IMPLEMENTASI SCRUM PADA PENGEMBANGAN SOFTWARE TERDISTRIBUSI

Rezania Agramanisti Azdy[1)], Azhari SN[2)]
[1)]Pascasarjana Ilmu Komputer, Universitas Gadjah Mada
Program Studi Monodisiplin S2/S3 Ilmu Komputer UGM, Gedung SIC Lt.3
FMIPA UGM Sekip Utara Bulak Sumur-Yogyakarta 55281 Telp/Fax : (0274) 5551333
[2)]Ilmu Komputer, Universitas Gadjah
Program Studi Monodisiplin S2/S3 Ilmu Komputer UGM, Gedung SIC Lt.3
FMIPA UGM Sekip Utara Bulak Sumur-Yogyakarta 55281 Telp/Fax : (0274) 5551333
e-mail: rezania.azdy@gmail.com, arisn.softcomp@gmail.com

Abstrak
Seiring berkembangnya jaman pengembangan software secara terdistribusi merupakan metode pendekatan pengembangan software yang makin sering digunakan. Dengan mengatasi kekurangan-kekurangan yang mungkin muncul dalam mengembangkan secara terdistribusi, pendekatan ini terbukti mampu memberikan hasil yang lebih baik dari pengembangan software yang dilakukan secara terpusat. Scrum merupakan salah satu varian dari metode agile yang iteratif dan incremental. Satu iterasi pada Scrum disebut dengan Sprint, dimana untuk setiap akhir dari Sprint produk yang sesuai dengan definisi "Done" harus diraih. Kedua pendekatan pengembangan software ini sama-sama menekankan pada pentingnya komunikasi dan kolaborasi antar berbagai pihak. Jurnal ini akan membahas bagaimana mengimplementasikan Scrum pada pengembangan software secara terdistribusi agar dapat menghasilkan sebuah produk yang deliverable dan useable dengan menganalisa beberapa kasus yang telah ada.

***Kata Kunci**: scrum, pengembangan software, pengembangan software terdistribusi*

1. PENDAHULUAN

Pengembangan software terdistribusi atau yang biasa disebut dengan Distributed Software Development (DSD) merupakan pengembangan software yang dilakukan oleh beberapa kelompok pengembang yang terdapat di lokasi yang berbeda, dipisahkan oleh jarak, bahkan juga dengan zona waktu yang berbeda.

DSD merupakan pendekatan pengembangan software terkini yang sesuai dengan tuntutan globalisasi (Leal, et al, 2010). Ketersediaan resource di lokasi yang berbeda, ketepatan waktu, atau kurangnya tenaga ahli lokal dapat menjadi alasan guna diaplikasikannya DSD (Keil, et al, 2011). Shrivastava et al (2010) menyatakan lokasi geografis pada DSD didefiniskan dalam dimensi : *onshore*, yang berarti seluruh kinerja pengembangan perusahaan berada di negara yang sama dengan *headquarter* dan segala operasinya dan *offshore* berarti bagian dari pengembangan yang terjadi di luar negeri.

Masalah yang umum dihadapi oleh perusahaan-perusahaan yang menggunakan DSD dalam pengembangan software-nya adalah komunikasi, koordinasi, dan kontrol (Paasivaara, et al, 2009). Leal et al (2010) membagi tantangan yang mungkin dihadapi oleh perusahaan-perusahaan ini menjadi 2 faktor, yaitu tantangan dari faktor teknis dan tantangan dari faktor non-teknis. Tantangan dari faktor teknis dapat berupa segala sesuatu yang berhubungan dengan konektivitas jaringan dan perbedaan antara lingkungan pengembangan dengan pengujian. Adapun tantangan dari segi non-teknis adalah kepercayaan, komunikasi, konflik, dan perbedaan budaya.

