**Tugas Paper Ilmiah**

***Extreme Programming***

**Sebagai Metodologi Pengembangan Sistem**

**M0214 - Topik - Topik Lanjutan Sistem Informasi**

**06-PFM**

**Disusun Oleh:**

**Machliza Devi Sasmita 1501169511**

****

**2014**

***Extreme Programming* Sebagai Metodologi Pengembangan Sistem**

**Machliza Devi Sasmita**

Jurusan Sistem Informasi, School of Information System, Universitas Bina Nusantara

Jl Kebon Jeruk Raya No. 27. Kebon Jeruk Jakarta Barat 11530

machlizadevi@gmail.com

**ABSTRAK**

Semakin meningkatkanya kepuasaan pelanggan, maka keinginan perusahaan untuk memberikan yang terbaik pun akan terus meningkat. Hal tersebut akan membuat sistem informasi dalam perusahaan harus terus dikembangkan setiap saat. Banyak sekali metodologi pengembangan sistem yang dapat digunakan. Salah satunya adalah *Extreme Programming*. *Extreme Programming* merupakan salah satu model proses dari *Agile Software Development* yang merupakan salah satu metodologi dalam pengembangan sistem berbasis *Software Development Life Cycle* (SDLC). Tujuan penulisan ini adalah untuk memberikan informasi dasar mengenai konsep *Extreme Programming*, kelemahan dan kekurangan maupun penerapannya dalam dunia nyata. Metodologi penulisan *paper* yang digunakan adalah dengan mencari teori-teori yang mendukung yang diperoleh dari Metode Studi Kepustakaan *(Library Research)* dan Studi *Web Browser (Web Research)*. Hasil yang dicapai adalah dengan adanya informasi mengenai *Extreme Programming* maka diharapkan tim *developer* dapat menentukan metodologi yang benar karena sudah mengetahui informasi dan kondisi seperti apa saat menerapkan *Extreme Programming*. Simpulan yang dapat diambil dari hasil penulisan ini adalah metodologi pengembangan sistem tidak hanya satu. Dan sebelum mengembangkan perangkat lunak, tim harus mempertimbangkan semua metodologi yang ada agar tidak terjadinya resiko kegagalan. *Extreme Programming* hanyalah satu dari sekian banyak metodologi yang ada.

Kata Kunci: *extreme programming*, metodologi, pengembangan sistem

**DAFTAR ISI**

Abstrak i

Daftar Isi ii

**BAB 1. PENDAHULUAN**

1.1 Latar Belakang 1

1.2 Ruang Lingkup 2

1.3 Tujuan dan Manfaat 2

1.4 Metodologi Penulisan 2

**BAB 2. LANDASAN TEORI**

2.1 Konsep Sistem Informasi 3

2.2 Konsep Pengembangan Sistem 5

**BAB 3. PEMBAHASAN**

3.1 Sejarah *Extreme Programming* 10

3.2 Nilai-Nilai Dasar *Extreme Programming* 10

3.3 Aspek Dasar *Extreme Programming* 12

3.4 Penerapan *Extreme Programming* 14

3.5 Kerangka Kerja *Extreme Programming* 16

3.6 Keuntungan dan Kerugian Dasar *Extreme Programming* 17

**BAB 5. PENUTUP**

4.1 Kesimpulan 19

4.2 Saran 19

**DAFTAR PUSTAKA** 20

**DAFTAR RIWAYAT HIDUP** L1

**BAB 1**

**PENDAHULUAN**

* 1. **Latar Belakang**

Dengan adanya perkembangan teknologi informasi yang semakin pesat, akan membuat proses bisnis dan sistem yang digunakan oleh perusahaan harus terus *up to date.* Selain karena banyak sekali teknologi baru yang bermunculan, tantangan perusahaan akan ditemukan pada semakin tingginya keinginan dan harapan pelanggan terhadap perusahaan, semakin tingginya persaingan dalam pasar internasional serta adanya ketidak puasan dengan sistem lama. Oleh karena itu, perusahaan harus terus melakukan pengembangan sistem informasinya demi mencapai *goal*, visi maupun misi perusahaan.

Pengembangan sistem informasi adalah suatu aktivitas untuk menyempurnakan sistem atau perangkat lunak yang pernah dibuat sebelumnya. pengembangan sistem ini, tidak bisa dilakukan hanya dalam waktu satu atau dua hari saja. pengembangan sistem harus dilakukan dengan matang dan terencana. Tidak hanya itu, tidak menutup kemungkinan dalam mengerjakan proyek pengembangan sistem informasi akan terjadi perubahan waktu secara tiba-tiba yang akan membuat tim pengembang sistem informasi dihadapkan kedalam kondisi maupun situasi yang mendesak. Sebagai contoh, saat Anda menjadi seorang *project leader* pada sebuah proyek pengembangan perangkat lunak, dan proyek yang Anda lakukan sudah hampir selesai tetapi klien tiba-tiba mengubah kebijakan sesuai keinginannya, apakah hal tersebut hal yang menyenangkan? Tentu tidak. Dari awal proyek dilaksanakan, tentunya tim pengembangan sudah melakukan rencana untuk kedepannya untuk menghindarinya hal-hal yang tidak diinginkan. Untuk menanggulangi situasi seperti itu, maka terdapat salah satu cara untuk mengatasi perubahan situasi dan kondisi cepat, yaitu *Extreme Programming*.

*Extreme Programming* merupakan salah satu model proses dari *Agile Software Development* yang merupakan salah satu metodologi dalam pengembangan sistem berbasis *Software Development Life Cycle* (SDLC). *Extreme Programming* atau yang dikenal sebagai XP adalah sebuah model pengembangan sistem yang menyederhanakan berbagai tahapan proses pengembangan tersebut agar tercapainya peningkatan efisiensi dan fleksibelitas sebuah proyek pengembangan perangkat lunak. Bahkan, menurut Shahriar Mohammadi, Bahman Nikkhahan, dan Sahar Sohrabi (2009) dalam jurnalnya yang berjudul *Challenges of user Involvement in Extreme Programming projects*, *Extreme Programming* merupakan metodologi yang tangkas dan banyak dipakai dalam perusahaan karena praktik *Extreme Programming* semuanya dilakukan untuk pelanggan. Tetapi, Anda tidak boleh salah sangka dengan kata-kata ‘*programming*’ yang terdapat dalam *Extreme Programming*, *Extreme Programming* tidak hanya berfokus pada *source code* atau *coding*, tetapi meliputi seluruh area pengembangan.

