BAB II

LANDASAN TEORI

Dalam merancang dan membangun aplikasi, sangatlah penting untuk mengetahui terlebih dahulu dasar-dasar teori yang digunakan. Dasar-dasar teori tersebut digunakan sebagai landasan berpikir dalam melakukan pembahasan lebih lanjut sehingga terbentuk suatu aplikasi yang sesuai dengan tujuan awal.

2.1 Perencanaan Bahan Baku

Perencanaan adalah suatu proyeksi tentang apa yang diperlukan dalam rangka mencapai tujuan absah dan bernilai (Herjanto, 1999). Perencanaan merupakan salah satu sarana manajemen untuk mencapai tujuan yang telah ditetapkan, karena itu, setiap tingkat manajemen dalam organisasi sangat membutuhkan aktivitas perencanaan. Perencanaan juga merupakan fungsi memilih sasaran perusahaan secara kebijaksanaan, program dan pemilihan langkah-langkah apa yang harus dilakukan, siapa yang melakukan dan kapan aktivitasnya dilaksanakan.

Bahan baku adalah barang-barang yang dibeli untuk digunakan dalam proses produksi (Baridwan, 2003). Bahan baku memiliki beberapa faktor yang perlu diperhatikan (Kholmi, 2003), yaitu :

1. Perkiraan pemakaian

   Merupakan perkiraan tentang jumlah bahan baku yang akan digunakan oleh perusahaan untuk proses produksi pada periode yang akan datang.

2. Harga bahan baku

   Merupakan dasar penyusunan perhitungan dari perusahaan yang harus disediakan untuk investasi dalam bahan baku tersebut.

3. Biaya-biaya persediaan

   Merupakan biaya-biaya yang dibutuhkan oleh perusahaan untuk pengadaan bahan baku

4. Kebijaksanaan pembelanjaan

   Merupakan faktor penentu dalam menentukan berapa besar persediaan bahan baku yang mendapatkan dana dari perusahaan.

5. Pemakaian sesungguhnya

   Merupakan pemakaian bahan baku yang sesungguhnya dari periode lalu dan merupakan salah satu faktor yang perlu diperhatikan.

6. Waktu tunggu

   Merupakan tenggang waktu yang tepat maka perusahaan dapat membeli bahan baku pada saat yang tepat pula, sehingga resiko penumpukan ataupun kekurangan persediaan dapat ditekan seminimal mungkin.

Sehingga dapat disimpulkan bahwa perencanaan bahan baku adalah proses analisis dan penyusunan suatu kebutuhan bahan baku dengan memproyeksikan secara tepat sesuai tujuan yang ingin dicapai.

2.2 Just In Time(JIT)

Secara singkat prinsip *Just In Time* adalah menghilangkan sumber-sumber pemborosan produksi dengan cara menerima jumlah yang tepat dari bahan baku dan memproduksinya dalam jumlah yang tepat pada tempat yang tepat dan waktu yang tepat pula (Indrajid dan Pranoto, 2003). Tujuan dari adanya manajemen menggunakan dan mengembangkan konsep manajemen *Just In Time* dalam perusahaan dapat dirangkum sebagai berikut.

1. Flexibilitas

Menciptakan fleksibilitas produk yang tinggi produksi, bersifat “sistem tarik” (*pull system*) memerlukan fleksibilitas tinggi untuk menanggapi tuntutan konsumen yang terus berkembang. Produksi dengan cara “sistem tarik” (pendekatan baru) merupakan produksi yang dilakukan untuk menganggapi permintaan, sedangkan produksi dengan “sistem dorong” (pendekatan lama) merupakan produksi yang ditetapkan produsen kepada konsumen.

2. Meningkatkan efisiensi proses produksi

Peningkatan efisiensi dapat dilakukan terutama melalui pengurangan persediaan barang sehingga mengakibatkan pengurangan biaya persediaan, atau dengan kata lain meningkatkan perputaran modal. Biaya persediaan ini sangat tinggi, berkisar antara 20 persen–40 persen dari harga barang pertahun. Efisiensi didapat juga dengan cara mendesain pabrik sedemikian rupa sehingga proses produksi dapat dilakukan dengan lebih cepat dan aman.