Scrum merupakan sebuah kerangka kerja dimana pihak-pihak dapat mencari jalan keluar dari permasalahan yang kompleks dan pada saat yang bersamaan membuat produk yang memiliki nilai setinggi mungkin secara produktif dan kreatif (Schwaber, et al, 2011). Scrum dirancang untuk meningkatkan kecepatan waktu pengembangan, menyatukan tujuan individu dan organisasi, menciptakan budaya yang diarahkan oleh performa, mendukung nilai kreasi dari *shareholder*, mencapai komunikasi yang stabil dan konsisten untuk setiap level performa, dan meningkatkan pengembangan dan kualitas hidup individu (Sutherland, et al, 2007). Schwaber (2011) memaparkan sifat-sifat yang dimiliki Scrum, yaitu :

1. Sederhana
2. Mudah dimengerti
3. Susah dikuasai

Menurut Sutherland, et al (2007), faktor utama yang mempengaruhi diperkenalkannya Scrum pada Easel Corporation berhubungan dengan permasalahan yang sering dijumpai pada pengembangan software berikut ini :

1. Ketidak-pastian yang tidak dapat dihindari dari proses pengembangan software – *Ziv's Uncertainty Principle*.

2. Untuk sebuah sistem software yang baru analisa kebutuhan tidak akan secara lengkap diketahui hingga user telah menggunakannya langsung – *Humphrey's Requirement Uncertainty Principle*.
3. Sangat tidak mungkin untuk menspesifikasikan secara lengkap system yang interaktif – *Wegner's Lemma*.
4. Kebutuhan yang selalu ambigu dan berubah, dikombinasikan dengan teknologi dan tools yang terus berkembang membuat implementasi tidak dapat diprediksi.

Sehingga diperlukan sebuah proses pembuatan software yang bekerja secara terus-menerus dalam analisa kebutuhan, design, coding, testing, dan juga mengantarkan sistem secara keseluruhan.

Jurnal ini disusun sebagai berikut. Bab 2 menyajikan relevant work yang telah ada yaitu implementasi Scrum pada DSD. Bab 3 menjelaskan secara singkat tentang Scrum dan pengembangan perangkat lunak secara terdistribusi. Bab 4 merupakan implementasi Scrum pada DSD yang dirangkum dari beberapa kasus yang ada, dan bab 5 menyajikan kesimpulan dari jurnal ini.

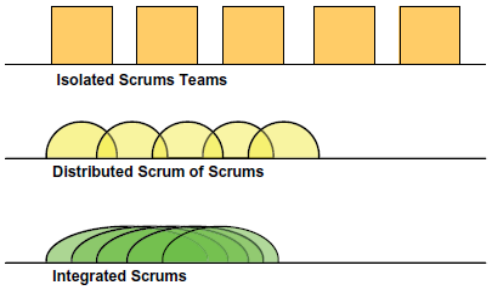
2. TINJAUAN PUSTAKA

Terdapat beberapa jurnal yang berhubungan dengan penggunaan scrum pada beberapa *study case* di perusahaan-perusahaan baik dengan tim pengembangan software secara terdistribusi, *offshore*, maupun *outsourcing*.

Paasivaara et al (2008) menyatakan bahwa segala bentuk penggunaan Scrum pada DSD telah memberikan hasil yang positif, dan mampu mengatasi masalah utama pada DSD yaitu komunikasi, dimana tim yang terdistribusi dipaksa untuk berkomunikasi dengan lebih sering dan mempelajari cara berkomunikasi dengan benar.

Sutherland et al (2007), memperkenalkan model Scrum terdistribusi yang sering ditemui dilapangan seperti berikut ini :

1. *Isolated Scrum Team* : tim terisolasi secara geografis. Pada beberapa kasus tim *offshore* tidak *crossfunctional* dan kemungkinan belum pernah menggunakan Scrum.
2. *Distributed Scrum of Scrums* : tim terisolasi secara geografis tapi terintegrasi oleh Scrum of Scrums yang bertemu secara rutin.
3. *Totally Integrated Scrums* : tim yang cross-functional yang tersebar antar geografis.



Gambar 1. Strategi tim Scrum terdistribusi

Penggunaan model *Integrated Scrum* dalam penelitian Sutherland et al (2007), menghasilkan bukti yang menyatakan bahwa tim yang terdistribusi sama produktifnya dengan tim yang berada di satu lokasi. Untuk mendapatkan hasil ini keseluruhan tim diharuskan berperan sebagai satu tim dengan global *repository*, *tracking* dan *reporting tool* yang sama, dan pertemuan harian (Daily Scrum) antar lokasi.