* 1. **Ruang Lingkup**

Pembahasan masalah yang akan di kerjakan dalam penulisan *paper* ilmiah adalah sebagai berikut:

* Sejarah *Extreme Programming*
* Aspek dasar *Extreme Programming*
* Nilai dasar *Extreme Programming*
* Kondisi dan penerapan *Extreme Programming*
* Keuntungan dan kerugian dalam penggunaan E*xtreme Programming*
* Aktivitas dalam *Extreme Programming*
  1. **Tujuan dan Manfaat**

Tujuan dari penulisan *paper* ilmiah ini adalah:

* Membuat pembaca mengerti dan memahami dengan pengertian dan konsep dari *Extreme Programming*.
* Menjelaskan keuntungan dan kelemahan dari Ex*treme Programming*
* Menjelaskan penerapan *Extreme Programming* dalam dunia nyata

Manfaat dari penulisan *paper* ilmiah ini adalah:

* Pembaca bisa menjelaskan dan mengerti tentang apa itu *Extreme Programming* dan konsep dari *Extreme Programming* itu sendiri.
* Pembaca tahu kapan harus menerapkan *Extreme Programming*, karena metodologi XP ini tidak bisa digunakan setiap saat.
* Pembaca dapat menerapkan metodologi *Extreme Programming* pada saat dibutuhkan.
  1. **Metodologi**

Metodologi yang digunakan dalam penulisan *paper* ilmiah ini adalah metode Studi Kepustakaan *(Library Research)* dan Studi *Web Browser (Web Research)*. Metode ini digunakan untuk mengumpulkan teori dan informasi dari buku-buku, hasil penelitian ilmiah, dan situs-situs yang mengandung informasi teori.

**BAB 2**

**LANDASAN TEORI**

**2.1 Konsep Sistem Informasi**

Pengertian sistem informasi sangat beragam. Satzinger(2010, p6-7) berpendapat bahwa sistem informasi merupakan gabungan dari pengertian sistem dan informasi, sehingga dapat disimpulkan sistem informasi adalah kumpulan-kumpulan komponen yang saling terhubung satu sama lain untuk mengumpulkan dan memproses data hingga menjadi suatu hasil yang tentunya dapat digunakan untuk menyelesaikan masalah. O’Brien (2008, p26) mengatakan bahwa sistem informasi terbagi menjadi banyak komponen, seperti sumber daya manusia, *hardware*, *software*, *data resources* dan *network research*.

1. ***People Resources***

Komponen pertama sistem informasi adalah sumber daya manusia. Sumber daya manusia sangat dibutuhkan dalam sistem informasi untuk mengoperasikan seluruh sistem informasi. Sumber daya manusia itu sendiri terbagi menjadi beberapa bagian, yaitu:

* *End Users*

Yang termasuk dalam *end user* disebut dengan *knowledge worker,* dimana mereka bekerjasama dalam tim untuk mengumpulkan, memakai dan mendistribusikan informasi.

* *IS specialists*

Merupakan orang-orang yang mengembangkan dan menjalankan sistem informasi. *IS Specialist* itu sendiri terbagi menjadi 3 bagian yaitu sistem analis, *software developer* dan sistem operator. Sistem analis bertugas untuk melakukan perancangan pada sistem sesuai dengan kebutuhan dari *end user* dan *software developer* bertugas untuk membuat program berdasarkan spesifikasi yang diberikan oleh sistem analis dan kemudian sistem operator bertugas dalam membantu memonitor dan menjalankan sistem komputer.

1. ***Hardware Resources***

Yang kedua adalah *Hardware resource*. *Hardware resource* merupakan semua perangkat fisik yang dipakai untuk memproses suatu informasi. Contoh dari hardware resource yaitu:

* Sistem komputer

Merupakan pusat dari bagian pemrosesan yang berisi *micro processor* dan perangkat pelengkap interkoneksi lainnya.

* *Computer peripherals*

Yang termasuk dalam *computer peripherals* yaitu perangkat fisik seperti *keyboard/mouse* yang digunakan untuk memasukkan data dan perintah dan video screen/printer yang digunakan untuk menghasilkan informasi dan *magnetic/optical disc* yang digunakan untuk menyimpan sumber dari data.

1. ***Software Resources***

Komponen ketiga adalah *Software Resource*. *Software Resource* merupakan sekumpulan petunjuk untuk digunakan untuk memproses informasi. Contoh dari *software resources* :

* Sistem *software*

Terdiri dari program pengoperasian sistem yang digunakan untuk mengendalikan dan mendukung operasi pada sistem komputer.

* Aplikasi *software*

Program yang digunakan untuk memproses sebagian penggunaan komputer secara lagsung dan dilakukan oleh *end users*.

* Prosedur

Petunjuk operasional yang ditujukan kepada user yang akan menggunakan sistem informasi tersebut.

1. ***Data resources***

Yang keempat adalah *data resources.* Sumber data yang diatur, disimpan dan diakses oleh manajemen sumber data ke dalam :

* + - * Database yang menangani pemrosesan dan pengaturan data.
      * Pengetahuan dasar yang menangani ilmu pengetahuan seperti fakta, aturan, contoh masalah mengenai praktik bisnis yang sukses.