3. Meningkatkan daya kompetisi

Meningkatnya efisiensi dalam proses produksi dengan sendirinya akan meningkatkan daya saing perusahaan. Hal ini dianggap salah satu tujuan yang paling penting, yaitu suatu tujuan strategis, karena peningkatan efisiensi berarti penurunan biaya dan ini memungkinkan perusahaan untuk tetap bertahan dalam persaingan pasar.

4. Meningkatkan mutu barang

Kemitraan pembeli-penjual yang dibina dan berlangsung dalam jangka panjang selalu berusaha untuk melakukan perbaikan secara terus menerus dalam hal mutu dan biaya barang. Mutu tinggi dari suku cadang atau komponen yang

dipasok oleh pemasok pada gilirannya akan meningkatkan mutu barang yang diproduksi oleh perusahaan. Kemitraan penjual pembeli memungkinkan melakukan pengendalian mutu suku cadang atau komponen dengan lebih murah dan lebih handal.

5. Mengurangi pemborosan

Pengurangan pemborosan terutama dalam bentuk barang yang terbuang, karena pada hakekatnya pemborosan adalah biaya. Menurut jenisnya, pemborosan dapat dibedakan dari cara pemborosan itu terjadi, yaitu:

a. Karena produksi berlebih (memproduksi barang dengan jumlah yang terlalu banyak).
b. Karena waktu tunggu (waktu tunggu yang tidak produktif dalam proses produksi perusahaan).
c. Karena transport (gerakan yang tidak perlu dalam proses produksi).
d. Karena proses (operasi atau proses yang tidak perlu).
e. Karena persediaan (penimbunan bahan baku, bahan setengah jadi, bahan jadi, atau bahan lain yang berlebih).
f. Karena gerakan (pengerjaan kembali atau hasil dari kegiatan-kegiatan yang tidak perlu).

2.3 Mesin Pemecah Batu (*Stone Crusher*)

Agregat penggunaan batu yang digunakan dalam pembangunan sebuah jalan, gedung, perumahan, dan bangunan lainnya dapat diambil dari alam (*quarry*) yang berupa pasir, kerikil atau batuan. Kadang batuan dari alam (*quarry*) berukuran besar sehingga perlu dilakukan pemecahan terhadap batuan tersebut agar dapat dimanfaatkan dalam campuran. Guna mendapatkan kerikil atau batuan pecah yang

sesuai dengan ukuran yang diharapkan, maka diperlukan suatu alat untuk memecah material tersebut. Alat pemecah batuan yang digunakan adalah *stone crusher*.

Pertama kepingan batu besar yang berdiameter ≤ 50 cm diambil dari *dump truck* dengan menggunakan *excavator bucket* kemudian diangkut ke mesin pemecah batu. Sebelum di tuang ke *stone crushing*, *dump truck* angkut tersebut di timbang, untuk mengetahui volume hasil penambangan.

Stone Crusher berfungsi untuk memecahkan batuan alam menjadi ukuran yang lebih kecil sesuai dengan spesifikasi yang dibutuhkan. Pada pekerjaan *crushing* ini biasanya diperlukan beberapa kali pengerjaan pemecahan, tahap-tahap pekerjaan ini beserta jenis *crusher* yang digunakan antara lain :

