jika tim pengembang perangkat lunak juga memberikan upaya (*effort*) yang lebih pada proses audit yang mungkin lebih efektif daripada *review*. Auditor independen memeriksa proses-proses dengan seperangkat *checklist* untuk area proses yang mengacu pada CMMI v1.2. NCs (*not compatibles*) yang ditemukan untuk area proses ditunjukkan oleh [6], yaitu mencakup *project planning*, *project monitoring and control*, *configuration management*, *software testing*, *review*, *project development lifecycle standard process*, *requirements development*, *organizational training*, dan *requirements management*.

Ditemukan bahwa *Project Monitoring and Control* (PMC) dan *Project Planning* (PP) berpengaruh besar pada NCs. [6] menemukan bahwa masalah utama bagi PMC adalah bahwa tim pengembang sering mengabaikan menyerahkan laporan proyek perangkat lunak berkala mereka. *Item* utama laporan tersebut meliputi rasio penyelesaian tugas, upaya terencana dan aktual, serta masalah atau risiko yang dihadapi tim pengembang. Laporan tersebut digunakan oleh manajer proyek untuk memantau kemajuan tugas tim pengembang, mengidentifikasi risiko dan isu-isu yang ada.

Dengan meneliti NCs yang terkait dengan PP [6], masalah yang paling umum adalah bahwa upaya yang terencana dan aktual memperlihatkan selisih yang besar. *Overestimating* dapat mengakibatkan keterlambatan proyek, mempersingkat waktu untuk pengujian (*testing*) dan dapat menyebabkan produk perangkat lunak berkualitas rendah, sementara *underestimates* mungkin dapat menyebabkan sumber daya terbuang sia-sia. Perbaikan lebih lanjut untuk masalah tersebut adalah dengan membuat pedoman untuk estimasi usaha dalam pembuatan maupun pengembangan proyek perangkat lunak, mengadopsi metode atau *tools* estimasi upaya yang lebih sistematis dan profesional, seperti penggunaan *tool* COCOMO (*Constructive Cost Model*).

## V. SIMPULAN

Untuk mengidentifikasi pola sebaran upaya (*effort*) yang paling efektif pada kegiatan penjaminan kualitas yang pada akhirnya

meningkatkan efektivitas seluruh proses, harus dilakukan serangkaian analisis empiris terhadap data proyek dari suatu organisasi perangkat lunak. Melakukan kegiatan *review* di awal lebih efektif daripada pengujian. Proses audit lebih efektif untuk mencegah adanya cacat (*defect*). Penilaian kinerja perangkat lunak oleh pengguna ataupun *stakeholder* lainnya juga mampu meningkatkan kualitas perangkat lunak melalui pengembangannya. Keempat jenis kegiatan penjaminan kualitas tersebut saling melengkapi satu sama lain untuk mencapai tujuan kualitas suatu produk perangkat lunak. Sehingga dapat memaksimalkan ROI (*return on investment*) perangkat lunak tersebut.

Kegiatan penjaminan kualitas perangkat lunak tersebut akan membantu manajer kualitas untuk mengidentifikasi jenis kegiatan penjaminan kualitas mana yang tidak seimbang sementara yang lain mungkin *over*, bagaimana mengidentifikasi dan memperbaiki bagian terlemah dari kegiatan penjaminan kualitas, bagaimana menyeimbangkan alokasi upaya (*effort*) dan perencanaan untuk proyek-proyek masa depan yang pada akhirnya mampu meningkatkan seluruh efektivitas proses dalam suatu organisasi.

## Daftar Pustaka

[1] N. Archer, "Is Return on Investment a Valid Criterion for Investing in Mobile Healthcare?," pp. 49–49, 2007.

[2] L. Yi, M. Zuo, and Z. Wang, "A Model for Analyzing and Evaluating the Return on Investment in e-Learning," in *Seventh IEEE International Conference on Advanced Learning Technologies, 2007. ICALT 2007*, pp. 79–81, 2007.

[3] S. Matalonga and T. S. Feliu, "Calculating return on investment of training using process variation," *IET Softw.*, vol. 6, no. 2, p. 140, 2012.

[4] M. Felderer and A. Beer, "Estimating the Return on Investment of Defect Taxonomy Supported System Testing in Industrial Projects," in *2012 38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 426–430, 2012.

[5] B. Kitchenham and S. Charters, *Guidelines for performing Systematic Literature Reviews in Software Engineering*. 2007.

[6] Q. Li, F. Shu, B. Boehm, and Q. Wang, "Improving the ROI of Software Quality Assurance Activities: An Empirical Study," in *New Modeling Concepts for Today's Software Processes*, J. Münch, Y. Yang, and W. Schäfer, Eds. Springer Berlin Heidelberg, pp. 357–368, 2010.