1. ***Network resources***

Dan yang terakhir adalah *Network Resources. Network Resources* memperhatikan teknologi komunikasi dan jaringan yang ada pada sumber daya dalam sistem informasi. Namun yang diperhatikan hanya yang termasuk kedalam komponen dasar dari sumber daya tersebut saja.Yang termasuk di dalam *network resources*:

* Media Komunikasi

Contohnya yaitu *twisted-pair wire*, kabel koaksial, dan *fiber-optical cable* dan *microwave*, *cellular* dan teknologi *wireless-satelite*.

* Network Support

*Network Support* memperhatikan *hardware, software* dan teknologi data yang dibutuhkan untuk mendukung operasi dan penggunaan jaringan komunikasi.

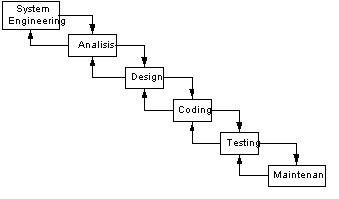
**2.2 Konsep Pengembangan Perangkat Lunak**

Seperti yang telah dibahas, perangkat lunak atau *software* merupakan salah satu komponen penting dalam sistem informasi. Menurut Al Bahra Bin Ladjamudin (2006, p3) perangkat lunak merupakan suatu program pada komputer yang menghubungkan antara *user* dengan *hardware*. Perangkat lunak juga berfungsi untuk menerjemahkan perintah yang diinput oleh *user* agar perintah tersebut dapat diproses oleh *hardware*. Tetapi, Perangkat lunak pada perusahaan tidak bisa terus menerus dipakai dalam jangka waktu yang lama. Mengapa demikian? Karena semakin tingginya kepuasaan dan ilmu yang dikuasai *user* maka semakin lama keinginan dan harapan akan sistem menjadi lebih tinggi. Oleh karena itu, perangkat lunak pada perusahaan harus terus dikembangkan agar menjadi lebih baik.

Menurut H.M Jogiyanto (2010, p59), pengembangan sistem informasi merupakan suatu kegiatan untuk mengembangkan sistem agar perusahaan mendapatkan sistem informasi yang dapat menyelesaikan masalah yang terjadi dalam perusahaan serta memanfaatkan peluang yang ada. pengembangan sistem itu sendiri tidak sekedar dikembangkan dalam waktu 1-2 hari saja, tetapi harus dirancang sedemikian rupa. Oleh karena itu terdapat banyak sekali metodologi pengembangan sistem informasi. Menurut Tata Sutabri (2013, p56) metodologi pengembangan sistem informasi adalah suatu metode yang dipakai untuk mengembangkan sistem perusahaan. Tata Sutabri juga berpendapat bahwa pengembangan sistem harus terencana dan menggambarkan tahapan-tahapan yang terinci yang meliputi *initiation, analysis, design* dan *implementation* yang sering disebut sebagai pendekatan *System Development Life Cycle* (SDLC). Sebelum mengembangkan perangkat lunak tersebut, tim pengembang harus memilih metodologi yang dipakai. Dan tentunya tidak boleh sampai salah pilih karena mereka memiliki keuntungan dan kerugian masing-masing. Berikut adalah beberapa contoh metodologi pengembangan sistem informasi:

* *Waterfall*

Merupakan suatu model pengembangan yang bersifat klasik dan sistematis karena model ini dikerjakan secara berurutan sehingga semuanya terinci. Tetapi, model ini akan memakan waktu yang cukup lama sehingga model ini sangat sedikit untuk digunakan dalam membuat perangkat lunak.



**Gambar 2.1 Tahap model *waterfall***

Tahap-tahap model *waterfall*:

1. *System Engineering*

Tahap ini digunakan untuk menganalisa kebutuhan user.

1. *Analysis*

Setelah menganalisa kebutuhan dari user maka proses selanjutnya yaitu menentukan jumlah arsitektur sistem, ukuran data dan jumlah data secara keseluruhan.

1. *Design*

Pada tahap ini akan dilakukan pemilihan struktur data, program, arsitektur data dan interaksi dengan user.

1. *Coding*

Setelah melakukan proses *design* maka *design* tersebut akan diubah menjadi *code program*.

1. *Testing*

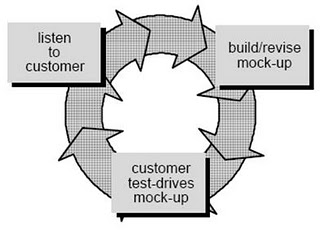
Setelah *program* telah di *coding* maka akan dilakukan *testing* pada *program* untuk mengecek apakah *program* masih ada *error* atau tidak.

1. *Maintenance*

Proses menjaga *software* agar dapat terus digunakan.

* *Prototype*

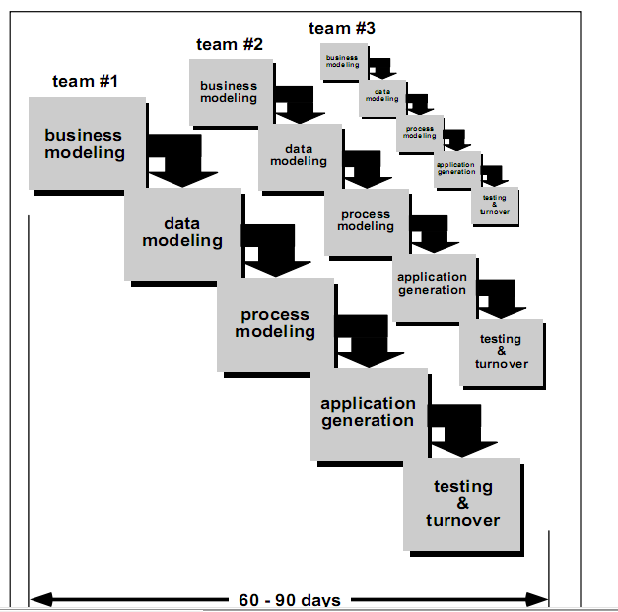
Metode ini memberikan deksripsi lengkap mengenai sistem sehingga user dapat melihat dari sisi tampilan maupun sisi teknik prosedural yang akan dibuat.