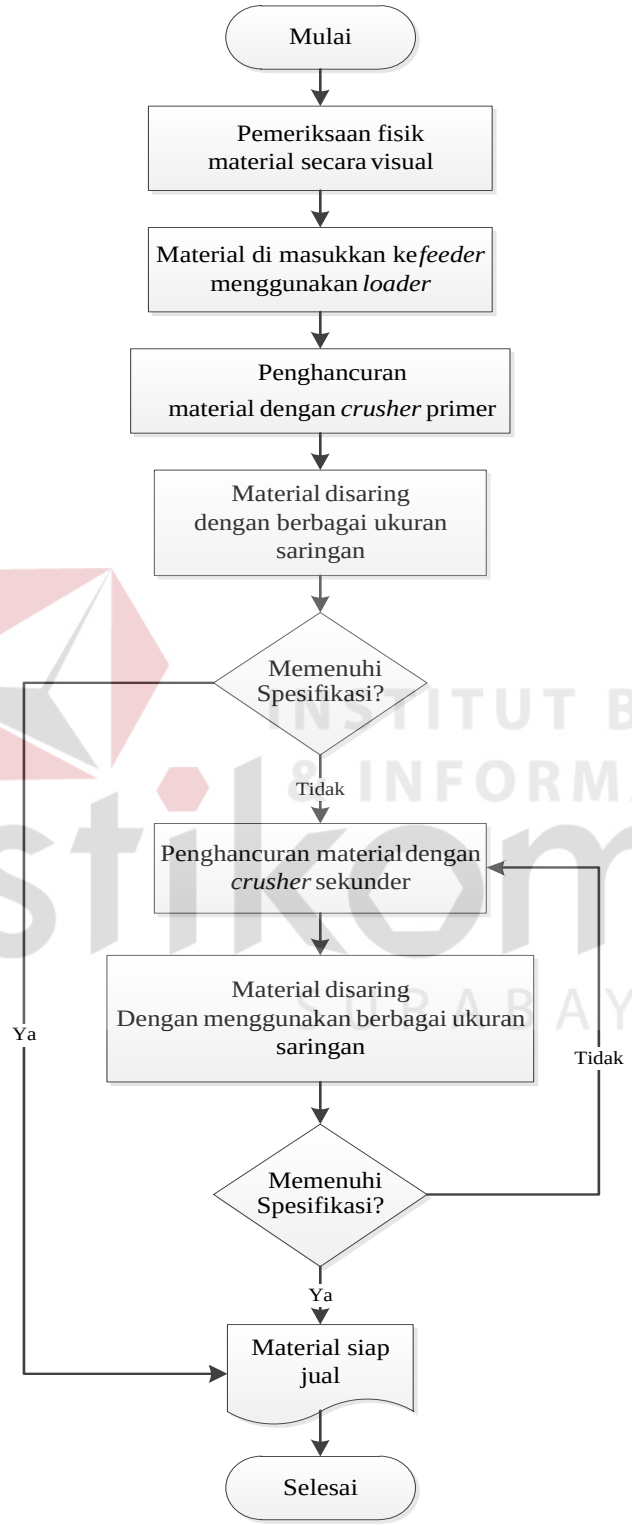
1. Pemecahan tahap pertama oleh jenis *primary crusher*.
2. Pemecahan tahap kedua oleh *secondary crusher*.
3. Pemecahan – pemecahan selanjutnya jika ternyata diperlukan, oleh *tertiary crusher.*

Pada perusahaan ini sudah meneliti sebuah perhitungan bahwa dalam sekali menghancurkan batu ukuran besar, maka yang diperoleh sebagai berikut:

Tabel 2.1 Macam-macam Ukuran Batu dan Presentase Hasil Tiap Ukuran

No.	Ukuran	Presentase (%)
1.	Abu Batu (<=4 mm)	10
2.	05-10 mm	25
3.	11-15 mm	25
4.	16-20 mm	25
5.	21-30 mm	15

Tahap-tahap pekerjaan pemecahan pada *crusher* dapat dari diagram alir material pada dilihat pada Gambar 2.1.



Gambar 2.1 Diagram Alir Material pada *Crusher*

Proses ini akan menghitung jumlah bahan baku yang harus dibeli oleh perusahaan berdasarkan jumlah dan jenis produk pada setiap hari menggunakan rumus dari analisis perhitungan yang dapat dilihat pada rumus 2.1.

$$BB = \frac{JB}{(x)\%} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

Keterangan :

