STRATEGI PENGEMBANGAN INDUSTRI PIRANTI LUNAK BERBASIS CMM DI INDONESIA

Dodi Wisaksono Sudiharto
dodi_ws@hotmail.com

ABSTRAK

Penulisan ini adalah mengenai kebijakan dan strategi bagi industri piranti lunak secara umum, dan khususnya industri piranti lunak dalam negeri.Adapun standar yang diterapkan adalah standar minimal sehingga memicu para pengembang piranti lunak untuk terus meningkatkan kualitas produknya. Dengan adanya kebijakan dan strategi ini diharapkan dapat memberi arahan untuk pengembangan lebih lanjut terhadap produksi industri piranti lunak untuk dapat memenuhi standar yang diakui di dunia internasional dengan menggunakan *CMM, Software.*

Kata Kunci : *CMM,Software,Industri*

PENDAHULUAN

Industri piranti lunak menunjukkan keuntungan bagi perekonomian Indonesia dalam rentang waktu 5 tahun (1996 - 2001). Hal ini bukan hanya memberikan dampak keuntungan bagi perekonomian, namunjuga membuka lapangan kerja baru di Indonesia Ironisnya, *Business Software Associate* (BSA) mencatat bahwa pembajakan piranti lunak di Indonesia termasuk yang tertinggi di dunia. Hal ini dapat merusak citra dari industti piranti Lunak Indonesia di masa-masa yang akan datang. Padahal tidak sedikit industn piranti lunak yang berdiri di Indonesia menghasilkan aplikasi yang berkualitas (bila ditinjau dari kemudahan pemakaian dan penggunaannya oleh *end-user).* Namun karena tidak adanya acuan baku yang dapat digunakan sebagai standar yang bisa diakui dunia internasional, sebagai bukti bahwa piranti lunak produksi dalam negeri cukup layak untuk digunakan dan berkualitas internasional (seperli standar ISO untuk kualitas produk), karena seringnya *user-requiretnetti* yang berubah-ubah baik karena kurangnya kemampuan *system* dari *fevetoper* maupun *itser* yang tidak mengetahui secara detail mengenai *process* maupun *deliverable* dari piranti lunak yang diinginkan, dan tidak adanya dokumen

resmi yang bisa diakui bersama oleh pihak *developer* maupun *user* atau pemakai, yang bisa dijadikan dasar *software-acceptance*, sehingga banyak terjadi kegagalan dalam memproduksi suatu piranti lunak.

Untuk itu, adanya standarisasi di bidang software menjadi sangat penting. Ini tidak hanya rnenyangkut hasil akhir dari peranti lunak, tetapi juga mencakup proses pembuatannya Sehingga semenjak awal siklus piranti lunak tersebut, sejak pendefnisian *user-requirement* sampai dengan realisasinya, semuanya terangkum dalam suatu dokumen baku yang disepakati bersama (atau disebut juga sebagai dokumen standar). Dokumen tersebut harus detail dan menggambarkan *deliverable* yang akan dihasilkan, karena akan menjadi acuan dalam realisasi peranti lunak sekaligus sebagai dasar *traceability* bila ada kesalahan dalam pengembangan peranti lunak tersebut

Standarisasi yang kami usulkan mengacu pada *Software Life Cycle Development* secara umum dan standar *Capability Maturity Model* (CMM) dari *Software Engineering Institute* (SEI) Mengingat tujuan dan kebijakan dan standarisasi ini adalah agar piranti lunak buatan industri dalam negeri dapat diakui di dunia internasional, dapat bersaing dengan industri-industri piranti lunak dari luar negeri, maka diambil standar minimal yang bisa diterima oleh standar international umuk menghasilkan piranti lunak yang berkualilas

PEMBAHASAN

Terminologi *Capability Maturity Model* (CMM) yang dikembangkan *Software Enigineering Institute* (SEI) digunakan dalam pengembangan proses piranti lunak di dunia. untuk menjabarkan kesatuan antara orang, metode serta alat bantu yang digunakan untuk menghasilkan piranti lunak CMM sendiri bukanlah *software life cycle model*, melainkan tingkatan yang bisa dicapai secara bertahap oleh pengembang piranti lunak

Untuk menghasilkan piranti lunak yang berkualitas dan berdaya saing serta berdaya jual, baik dari segi mutu maupun orang-orang yang terlibat didalam pengembangannya, maka kebijakan yang diusulkan tidak dikotak-kotakkan dalam beberapa level seperti CMM yang kita kenal, tetapi

menetapkan standar minimal yang diambil dari aktivitas-aktivitas maupun metode terbaik yang ada di tiap level dari CMM dengan tidak adanya pengkotak-kotakan tersebut, diharapkan industri piranti lunak Indonesia tidak berpuas diri dengan melakukan pencapaian di suatu level tertentu. Dan nilai minimun lerbaik dari level CMM yang bisa dijadikan dasar acuan adalah nilai dari *define level*

Pengadaan kebijakan standarisasi ini disebabkan adanya perbedaan mendasar antara standarisasi untuk produksi industri piranti lunak umumnya dengan produksi perangkat keras yang menggunakan standarisasi ISO. Selain itu, juga berkaitan dengan manajemen IT yang berbeda dengan manajemen yang digunakan untuk IT .