**Gambar 2.2 Tahap model *prototype***

* *RAD (Rapid Application development)*

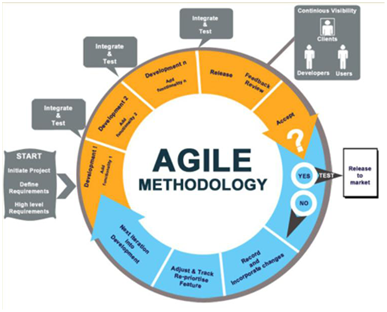
Merupakan model pengembangan berbasis komponen yang dapat dibuat dengan cepat. Untuk membuat siklus pengembangan dengan cepat, maka tim *developer* harus memahami kebutuhan *user* dengan baik.



**Gambar 2.3 Tahap model *RAD***

* *Agile Sofware Development*

Merupakan model pengembangan sistem yang cepat dan lebih mementingkan interaksi dengan *user* daripada proses dan alat. Karena kepuasan klien merupakan prioritas utama bagi *agile software development*.



**Gambar 2.4 Tahap model *Agile Software Development***

Berikut adalah beberapa *agile process model*:

* *Adaptive Software Development (ASD)*

**

**Gambar 2.5 Tahap model *ASD***

Merupakan metode yang digunakan untuk membangun *software* dan sistem yang kompleks.

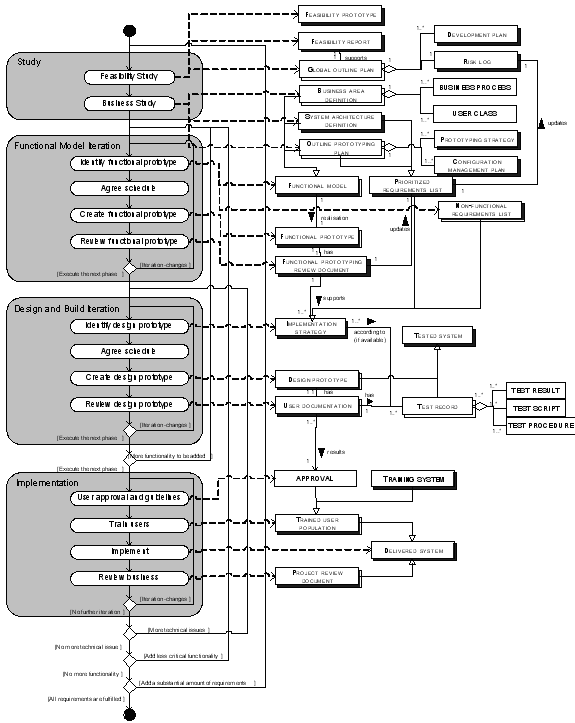
Beberapa aktivitas dalam ASD :

* + - *Speculation*
    - *Collaboration*
    - *Learning*
* *Dynamic Systems Development Method (DSDM)*

Merupakan metode yang menyediakan *framework* untuk membuat dan menjaga sistem dengan waktu yang terbatas melalui *prototyping*. Metode ini membangun *software* dengan cepat.

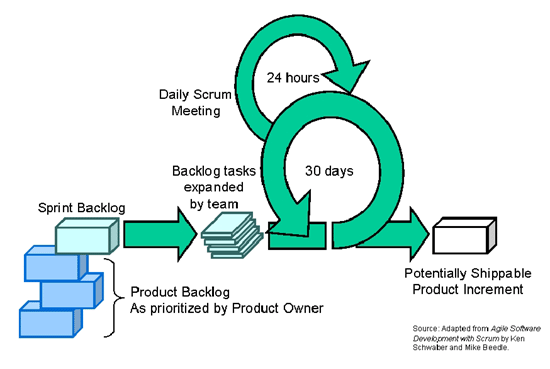
Beberapa aktivitas dalam DSDM:

* + 1. *Feasibility study*
    2. *Business Study*
    3. *Functional model iteration*
    4. *Design and build iteration*
    5. *Implementation*

**

**Gambar 2.6 Tahap model *DSDM***

* *Scrum*

**

**Gambar 2.7 Tahap model *scrum***

Merupakan metode yang dapat menyesuaikan diri dengan perubahan teknis maupun bisnis. Pada proses *scrum, testing* dan dokumentasi tetap dilakukan meskipun *software* sudah selesai dibuat. Beberapa aktivitas pada *Scrum*:

* *Backlog*
* *Sprints*
* *Scrum Meetings*
* *Demo*
* *Extreme Programming* (XP)

Merupakan suatu pendekatan pengembangan *software* yang digunakan untuk meningkatkan dan menyederhanakan suatu proyek agar menjadi lebih fleksibel. Dalam *paper* ilmiah ini akan dijelaskan lebih lanjut mengenai *Extreme Programming*.

**BAB 3**

**PEMBAHASAN**

**3.1 Sejarah *Extreme Programming***

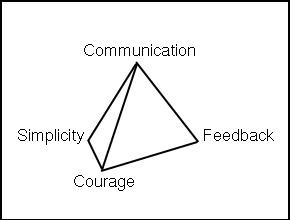
Sebelum mengenal lebih dalam mengenai kinerja dan proses E*xtreme Programming*, pertama-tama, kita harus tahu apa yang dimaksud dengan *Extreme Programming* dan bagaimana sejarah singkatnya. *Extreme Programming* atau yang biasa dikenal dengan “XP”, ini adalah sebuah metodologi yang diciptakan oleh Kent Beck ketika menangani sebuah project dari Chrysler yang dikenal dengan C3 (*Chrysler Comprehensive Compensation*). Proyek ini dimulai pada bulan Maret 1996 dan terancam gagal karena rumitnya sistem yang dibangun dan kegagalan pada saat testing. Pada saat itu, Chrysler menyewa Kent Beck sebagai salah satu pakar *software engineering* yang dikemudian hari dikenal sebagai pencetus awal dari *Extreme Programming*. Beck bersama rekannya Ron Jeffries dengan kewenangan yang diberikan oleh Chrysler melakukan berbagai perubahan di C3 *Project* untuk membuat project tersebut lebih efisien, dan fleksibel. Beck dan Jeffries akhirnya berhasil menyelesaikan target Chrysler dengan menerapkan berbagai metode dalam proses pengembangan sistem tersebut. Kumpulan metode inilah yang kemudian dikenal sebagai metodologi atau pendekatan *Extreme Programming* dalam pengembangan perangkat lunak.