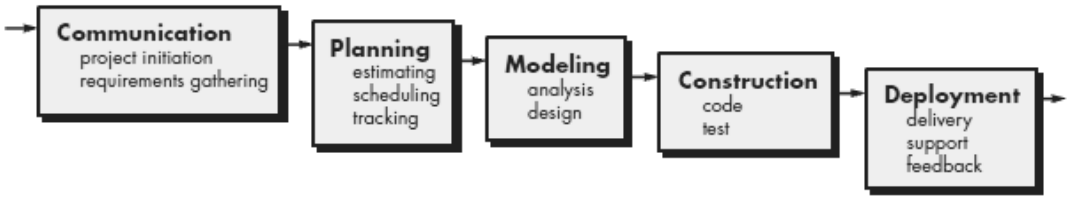
BB = Bahan baku yang dibutuhkan

$(x)\%$ = Persentase batu yang dihasilkan dari satu kali produksi

JB = Jumlah bahan baku yang dibutuhkan

2.4 Software Development Life Cycle (SDLC)

Model *Waterfall* adalah model klasik yang bersifat sistematis, dimana hal ini menyiratkan pendekatan yang sistematis dan berurutan (sekuensial) pada pengembangan perangkat lunak, yang dimulai dengan spesifikasi kebutuhan pengguna dan berlanjut melalui tahapan-tahapan perencanaan (*planning*), pemodelan (*modeling*), konstruksi (*construction*), serta penyerahan sistem perangkat lunak ke para pelanggan/pengguna (*deployment*), yang diakhiri dengan dukungan berkelanjutan pada perangkat lunak lengkap yang dihasilkan (Pressman, 2010). Fase-fase dalam model *waterfall*:



Gambar 2.2 Model Waterfall (Pressman, 2010)

1. *Communication*

   Langkah ini merupakan analisis terhadap kebutuhan *software*, dan tahap untuk mengadakan pengumpulan data dengan melakukan pertemuan dengan *customer*, maupun mengumpulkan data-data tambahan baik yang ada di jurnal, artikel, maupun dari internet.

2. *Planning*

   Proses *planning* merupakan lanjutan dari proses *communication (analysis requirement).* Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan *user* dalam pembuatan *software*, termasuk rencana yang akan dilakukan.

3. *Modeling*

   Proses *modeling* ini akan menerjemahkan syarat kebutuhan ke sebuah perancangan *software* yang dapat diperkirakan sebelum dibuat *coding*. Proses ini berfokus pada rancangan struktur data, arsitektur *software*, representasi *interface*, dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut *software requirement*.

4. *Construction*

   Construction merupakan proses membuat kode. *Coding* atau pengkodean merupakan penerjemahan desain dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan transaksi yang diminta oleh *user*. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu *software*, artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan *testing* terhadap sistem yang telah dibuat tadi. Tujuan *testing* adalah

menemukan kesalahan-kesalahan terhadap sistem tersebut untuk kemudian bisa diperbaiki.

5. *Deployment*

   Tahapan ini bisa dikatakan final dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan oleh *user*. Kemudian *software* yang telah dibuat harus dilakukan pemeliharaan secara berkala.

2.5 Sistem Informasi

Sistem informasi terdiri dari *input*, *proses* dan *output*. Pada proses ini terdapat hubungan timbal balik dengan dua elemen yaitu *control* dari kinerja sistem dan sumber-sumber penyimpanan data. *Input* yang akan diproses berupa data, baik berupa karakter-karakter huruf maupun numeri (Herlambang, 2005).

Data ini diproses dengan metode-metode tertentu dan akan menghasilkan *output* yang berupa informasi. Informasi yang dihasilkan dapat berupa laporan atau *report* maupun solusi dari proses yang telah dijalankan.

Sistem informasi terdiri dari komponen-komponen yang saling berinteraksi yaitu:

1. Komponen masukkan, yaitu data yang masuk ke dalam sistem informasi yang dapat berupa dokumen-dokumen dasar.
2. Komponen model, yaitu komponen yang terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah ditentukan untuk menghasilkan keluaran yang diinginkan

3. Komponen keluaran, yaitu komponen yang merupakan informasi yang berkualitas dan dokumentasi yang berguna.
4. Komponen teknologi, yaitu komponen yang digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian sistem secara keseluruhan. Komponen ini terbagi menjadi tiga bagian yaitu teknisi, perangkat lunak dan perangkat keras.
5. Komponen basis data, merupakan kumpulan data yang saling berkaitan dan berhubungan antara satu dengan lainnya. Basis data tersimpan dalam perangkat keras komputer dan perangkat lunak untuk memanipulasinya. Data dalam basis data perlu diorganisasikan sedemikian rupa dan digunakan untuk keperluan penyediaan informasi.