Nuevo et al (2011) pada penelitiannya menggabungkan Scrum dengan Rational Unified Process (RUP) untuk digunakan pada DSD. Pengadopsian Scrum pada DSD sangatlah bermanfaat karena metode ini menekankan pada tantangan utama dari DSD itu sendiri yaitu komunikasi dan koordinasi, sedangkan RUP pada DSD memastikan perlu adanya dokumentasi yang memungkinkan monitoring untuk tim yang terdistribusi.

Beberapa penelitian lain (Hossain, et al, 2009) menunjukkan bahwa tim terdistribusi yang menggunakan Scrum secara global menghadapi tantangan pada proses komunikasi, kontrol, dan koordinasinya. Hal ini dapat dipengaruhi oleh faktor distribusi yang memiliki jarak baik secara geografis maupun sosial-budaya. Perbedaan budaya ini nantinya dapat mempengaruhi proses kolaborasi.
Hossain et al (2009) juga menyatakan bahwa dalam praktik penggunaannya Scrum perlu diperluas agar dapat diadopsikan pada DSD dengan baik. Diperlukan beberapa strategi yang cocok untuk lingkungan pengembangan software, baik dengan menggunakan tim model yang berbeda, penggunaan jam kerja yang tersinkronisasi, tim *gathering*, dan juga dukungan dari berbagai macam *tools*.

Perbedaan antara jurnal ini dengan jurnal yang telah ada adalah bahwa jurnal ini mempelajari pengimplementasian Scrum pada DSD yang telah secara langsung diterapkan di perusahaan-perusahaan besar mancanegara, dan menyimpulkan manfaat serta tantangan yang ada pada pengimplementasian Scrum di DSD tersebut.

3. SCRUM DAN DSD

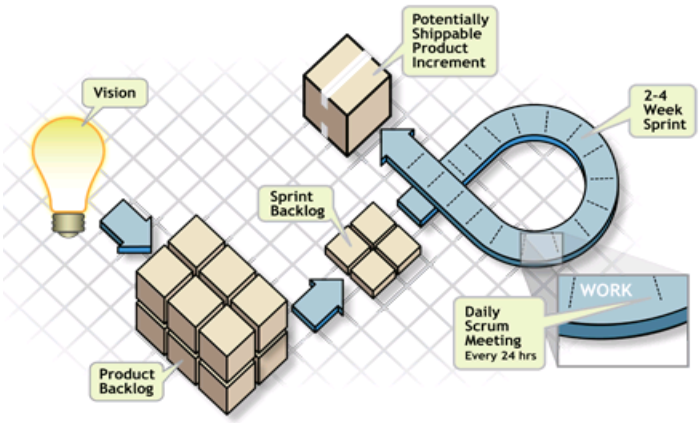
a. Praktik Scrum

Scrum bukanlah sebagai sebuah teknik untuk membuat produk, melainkan sebuah kerangka kerja dimana didalamnya kita dapat memasukkan berbagai proses dan teknik. Scrum akan menunjukkan hasil dari praktik pengelolaan dan pengembangan produk sehingga dapat menghasilkan produk yang lebih baik.

Schwaber, et al (2011) mendefinisikan kerangka kerja dari Scrum yang terdiri dari tim Scrum dan peran-peran yang diperlukan, acara (*event*), artefak (*artifact*), dan aturan main. Aturan main dari Scrum mengikat acara, peran, dan artefak, serta menggambarkan hubungan dan interaksi antara satu komponen dengan yang lainnya. Scrum Alliance mendefiniskan komponen-komponen yang terdapat pada Scrum adalah sebagai berikut:

1. Peranan :
   a. *Product owner* : bertanggung jawab terhadap nilai bisnis dari sebuah produk.
   b. *ScrumMaster* : memastikan tim yang dapat bekerja secara fungsional dan produktif.
   c. Tim : *self-organize* untuk menyelesaikan pekerjaan.
2. Acara :
   a. *Sprint planning* : tim bertemu dengan Product owner untuk memilih pekerjaan yang akan diselesaikan selama Sprint.
   b. *Daily Scrum* : tim bertemu setiap harinya untuk berbagi progress.
   c. *Sprint review* : tim mendemonstrasikan apa yang telah diselesaikan selama Sprint kepada Product owner.
   d. *Sprint retrospective* : tim mencari cara untuk meningkatkan produk dan proses.
3. Artefak :
   a. *Product backlog* : daftar prioritas dari proyek yang diinginkan.
   b. *Sprint backlog* : kumpulan pekerjaan yang disetujui oleh tim untuk diselesaikan dalam satu Sprint, dipecah menjadi beberapa *task*.
   c. *Burndown chart* : tampilan sekilas terhadap pekerjaan yang tersisa.