*Extreme Programming* terlihat sebagai kumpulan ide lama yang terlalu sederhana dan tidak akan memberikan efek apapun pada sebuah proyek pengembangan perangkat lunak. Kent dan Beck sendiri mengakui dan menegaskan bahwa *Extreme Programming* tidak selalu cocok untuk setiap proyek pengembangan perangkat lunak. Kelebihan *Extreme Programming* adalah sesuai untuk digunakan pada proyek yang memiliki *Dynamic requirements*, atau proyek tersebut memiliki berbagai *requirements* yang tidak jelas dari user.

Proyek semacam ini memerlukan adaptasi yang cepat dalam mengatasi perubahan-perubahan yang terjadi selama proses pengembangan sistem berlangsung. *Extreme Programming* juga cocok untuk proyek dengan jumlah anggota tim yang tidak terlalu banyak dan berada pada lokasi yang sama.

**3.2 Nilai-Nilai Dasar *Extreme Programming***

**Pada setiap metodologi pengembangan sistem tentunya ada beberapa hal dasar yang harus diperhatikan sebelum lanjut kedalam kerangka kerja agar tidak terjadi kesalahan pemilihan metodologi. Pada setiap tahapan proses pengembangan sistem yang menggunakan model proses *Extreme Programming* juga memiliki nilai-nilai mendasar yang harus diperhatikan, yaitu:**



**Gambar 3.1 Nilai Dasar *Extreme Programming***

***Communication***

***Extreme Programming* berfokus bagaimana cara agar hubungan komunikasi antar anggota tim terjalin dengan baik. Setiap anggota tim harus bisa saling pengertian dan berbagi pengetahuan serta keterampilan mengenai pengembangan perangkat lunak. Terutama dalam hal ego yang tinggi para *programmer*, disini mereka harus bisa menekankan ego masing-masing dan bersosialisasi agar dapat bekerja sama dengan *programmer* lain, karena ini merupakan tugas utama tim *developer*.**

***Courage***

**Demi tercapainya proyek pengembangan yang sukses, setiap masing-masing anggota tim harus selalu memiliki keberanian, keyakinan serta integritas dalam melakukan tugasnya. Anggota tim harus konsisten dalam menjaga segala jenis kondisi termasuk saat adanya tekanan dari klien atau pemilik perusahaan. Anggota tim juga dituntut untuk berani mengerjakan tugas dan setiap menemukan kesalahan harus segera diperbaiki.**

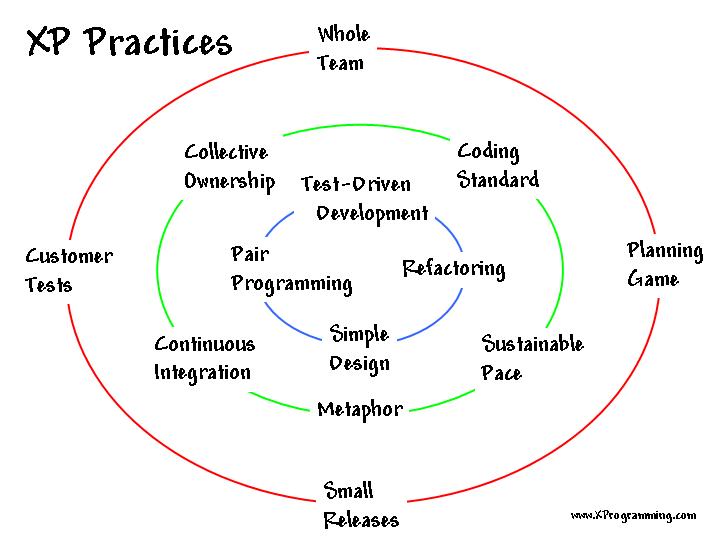
***Simplicity***

**Dengan menggunakan *Extreme Programming*, artinya tim *developer* harus tetap melakukan semuanya dengan sederhana, karena salah satu nilai dasar dari *Extreme Programming* adalahmencoba untuk selalu mencari solusi yang paling sederhana dan paling praktis. Gunakanlah *method* yang mudah, pendek atau tidak terlalu rumit dalam pembuatan desain serta hilangkanlah fitur-fitur yang tidak berguna. Inilah yang membedakan *Extreme Programming* dengan metodologi pengembangan sistem konvensional lainnya. *Extreme Programming* berfokus pada kebutuhan saat ini dibandingkan kebutuhan besok.**

***Feedback***

**Hal ini merupakan salah satu hal penting yang bertujuan untuk mengetahui kemajuan dari proses maupun kualitas perangkat lunak yang dibangun. Dan sudah menjadi kewajiban anggota tim maupun pihak lainnya untuk terlibat dalam memberikan *feedback* yang baik. Anggota tim harus selalu mendengarkan pendapat anggota tim lainnya tanpa adanya cemohan terhadap pendapatnya tersebut. Dengan hal tersebutlah terkadang akan menyadari kita di bagian mana yang dapat ditingkatkan lagi.**

**3.3 Aspek Dasar *Extreme Programming***

****

**Gambar 3.2 Aspek Dasar *Extreme Programming***

**Terdapat 12 aspek dasar *Extreme Programming*, yaitu:**

1. ***The Planning Game***

Pendekatan ***Extreme Programming*** pada tahap ini sangat mirip dengan metode penerapan RAD (*Rapid Application Development*) yang prosesnya pendek dan cepat serta mengutamakan aspek teknik. Proses ini juga menggunakan istilah game karena menurut Beck, untuk menentukan *requirements* dibutuhkan penggunaan teknik *score card*.