2.6 *Database*

Database adalah suatu susunan/kumpulan data kegiatan lengkap dari suatu organisasi/perusahaan yang diorganisir/dikelola dan disimpan secara terintegrasi dengan menggunakan metode tertentu menggunakan komputer sehingga mampu menyediakan informasi optimal yang diperlukan pemakainya (Marlinda, 2004). Penyusunan satu *database* digunakan untuk mengatasi masalah-masalah pada penyusunan data yaitu redundansi dan inkonsistensi data, kesulitan pengaksesan data, isolasi data untuk standarisasi, banyak pemakai (*multiple user*), masalah keamanan (*security*), masalah kesatuan (*integration*), dan masalah kebebasan data (*data independence*).

Database memiliki sifat secara implisit sebagai berikut (Elmasri & Navathe, 2011):

1. Suatu *database* merepresentasikan beberapa aspek dari dunia nyata, kadang-kadang disebut dengan *miniworld* atau *universe of discourse* (UoD). Perubahan menjadi *miniworld* tersebut tercermin dalam *database*.
2. Suatu *database* secara logika adalah pengumpulan data secara koheren dengan beberapa makna tertentu. Sekumpulan data acak tidak dapat disebut dengan *database*.
3. *Database* didisain, dibangun, dan dibentuk dengan data untuk tujuan tertentu. Hal tersebut berdasarkan dari tujuan sekelompok penggunanya dan beberapa aplikasi tertentu yang sesuai dengan ketertarikan mereka.

2.7 PHP

PHP adalah akronim dari *Hypertext Preprocessor*, yaitu suatu bahasa pemrograman berbasiskan kode-kode yang digunakan untuk mengolah suatu data dan mengirimkannya kembali ke web *browser* menjadi kode HTML (Oktavian, 2010).

PHP adalah skrip bersifat *server-side* yang ditambahkan ke dalam *Hyper Text Markup Language* (HTML). Sifat *server-side* berarti pengerjaan skrip dilakukan di *server*, yang kemudian hasilnya dikirim kembali ke *browser*. Cara penulisan skrip PHP dapat dilakukan dengan 2 teknik, yaitu *Embedded Script* dan *Non embedded Script* (Kustiyaningsih, 2011). Seiring dengan perkembangan teknologi maka lahirnya PHP sebagai bahasa pemrograman open source yang digunakan secara luas terutama untuk pengembangan web dan dapat disimpan dalam bentuk HTML. Sehingga web tidak hanya memberikan informasi tetapi terjalin interaksi dan

menjadikan web bersifat dinamis dan diintegrasikan dengan web server Apache, PWS, dan IIS.

Kelahiran PHP bermula saat Rasmus Lerdorf membuat sejumlah skrip PERL yang dapat mengamati siapa yang yang melihat-lihat daftar riwayat hidupnya pada tahun 1994. Pada tahun 1995, Ramus menciptakan PHP/FI versi 2, dimana versi tersebut dapat menempelkan kode terstruktur dalam tag HTML dan juga PHP dapat digunakan untuk berkomunikasi dengan *database.*

PHP biasanya dipergunakan untuk pemrograman berbasis web yang tidak hanya menampilkan halaman secara statis, namun menampilkan website berbentuk dinamis dimana data diambil dari dalam database. PHP memiliki kelebihan yaitu PHP bersifat sederhana dan memiliki kemampuan untuk menghasilkan berbagai aplikasi web, selain itu PHP juga bersfiat multiplatform seperti Windows, Linux, dan Mac.

