Rekayasa Perangkat Lunak

Modul 5

Software Requirement Specification (SRS)

Disajikan Oleh :
Irman Hariman, ST., MT.
Email : iirmanhariman@gmail.com

Software Requirement Specification

Merupakan sebuah dokumen yang berisi pernyataan lengkap dari apa yang dapat dilakukan oleh perangkat lunak, tanpa menjelaskan bagaimana hal tersebut dikerjakan oleh perangkat lunak.

Tujuan Pembuatan SKPL (SRS)

Tujuan penulisan SKPL adalah untuk mendefinisikan keinginan yang biasanya dinyatakan dalam bentuk penjelasan umum.

Untuk yang kedua, tujuan pembuatan SKPL adalah :

- Sarana komunikasi antara pelanggan, pemakai, analis, dan perancang perangkat lunak.
- Dasar untuk merencanakan dan melaksanakan aktivitas pengujian sistem.
- Acuan untuk melakukan perbaikan dan perubahan perangkat lunak.

Manfaat dan kegunaan SKPL menurut Witarto[WIT04] dari IEEE, adalah :

- Memastikan kesamaan antara kebutuhan pengembangan dengan kebutuhan yang ditulis didalam dokumen.
- Mendefinisikan kerangka kerja untuk proses-proses pengembangan perangkat lunak.
- Memperjelas peran dan antarmuka bagi para pihak yang terlibat dalam proses pengembangan perangkat lunak.
- Memperjelas jenis dan isi dokumen.
- Mengenali tugas, tahapan, baseline, aktivitas kaji ulang, dan dokumentasinya.
- Belajar pendekatan praktis yg diterapkan didunia industri.
- Menghilangkan persoalan-persoalan seperti yang pernah dialami masa lalu.

Syarat Pembentukan SKPL (SRS)

Empat syarat yang harus diperhatikan saat pembentukan SKPL, yaitu :

- Mudah diidentifikasi
- Diuraikan dengan jelas, simple, sederhana, dan concise (jelas, tidak ambiguous)
- Bisa divalidasi dan bisa dites (test reliable, test accessable)
- Mampu untuk ditelusuri kembali (tracebility)

Hindari hal-hal berikut saat pembentukan SKPL (SRS)

Empat syarat diatas harus memperhatikan kaidah-kaidah yang harus dihindari saat menyusunnya dan pastikan selalu hal dibawah ini dihindari :

- Over specification (penjelasan berlebih dan berulang-ulang sehingga menjadi tidak jelas)
- Tindakan unconcistency (seperti menggunakan istilah yang tidak konsisten)
- Ambiguity dalam kata atau kalimat seperti menyatakan keterukuran kebutuhan secara tidak jelas misalkan menggunakan kata-kata : minimal,maksimal, optimal, cepat, user friendly, efisien, fleksible dan lainnya.
- Menuliskan “mimpi-mimpi”, yaitu hal-hal yang tidak bisa dilakukan

Dua Aspek yang harus bisa dilihat :

- Fungsi; Menjelaskan fungsi dari perangkat lunak (digunakan untuk apa keperluan apa), sifat perangkat lunak, dan datanya.
- Non-fungsi :
    - Dependability
    - Ergonomic
    - Performance
    - Contraint

Atribut Penulisan SKPL (SRS) yang baik :

Dokumen SKPL yang baik (sempurna) akan ditulis secara :

1. Benar (correct)
   Suatu dokumen SKPL disebut benar jika dan hanya jika setiap kebutuhan yang dinyatakan dalam dokumen merepresentasikan sesuatu yang disyaratkan dari sistem yang akan diangun.

2. Tepat (precise)
   Berpengaruh pada hasil perancangan dan pembuatan software requirements design (SRD).

3. Unambiguouity
   Setiap permintaan harus punya satu intepretasi, atau hanya ada satu arti dalam satu kalimat.

4. Lengkap (complete)
   - Lengkap jika dilihat dari dua sudut pandang :
   - dokumen memuat tabel isi, nomor halaman, nomor gambar, nomor tabel, dan sebagainya.
   - tidak ada bagian yang hilang (to be define)

5. Bisa diverifikasi (verifiable)
   - Bisa diperiksa dan dicek kebenarannya. Setiap kebutuhan selalu dimulai dengan dokumen yang bisa diperiksa.