1. ***Small Releases***

Semua developer akan menyelesaikan setiap bagian dari perangkat lunak dan kemudian hasilnya akan dipresentasikan kepada perusahaan (klien). Apabila bagian sistem tersebut disetujui oleh perusahaan maka bagian tersebut akan diterapkan ke seluruh sistem yang ada di perusahaan dan kemudian akan dilakukan tes awal pada penerapan sistem tersebut.

1. ***Metaphor***

*Metaphor* pada dasarnya sama dengan arsitektur *software* yaitu sama-sama menggambarkan visi tujuan dari pengembangan sistem secara luas. Selain itu, *metaphor* bertujuan untuk menjaga komunikasi antara klien dengan *developer* tetap baik.

1. ***Simple Design***

Pada proses ini, ***Extreme Programming*** didesain sesederhana mungkin untuk menghindari pengembangan sistem yang sulit dimengerti. Dengan desain yang sederhana ini akan memberikan keuntungan apabila terjadi perubahan pada desain yang mengharuskan untuk membuat desain baru maka resiko kegagalan pada desain dapat diperkecil.

1. ***Refactoring***

*Refactoring* merupakan suatu aspek khusus pada ***Extreme Programming***. *Refactoring* adalah proses mengubah kode program dari perangkat lunak yang bertujuan untuk meningkatkan kualitas dari struktur program agar *developer* tidak harus mengulang proses desain.

1. ***Testing***

Pada proses ini, *developer team* harus membuat tes yang akan dipakai untuk menjalankan *software* dan mengembangkannya. Pada saat proses *coding* selesai maka *software* tersebut akan diuji menggunakan model tes yang telah dibuat sebelumnya.

1. ***Pair Programming***

Pair programming merupakan aspek di mana 2 orang *programmer* bersama-sama dalam satu tempat atau komputer ,mereka saling berinteraksi dan berdiskusi tentang suatu masalah dan menyelesaikannya bersama.

1. ***Collective Ownership***

Didalam aspek ini, para *programmer* dituntut untuk saling berbagi informasi, sehingga ketergantungant erhadap seorang *programmer* terhadap programmer lain dapat diminimalisir.

1. ***Coding Standart***

*Coding standart* merupakan suatu kesepakatan antara *programmer* dalam memahami *script/program* tersebut, sehingga para *programmer* memiliki pemahaman yang sama.

1. ***Continous Integration***

Di dalam aspek ini, para *software developer* diwajibkan untuk melakukan *build*, *build* ditujukan agar kesalahanpada program dapat dideteksi dan diperbaiki secepat mungkin. Maksimal waktu yang di butuhkan untuk melakukan build di dalam ***Extreme Programming*** adalah 1 hari.

1. ***40-hours week***

Kent Beck berpendapat bahwa, waktu ideal seorang *programmer* bekerja adalah 8 jam/hari atau 5 kali dalam seminggu.

1. ***On-Side Customer***

Dalam proses pengembangan *software*, ***Extreme Programming*** menganjurkan untuk mengikut sertakan seorang klien. Klien tersebut ikut berpartisipasi dalam proses *build* dan *test* yang di lakukan. Klien tersebut di harapkan dapat memberikan masukan dan koreksi atas pengembangan yang di lakukan.

**3.4 Penerapan *Extreme Programming***

Penggunaan sebuah metodologi pengembangan sistem tidaklah sembarangan. *Extreme Programming* itu sendiri tepat digunakan jika:

* Keperluan berubah sangat cepat
* Tingginya resiko
* Terdapat proyek dengan tantangan baru
* Memiliki sedikit tim *programmer* hanya sekitar 2-10 orang
* Mampu mengotomatisasikan tes
* Adanya peran serta pelanggan secara langsung

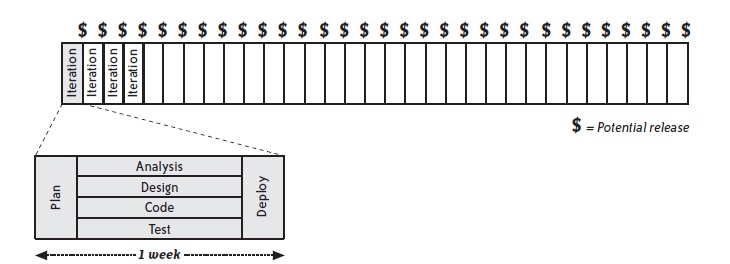
Jadi, untuk menerapkan *Extreme Programmin*g, tim dalam sebuah organisasi tidak boleh asal sembarangan saja, tetapi mereka harus mengetahui metodologi yang akan di pakai. Ada beberapa metodologi lain selain *Extreme Programming* seperti metodologi *waterfall*, metodologi *agile*, metodologi *scrum*, metodologi *iterative* dan metodologi lainnya.

Sebuah tim dalam E*xtreme Programming* harus melakukan pengembangan sistem secara bersamaan. Analisis, Desain, Coding, Testing, Implementasi, terjadi sangat cepat ketika tim sedang melakukan pengembangan perangkat lunak.

Ada beberapa hal penting yang harus diketahui oleh tim *Extreme Programming*, tentunya seperti :

1. Tim harus secara intensif berhubungan dengan user sehingga apa yang di inginkan oleh user untuk sistem yang sedang dikembangkan sesuai dengan apa yang ditentukan oleh user.
2. Tim harus berkomunikasi dengan baik antar sesama anggota ataupun dengan user, agar tujuan pengembangan sistem dapat diselesaikan dengan cepat.
3. Ada baiknya jika manajer tim proyek pernah menangani dalam pengembangan, pembuatan, pembaharuan suatu sistem, agar manajer proyek bisa memimpin timnya dengan lancar.

Penerapan *Extreme Programming* bisa dilakukan di banyak perusahaan atau organisasi. Sebuah organisasi seperti ***software house*** atau pengembangan sebuah software dapat menerapkan *extreme programming* karena *extreme programming* mengedepankan *extreme programming life cycle*.