Standar Software Life Cycle Development

Software Life Cycle Development adalah aktivitas yang terjadi selama pengembangan, pemakaian dan pemeliharaan sistem perangkat lunak. Menurut Pressman (2001), *life cycle* atau proses iterasi atau rangkaian aktiitas dalam *software engineering* dapat disebut juga sebagai suatu evolusi dalam *software process.*

Ada beberapa jenis *software life cycle material*, tetapi beberapa aktvitas yang mutlak perlu dalam pengembangan suatu perangkat lunak dan dianjurkan sebagai standar adalah sebagai berikut :

- *Gather/rig Software Requirement*
- *Software Destgn*
- *Construction*
- *Software Testing*
- *Software Maintenance or Implementation*

Adapun untuk masing-masing aktivitas di atas juga hatus diakhiri dengan adanya *user documentation acceptance* dan *user phase approval.* Bertujuan agar akhir dan *deliverable* dari tiap-tiap fase disadari tidak hanya oleh pengembang peranti lunak, tapi juga oleh pemakai agar masing-masing pihak menyadari dan *aware* bahwa hasil dari piranti lunak yang berkualitas juga dapat tergantung oleh orang-orang di sekitar pcngembangan peranti lunak tersebut, yang dalam hal ini adalah pengembang dan *user* atau pemakai.

Gathering Software Requirement

Gathering Software Requirement adalah aktivitas mengumpulkan semua informasi dan dokumen yang dibutuhkan untuk *meng-captare* kebutuhan *user* akan piranti lunak yang dibuat Dalam hal ini item-item yang perlu didapat, dipahami bersama baik oleh pengembang maupun pemakai di antaranya adalah. *Huwiie A process, flow process, flow document* dan *user interface* yang diinginkan

Sedangkan *deliverable* dokumen yang dihasilkan pada fase *gathering software requirement* adalah :

- *Software Development Plan* (SDP)
- *Software Standar and Procedure Manual*
- *Software Configuration Management Project*
- *Software Requirement Specification*
- *Interfax Requirement Specfication*
- *Operational Concept Document*

Setelah pengumpulan *software requirement* aktivitas selanjutnya adalah mendesain piranti lunak tersebut. fase ini terbagi dua, yaitu fase *preliminary design* dan fase *detail design.*

Deliverable dari fase *preliminary design* adalah :

- *Software lest pc-Ian*
- *Operator's manual*
- *User's manual*
- *Diagnostic manual* Sedangkan *deliverable* dari detail desain adalah :
  - *Software Detail Design Document*
  - *Interface Design Document*
  - *Database Design Document*
  - *Software DevelopmentFiles*
  - *Integration Test Cases*
  - *Software Test Description*

Coding Construction

Coding construction adalah tahap *transition* dari piranti lunak Pada tahap ini, yang menjadi acuan dari pengembangan oleh pihak pengembang adalah dokumen-dokumen yang dihasilkan pada tahap sebelumnya. Pada tahap ini, bila ada perubahan yang diminta oleh *end-user,* maka pihak pengembang akan mengacu pada dokumen yang sudah disepakati bersama Bila perubahan tersebut hanya merupakan tambahan kecil yang belum berkembang dapat merealisasikannya dan mencantumkan perubahan yang ada di setiap dokumen.

Kesimpulan

Tidak ada proses pengulangan di setiap pengembangan sistem / perangkat lunak adalah harapan semua pengembang, sehingga tidak ada waktu dan effort yang terbuang percuma. Dengan metode CMM ini diharapkan di setiap level pengembangan yang ada pada *Software Life Cycle Development* memeiliki nilai atau kualitas yang terbaik.

DAFTAR PUSTAKA

Jeffery L. Whitten, Lonnie D. Bentley, System Analysis and Design Methods 5th Edition , Mc Graw Hill, 1997

Roger R. Presman, Software Engineering A Practitioner's Approach, Fifth Edition, Mc Graw Hill, 2000