Scrum Alliance juga memberikan gambaran singkat tentang proses yang terjadi di dalam Scrum seperti diperlihatkan pada gambar berikut :



Gambar 2. Alur kerja Scrum

Penjelasan ringkas gambar diatas adalah :

- Product owner membuat daftar kebutuhan berprioritas yang disebut dengan Product backlog.
- Selama Sprint planning (meeting), tim (pengembang) memilah urutan teratas dari Product backlog (yang kemudian disebut dengan Sprint backlog), dan menentukan bagaimana cara pengimplementasiannya.
- Tim memiliki batasan waktu tertentu (Sprint) untuk menyelesaikan pekerjaanya, tetapi bertemu setiap hari untuk membicarakan progresnya (Daily Scrum).
- Selama Sprint, ScrumMaster memastikan tim tetap pada tujuannya.
- Pada akhir dari Sprint, produk harus telah siap "dipasarkan".
- Sprint berakhir dengan Sprint review dan retrospective.
- Ketika Sprint berikutnya dimulai, tim memilih urutan teratas berikutnya dari Product backlog.

b. Praktik DSD

Kendala utama yang dihadapi pada pengembangan software terdistribusi adalah kurangnya komunikasi antar tim yang terpisah baik oleh jarak maupun waktu. Selain itu, *resource* atau asset yang dimiliki, dan juga segi keperrcayaan sangat penting diperhatikan dalam pengembangan software terdistribusi (Leal, et al, 2010).

1. Komunikasi
   Tidak ada yang lebih baik dari komunikasi *face-to-face* (Guck, 2003). Penelitian (Niinimaki, 2011) menunjukkan keefektifan bentuk komunikasi secara langsung adalah dengan menggunakan :
   a. Komunikasi verbal : komunikasi secara langsung yang sering dilakukan oleh tim yang berada pada lokasi yang sama, atau jika tidak memungkinkan maka dengan bantuan teknologi sekarang ini dapat digunakan *teleconferencing* atau *videoconferencing.*
   b. Komunikasi email : untuk beberapa individu, email merupakan sarana komunikasi yang lebih dipilih dikarenakan faktor bahasa.
   c. Komunikasi IM : digunakan untuk *quick question*/*answer* untuk tim yang mengerjakan *task* yang sama.
2. *Sharing repositories*
   Penggunaan *repository* bersama memungkinkan seluruh tim untuk dapat berbagi *resource* dan asset yang dapat digunakan dalam pengembangan software.
3. *Keep remote developers involved*
   Perlu untuk membuat anggota tim merasa diperlukan di dalam tim. Hal ini bisa dilakukan dengan melakukan kunjungan-kunjungan ke tim yang lain, melakukan diskusi dan bertukar pikiran secara langsung, menghargai individu lain sehingga antar individu memiliki keterikatan untuk meraih tujuan bersama.

4. DISKUSI

Pada bagian ini akan dipaparkan bagaimana mengimplementasikan Scrum pada DSD berdasarkan *event-event* yang terdapat pada Scrum.

a. Sprint

Product owner menentukan lamanya Sprint dan merupakan satu-satunya pihak yang dapat membatalkan Sprint (Schwaber, et al, 2011). Sprint dimulai langsung setelah dihasilkan kesimpulan dari Sprint sebelumnya.