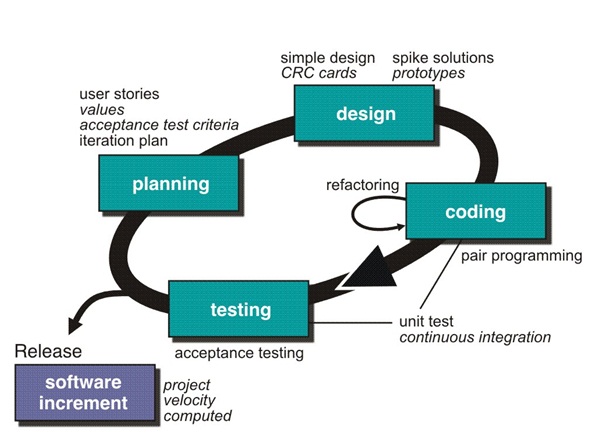


**Gambar 3.3** *Extreme programming* Life Cycle

*Extreme programming* sangat menghemat waktu untuk membangun atau membuat sebuah sistem. Dalam *software house,* penerapan *extreme programming* dapat dilakukan karena struktur kerja dan struktur organisasi yang menyerupai *extreme programming*. Pada organisasi *software house* terdapat analis, desainer, *programmer*, dan *tester*. Oleh karena itu *extreme programing* sangat tepat diterapkan di *software house*.

**3.6 Kerangka Kerja *Extreme Programming***

Menurut Pressman dalam bukunya yang berjudul *Software Engineering*, edisi keenam, proses *Extreme Programming* memiliki kerangka kerja yang terbagi menjadi empat konteks aktivitas utama. Empat konteks tersebut adalah *Planning, Design, Coding* dan *Testing*. Keempat aktivitas inilah yang akan menghasilkan sebuah perangkat lunakyang didasari dengan konsep model *Extreme Programming*. Selain gambar dibawah ini yang memberikan kesimpulan bagaimana penggunaan *Extreme Programming*, akan dijelaskan mengenai empat konteks tersebut secara lebih detail.



**Gambar 3. Kerangka Kerja** *Extreme Programming*

***Planning***

Pada *Planning* berfokus untuk mendapatkan gambaran fitur dan fungsi dari perangkat lunak yang akan dibangun. Aktivitas *planning* dimulai dengan membuat kumpulan gambaran atau cerita yang telah diberikan oleh klien yang akan menjadi gambaran dasar dari perangkat lunak tersebut. Kumpulan gambaran atau cerita tersebut akan dikumpulkan dalam sebuah indeks dimana setiap poin memiliki prioritasnya masing-masing. Tim pengembang aplikasi juga akan menentukan perkiraaan waktu serta biaya yang dibutuhkan untuk masing-masing indeks. Setelah semua kebutuhan terpenuhi, tim XP akan menentukan alur dari pengembangan aplikasi sebelum memulai pengembangan tugas.

Selama proses pengembangan perangkat lunak, klien dapat mengubah setiap rencana dari aplikasi yang dibuat. Tim XP akan mempertimbangkan semua hal yang ingin diubah oleh klien sebelum mengubah aplikasi tersebut.

***Design***

Aktivitas *design* dalam pengembangan aplikasi ini, bertujuan untuk mengatur pola logika dalam sistem. Sebuah desain aplikasi yang baik adalah desain yang dapat mengurangi ketergantungan antar setiap proses pada sebuah sistem. Jika salah satu fitur pada sistem mengalami kerusakan, maka hal tersebut tidak akan mempengaruhi sistem secara keseluruhan.

Tahap *Design* pada model proses *Extreme Programming* merupakan panduan dalam membangun perangkat lunak yang didasari dari cerita klien sebelumnya yang telah dikumpulkan pada tahap *planning*. Dalam XP, proses *design* terjadi sebelum dan sesudah aktivitas *coding* berlangsung. Artinya, aktivitas *design* terjadi secara terus-menerus selama proses pengembangan aplikasi berlangsung.

***Coding***

Setelah menyelesaikan gambaran dasar perangkat lunak dan menyelesaikan *design* untuk aplikasi secara keseluruhan, XP lebih merekomendasikan tim untuk membuat modul unit tes terlebih dahulu yang bertujuan untuk melakukan uji coba setiap cerita dan gambaran yang diberikan oleh klien. Setelah berbagai unit tes selesai dibangun, tim barulah melanjutkan aktivitasnya ke penulisan *coding* aplikasi. XP menerapkan konsep *Pair Programming* dimana setiap tugas sebuah modul dikembangkan oleh dua orang *programmer*. XP beranggapan, 2 orang akan lebih cepat dan baik dalam menyelesaikan sebuah masalah. Selanjutnya, modul aplikasi yang sudah selesai dibangun akan digabungkan dengan aplikasi utama.

***Testing***

Walaupun tahapan uji coba sudah dilakukan pada tahapan *coding*, XP juga akan melakukan pengujian sistem yang sudah sempurna. Pada tahap *coding*, XP akan terus mengecek dan memperbaiki semua masalah-masalah yang terjadi walaupun hanya masalah kecil. Setiap modul yang sedang dikembangkan, akan diuji terlebih dahulu dengan modul unit tes yang telah dibuat sebelumnya.

Setelah semua modul selesai dan dikumpulkan ke dalam sebuah sistem yang sempurna, maka tim XP akan melakukan pengujian penerimaan atau *acceptance test*. Pada tahap ini, aplikasi akan langsung diuji coba oleh *user* dank lien agar mendapat tanggapan langsung mengenai penerapan gambaran dan cerita yang telah dilakukan sebelumnya.

**3.6 Keuntungan dan Kerugian *Extreme Programming***

*Extreme Programming* merupakan serangkaian proses yang dapat dilihat dan juga dapat dipertanggung jawabkan. Para *developer* akan membuat komitmen tentang tujuan mereka dan memperlihatkan perkembangannya dalam bentuk *deployable* *software*, dan ketika tujuan mereka tercapai, mereka akan merincikan apa yang mereka lakukan dalam proses tersebut serta perubahannya. Dan hal ini memungkinkan orang lain untuk membuat bisnis dan mengambil keuntungan dari peluang yang ada, sehingga bisnis mereka tidak akan mudah mati.