6. Konsisten

   Nilai-nilai kebutuhan harus tetap sama baik dalam karakteristik maupun spesifikasi, misalnya diminta A tetap ditulis A.

7. Understandable

   Dapat dimengerti oleh pemrogram, analis sistem atau system engineer.

8. Bisa dimodifikasi (modifiedable)
   Bisa diubah-ubah dan pengubahannya sangat sederhana tetapi tetap konsisten dan lengkap.
9. Dapat ditelusuri (traceable)
   Jika ditelusuri, harus tahu mana bagian yang diubah.
10. Harus dapat dibedakan bagian what (bagian spesifikasi) dan how (bagian yang menjelaskan bagaimana menyelesaikan what tadi).
11. Dapat mencakup dan melingkupi seluruh sistem
12. Dapat melingkupi semua lingkungan operasional,
13. Bisa menggambarkan sistem yang sesuai pemakai.
14. Harus toleran (bisa terima) terhadap ketidaklengkapan, ketidakpastian (ambiguous) dan ketidakkonsistenan.
15. Harus bisa dilokalisasi dengan sebuah coupling, yaitu hubungan ketergantungan antara dua model yang tidak terlalu erat.

Ada 9 macam orang yang terlibat dalam pembuatan SKPL (SRS) :

1. Pemakai (user)

   Kelompok orang yang mengoperasikan/menggunakan produk final dari PL yang dibuat.

2. Client

   Orang atau perusahaan yang mau membuat sistem (yang menentukan).

3. System analyst (system engineer)

   Kelompok orang yang biasa melakukan kontak teknik pertama dengan client. Bertugas menganalisis persoalan, menerima requirement dan menulis requirement.

4. Software engineer

   Kelompok orang yang bekerja setelah kebutuhan perangkat lunak dibuat (bekerja sama dgn system engineer saat mendefinisikan kebutuhan perangkat lunak & membuat deskripsi perancangannya).

1. Programmer

   Kelompok orang yang menerima spesifikasi perancangan perangkat lunak, membuat kode dalam bentuk modul, menguji dan memeriksa (tes) modul.
2. Test integration group, Kelompok orang yang melakukan tes dan mengintegrasi modul.
3. Maintenance group

   Kelompok orang yang memantau dan merawat performansi sistem perangkat lunak yang dibuat selama pelaksanaan dan pada saat modifikasi muncul (80% dari pekerjaan).
4. Technical Support

   Orang-orang yang mengelola (manage) pengembang perangkat lunak, termasuk konsultan atau orang yang mempunyai kepandaian lebih tinggi.
5. Staff dan Clerical Work

   Kelompok orang yang bertugas mengetik, memasukkan data, membuat dokumen.

Keberhasilan pengembangan perangkat lunak

1. Ketelitian dari pembuatnya
2. Kualitas dari spesifikasi perangkat lunak yang dihasilkan (baik, jika ada sedikit kesalahan)
3. Integritas
4. Ketelitian
5. Proses pembuatan yang mantap
6. Mudah dikembangkan
7. Jumlah versi tidak banyak
8. Ketelitian dari model pengembangan yang digunakan untuk meramal atribut perangkat lunak
9. Efektivitas rencana tes dan integrasi
10. Tingkat persiapan untuk sistem perawatan (mempersiapkan pencarian bugs)

Setiap metode analisis mempunyai pandangan yang berbeda. Tetapi pada dasarnya semua metode analisis memiliki prinsip analisis yang sama, yaitu :

1. Menggambarkan domain informasi masalah
2. Mendefinisikan fungsi perangkat lunak
3. Menghasilkan model yang menggambarkan informasi, fungsi dan kelakuan yang dibagi secara rinci pada sebuah model lapisan (hirarki)
4. Informasi pokok pada tahap analisis memudahkan tahap implementasi yang lebih rinci.

Tujuan kegiatan di tahap analisis

- Menjabarkan kebutuhan pemakai
- Meletakkan dasar-dasar untuk tahap perancangan perangkat lunak
- Mendefinisikan semua kebutuhan pemakai sesuai dengan lingkup kontrak yang disepakati kedua belah pihak (pengembang dan pengguna).