Lamanya Sprint dapat berbeda unutuk *onshore* dengan *offshore*-nya. Pada penelitiannya Paasivaara et al (2009) menemukan terdapat tim *onshore* yang memiliki panjang Sprint selama 4 minggu dan untuk tim *offshore*-nya hanya memiliki panjang 2 minggu. Hal ini ditujukan untuk memudahkan pemonitoran tim *offshore* oleh tim *onshore*.

b. Sprint Planning Meeting

Pada awal pertemuan dibahas Scrum *framework* yang memberikan gambaran tentang kerangka kerja Scrum (Paasivaara, et al, 2009). Umumnya pihak *onshore* lebih paham tentang hal ini daripada pihak *offshore*. Sehingga dibeberapa kasus, pihak *onshore* melakukan kunjungan kerja ke pihak *offshore* untuk menjelaskan tentang kerangka kerja Scrum, berbagi pengetahuan, mengenal satu sama lain, menyetujui cara kerja dan mengatur lingkungan kerja yang fungsional. Pada kunjungan kerja kali ini juga dibahas tentang tujuan jangkauan pendek dan mengengah yang akan dicapai, dan menentukan kriteria sukses serta ukuran kolaborasinya (Sutherland, et al, 2009).

Beberapa tim berbagi backlog melalui wiki (Paasivaara, et al, 2009), dan ketika dirasa wiki tak cukup untuk keperluan ini, beberapa tim lainnya menggunakan *tool* lain yang disebut Jira.

Pertemuan dilakukan oleh seluruh pihak (Product owner, ScrumMaster, dan tim yang terdapat di *onshore* maupun *offshore*) dengan menggunakan *tele* atau *videoconferencing* dan untuk berbagi aplikasi digunakan (Microsoft NetMeeting) (Paasivaara, et al, 2009).

Product owner memilih prioritas tertinggi untuk dikerjakan pada sprint berikutnya, yang kemudian dibagi menjadi beberapa task untuk dikerjakan secara paralel baik oleh tim *onshore* maupun tim *offshore*. ScrumMaster memberikan rancangan untuk Sprint dan ketergantungan-ketergantungan yang diidentifikasi selama perencanaan dengan tujuan untuk mengatasi masalah yang mungkin ada dari ketergantungan antar tim yang berbeda di setiap Sprint (Smits, et al, 2007).

c. Daily Scrum

Daily Scrum merupakan pertemuan yang dibatasi hanya selama 15 menit yang diadakan setiap harinya untuk tim pengembang mensikronisasikan segala aktifitas dan rencana yang akan dikerjakan selama 24 jam kedepan. Pada pertemuan ini, masing-masing anggota tim menyampaikan jawaban dari 3 pertanyaan :

1. Apa yang telah diperoleh/diselesaikan sejak pertemuan terakhir?
2. Apa yang akan dikerjakan sebelum pertemuan berikutnya?
3. Hambatan apa saja yang dijumpai oleh tim pengembang?

Beberapa tim merasakan perlunya menuliskan terlebih dahulu jawaban dari pertanyaan diatas, untuk menyingkat waktu dan membantu mengatasi masalah batasan-batasan bahasa (Sutherland, et al, 2007).

Pada kasus tertentu Daily Scrum dilakukan secara terdistribusi. Tim dengan lokasi yang sama mengadakan pertemuan di satu tempat. Tim *onshore* berpartisipasi di Daily Scrum tim *offshore* pada waktu tertentu untuk memonitor perkembangan tim *offshore*, seminggu pada awalnya, dan semakin jarang serta lebih sering menggunakan *teleconference* untuk mengikuti Daily Scrum tim *offshore* (Paasivaara, et al, 2009).

Pada kasus lainnya, dengan jumlah tim yang sangat besar dimungkinkan terjadinya *multiple* Daily Scrum. Yaitu masing-masing tim melakukan Daily Scrum untuk berbagi 3 jawaban dengan anggota lainnya sesama tim, kemudian diwakili oleh Scrumaster yang akan melakukan Daily Scrum dengan Scrum Master lainnya yang merupakan perwakilan dari tim-tim lainnya. Tantangan terbesar dari Daily Scrum jenis ini adalah bagaimana menjaga aliran informasi tetap sama hingga ke atas dan turun lagi ke tim pengembang (Smits, et al, 2007).

d. Sprint Review dan Sprint Retrospective

Sprint review merupakan pertemuan yang membahas tentang apa saja yang telah dilakukan selama Sprint. Sprint retrospective dilakukan setelah sprint review dan digunakan untuk inspeksi diri serta membahas apa saja yang akan dilakukan pada Sprint berikutnya (Schwaber, et al, 2011).