Sehingga dapat disimpulkan keuntungan dan kelemahan dari *Extreme Programming* adalah:

Keuntungan:

* Ada interaksi yang baik antara *developer*
* Memberikan pembelajaran kepada orang lain (*client*)
* Pelanggan mendapatkan timbal balik yang akurat
* Dapat mengubah pemikiran pelanggan
* *Developer* tidak bekerja secara berlebihan
* Dapat membuat keputusan teknikal

Kelemahan:

* *Extreme Programing* hanya menjalankan 1 proyek dan dijalankan oleh 1 tim
* *Developer* harus dan akan menghadapi perubahan yang harus mereka jalankan.
* *Extreme Programming* tidak dapat dikerjakan jika *developer* saling terpisah
* *Extreme programming* belum terbukti dapat bekerja di dalam sistem yang mempunyai masalah skalabilitas.

**BAB 4**

**PENUTUPAN**

**4.1 Kesimpulan**

Berdasarkan pembahasan dari *Extreme Programming* pada bab-bab sebelumnya, maka dapat disimpulkan sebagai berikut:

1. *Extreme Programming* berguna untuk mempercepat pekerjaan suatu tim dalam organisasi atau perusahaan. Karena dalam Extreme Programming life cycle menuntut ke suatu tim untuk menyelesaikan rangkaian aktivitas *Planning, Analys, Design & Code, Test, Deploy* dalam tempo waktu yang telah ditentukan.
2. *Extreme Programming* merupakan salah satu metodologi yang mendukung percepatan pembangunan suatu sistem.
3. *Extreme Programming* memprioritaskan komunikasi yang baik antar klien maupun antar sesama anggota tim.
4. Dalam penerapan *Extreme Programming* tanggung jawab per individu menjadi berkurang. Karena dalam mengembangkan suatu sistem, anggota tim selalu bekerja bersama-sama sehingga tanggung jawab per individu pun berkurang.
5. Semua anggota tim di tuntut untuk melakukan aktivitas *Planning, Analys, Design & Code, Test, Deploy.*
6. Tidak semua perusahaan atau organisasi bisa menerapkan metodologi eXtreme Programming, karena eXtreme Programming menuntut beberapa individu dengan spesialis berbeda (*Programmer, Designer, Analys, Tester*).

**4.2 Saran**

*Extreme Programming* bisa di terapkan pada perusahaan atau organisasi yang mempunyai tim khusus untuk pengembangan suatu sistem. Perusahaan atau organisasi yang ingin menerapkan *Extreme Programming* diharapkan untuk bisa memahami dan mengerti *Extreme Programming* *practice,* dan *scrum*. Jika perusahaan atau organisasi sudah mengerti Extreme Programming practice dan *scrum*, maka bisa mempelajari scrum terlebih dahulu lalu *full* *Extreme Programming*.

**DAFTAR PUSTAKA**

Ariello. 2009. Extreme Programming. *http://blog-ariello.blogspot.com/2009/03/extreme-programming.html*. 11 April 2014

Beck, Kent. 1999. Extreme Programming Explained: Embrace Change. Addison-Wesley.

Bin Ladjamudin, Al Bahra (2006). Rekayasa Perangkat Lunak. Tangerang: Graha Ilmu.

Iqbal, N., M.U. Hassan, A.R. Osman, M. Ahmad. 2013. A framework for partial implementation of PSP in Extreme programming. *International Journal of Engineering Research and Applications*. 3(2): 604-607

Jogiyanto, H.M. (2010). Analisis dan Desain. Yogyakarta : Andi Offset.

Levi . 2013. Model Extreme Programming (XP). *http://levi91.wordpress.com/2013/03/17/*

*model-extreme-programming-xp.* 10 April 2014.

Mohammadi, S., B. Nikkahan, S. Sohrabi. 2009. Challenges of user Involvement in Extreme Programming projects. *International Journal of Software Engineering and Its Applications*. 3(1): 20.

O'Brien, J. A., & Marakas, G. M. (2008). *Pengantar Sistem Informasi: Perspektif Bisnis dan Manajerial Edisi 12*. Jakarta: Salemba Empat.

Satzinger, W, J., Jackson, B, R., Burd, S. D (2010). *System Analysis and Design in a Changing World*. USA: Cengage Learning.

Shore, James., Warden, Shane. 2007. *The Art of Agile Development*. O’Reilly Media.

Sutabri, Tata. (2012). *Analisis Sistem Informasi*. Yogyakarta : Andi Offset.

Umar, Abdullah. Extreme Programming. *http://sistementerprise.weebly.com/extreme-programming.html*. 10 April 2014

Wibowo, Ary. 2012. Metode pengembangan sistem AGILE. *http://blog.arywibowo.com/2012/05/metode-pengembangan-perangkat-lunak.html.* 10 April 2014.

**Daftar Riwayat Hidup**

Nama : Machliza Devi Sasmita

Tempat, Tanggal Lahir : Jakarta, 04 Agustus 1993

Jenis Kelamin : Perempuan

Alamat : Jl. KH Syahdan No 4B RT 002/012 Kelurahan

Palmerah Kecamatan Palmerah 11480 Jakarta Barat

Nomor Telepon : 081290603190

**Pendidikan Formal**

1997 - 1999 TK Nurul Islam, Jakarta

1999 - 2005 SD Negeri 15 Pagi, Jakarta

2005 - 2008 SMP Negeri 111, Jakarta

2008 - 2011 SMA Negeri 112, Jakarta

2011 - SekarangBina Nusantara University Jurusan Sistem Informasi,

Jakarta

**Pengalaman Kerja**

**2009-2010 Representatives of Provoke! Magazine**

2009-2011 Reporter My School Page Hai Magazine

2013 BINUS Online Job Expo Committee

2013 *Copywriter* Sribu