2.8 MySQL

MySQL adalah database server relasional yang gratis di bawah lisensi GNU (General Public License). Dengan sifatnya yang open source, memungkinkan user untuk melakukan modifikasi pada *source code*-nya untuk memenuhi kebutuhan spesifik mereka sendiri (Utdirartatmo, 2002). MySQL merupakan *database server multi-user* dan *multi-threaded* yang tangguh yang memungkinkan *backend* yang berbeda, sejumlah program *client* dan *library* yang berbeda, *tool administratif*, dan beberapa antarmuka pemrograman. MySQL juga tersedia sebagai *library* yang bisa digabungkan ke aplikasi.

MySQL juga dapat berperan sebagai *client/server*, dengan kemampuan dapat berjalan baik di OS manapun (multiplatform). MySQL menggunakan bahasa

standar yaitu SQL (*Structured Query Language*) yang merupakan bahasa yang sama dengan *database* lainnya. MySQL lebih sering digunakan bersamaan dengan PHP dalam pengembangan *website* dinamis atau aplikasi web karena kecepatan dan fleksibilitas yang dimiliki oleh MySQL yang tinggi terhadap PHP.

2.9 Konsep Data Flow Diagram

D*ata flow diagram* digunakan untuk menggambarkan arus data yang mengalir di dalam suatu sistem secara keseluruhan (Kendall, 2008). Simbol yang digunakan dalam *data flow diagram*, antara lain:

1. *External entity* (kesatuan luar), merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi, atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan input atau menerima output dari sistem.
2. *Data flow* (arus data), menunjukkan arus dari data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem.
3. *Process* (proses), kegiatan atau kerja yang dilakukan oleh organisasi, mesin atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.
4. *Data store* (simpanan data), merupakan simpanan dari data yang dapat berupa file, arsip, tabel dan lain-lain.

2.10 Konsep Entity Relationship Diagram

Entity Relationship Diagram (ERD) digunakan untuk membantu perancangan konseptual database (Kendall, 2008). Dalam hal ini terdapat tiga macam hubungan antar *entity*, yaitu :

1. *One to one relationship 2 field*, hubungan antara *field* pertama dengan *field* kedua adalah satu berbanding satu.
2. *One to many relationship 2 field*, hubungan antara *field* pertama dengan *field* kedua adalah satu berbanding banyak atau dapat pula sebaliknya.
3. *Many to many relationship 2 field*, hubungan antara *field* pertama dengan *field* kedua adalah banyak berbanding banyak.

2.11 Testing

Menurut Romeo (2003), testing adalah proses pemantapan kepercayaan akan kinerja program atau sistem sebagaimana yang diharapkan. Testing *Software* adalah proses pengoperasikan *software* dalam suatu kondisi yang dikendalikan untuk verifikasi, mendeteksi *error* dan validasi. Verifikasi adalah pengecekkan atau pengetesan entitas-entitas, termasuk *software*, untuk pemenuhan dan konsistensi dengan melakukan evaluasi hasil terhadap kebutuhan yang telah ditetapkan. Validasi adalah melihat kebenaran sistem apakah proses yang telah dituliskan sudah sesuai dengan apa yang dibutuhkan oleh pengguna. Deteksi *error* adalah testing yang berorentasi untuk membuat kesalahan secara intensif, untuk menentukan apakah suatu hal tersebut tidak terjadi. *Test case* merupakan suatu tes yang dilakukan berdasarkan pada suatu inisialisasi, masukan, kondisi ataupun hasil yang telah ditentukan sebelumnya.

White box testing adalah suatu metode desain *test case* yang menggunakan struktur kendali dari desain prosedural. Seringkali *white box testing* diasosiasikan dengan pengukuran cakupan tes, yang mengukur persentase jalur-jalir dari tipe yang dipilih untuk dieksekusi oleh *test cases*. *White box testing* dapat menjamin semua struktur *internal* data dapat dites untuk memastikan validitasnya.

Black box testing dilakukan tanpa adanya suatu pengetahuan tentang detail struktur internal dari sistem atau komponen yang dites, juga disebut sebagai *functional testing*. *Black box testing* bergfokus pada kebutuhan fungsional pada *software*, berdasarkan pada spesifikasi kebutuhan dari *software*.