Demonstrasi proyek dilakukan dengan menggunakan *teleconferencing* dan *application sharing* dan dihadiri baik oleh tim *onshore* maupun *offshore*. Umumnya hanya ScrumMaster tim *onshore* dan *offshore* yang berpartisipasi dalam Sprint Retrospective (Paasivaara, et al, 2009).

5. KESIMPULAN

Pengembangan software secara terdistribusi membuktikan telah mampu menjawab permasalahan yang mungkin terjadi pada saat pengembangan software. Mulai dari kurangnya resource yang memadai di lokasi pengembangan, tim tenaga ahli yang kurang, hingga mengejar ketepatan waktu dapat diatasi dengan menggunakan pendekatan pengembangan software secara terdistribusi ini.

Komunikasi tetap menjadi kendala utama yang dihadapi dalam pengembangan software terdistribusi baik secara teknis maupun nonteknis. Secara teknis kendala komunikasi ditemui ketika terjadi ketidakstabilan koneksi jaringan yang digunakan dan kendala nonteknis berhubungan dengan bahasa dan latar belakang budaya anggota tim.

Scrum merupakan kerangka kerja yang dirancang untuk meningkatkan kecepatan waktu pengembangan software, memungkinkan adanya transfer ilmu pengetahuan dengan antar anggota tim, dan menciptakan komunikasi yang stabil dan konsisten bagi seluruh pihak yang terkait didalamnya.

Pengimplementasian Scrum pada pengembangan software secara terdistribusi dapat dilakukan dengan memberikan sedikit modifikasi akan alur kerja Scrum. Memilih anggota tim dengan skill tinggi, mengalokasikan lebih sedikit budget untuk pertemuan *face-to-face*, dan membangun lingkungan kerja yang kondusif merupakan syarat utama keberhasilan penggabungan kedua metode ini.

DAFTAR PUSTAKA

Guck, R., 2003, *Managing Distributed Software Development*, http://www.stickyminds.com/sitewide.asp?ObjectId=6002&Function=DETAILBROWSE&ObjectType=ART

Hossain, E., Babar, M.A., Paik, H., 2009, *Using Scrum in Global Software Development: A Systematic Literature Review*, IEEE Computer Science.

Keil, P., Kuhrmann, M., Niinimäki, T., 2011, *5th International Workshop on Tool Support Development and Management in Distributed Software Projects (REMIDI'11)*, IEEE Computer Science.

Leal, G.C.L., Delamaro, M.E., Huzita, E.H.M., Silva, C.A., 2010, *A Viability Study of An Integrated Approach of Software Development and Test to Distributed Teams*, IEEE Computer Science.

Niinimäki, T., 2011, *Face-to-face Email and Instant Messaging in Distributed Agile Software Development Project*, IEEE Computer Society.

Nordio, M., Estler, H.C., Meyer, B., Tschannen, J., Ghezzi, C., Nitto, E.D., 2011, *How do Distribution and Time Zones Affect Software Development? A Case Study on Communication*, IEEE Computer Society.

Nuevo, E., Piattini, M., Pino, F.J., 2011, *Scrum-based Methodology for Distributed Software Development*, IEEE Computer Society.

Paasivaara, M., Durasiewicz, S., Lassenius, C., 2008, *Distributed Agile Development: Using Scrum in a Large Project*, IEEE Computer Society.

Paasivaara, M., Durasiewicz, S., Lassenius, C., 2009, *Using Scrum in Distributed Agile Development: A Multiple Case Study*, IEEE Computer Science.

Schwaber, K., Sutherland, J., 2011, *The Definitive Guide to Scrum: The Rules of The Game*.

ScrumAlliance, *Scrum: The Basic*, http://www.scrumalliance.org

Shrivastava, S.V., Date, H., 2010, *Distributed Agile Software Development: A Review*, J of Computer Science and Engineering, vol 1 (1), pp 10-17.

Smits, H., Pshigoda, G., 2007, *Implementing Scrum in a Distributed Software Development Organization*, IEEE Computer Society.

Sutherland, J., Viktorov, A., Blount, J., & Puntikov, N., 2007, *Distributed Scrum: Agile Project Management with Outsourced Development Teams*, IEEE Computer Society.