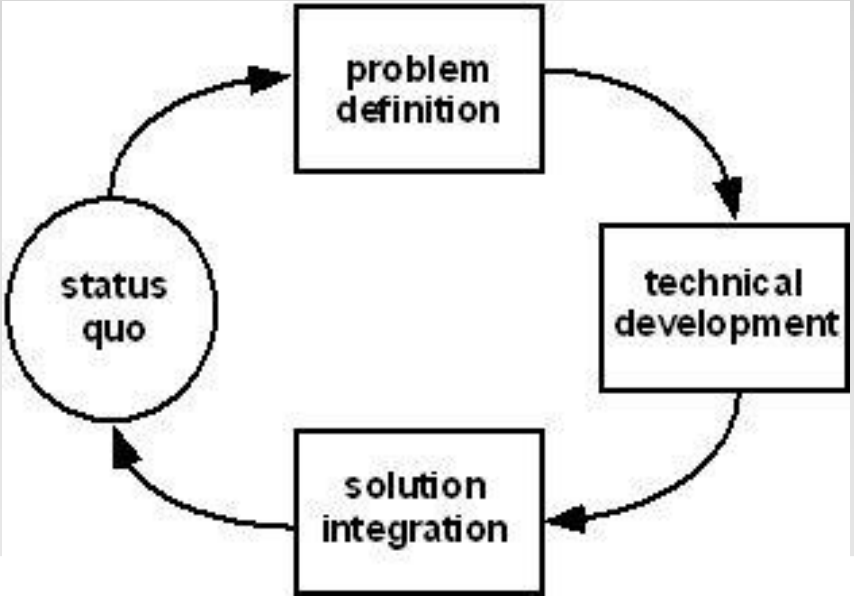
PENGEMBANGAN PERANGKAT LUNAK

- pengembangan perangkat lunak (PL) dapat dianggap sebagai lingkaran pemecahan masalah. Untuk menyelesaikan masalah besar, dipecah menjadi kecil terus-menerus sampai paling kecil, kemudian diselesaikan (*recursive*).

- Model proses / daur hidup PL : model proses untuk Rekayasa Perangkat Lunak (RPL) dipilih berdasarkan sifat aplikasi dan proyeknya, metode dan *tool* yang digunakan, serta kontrol *dan deliverable* yang diinginkan.

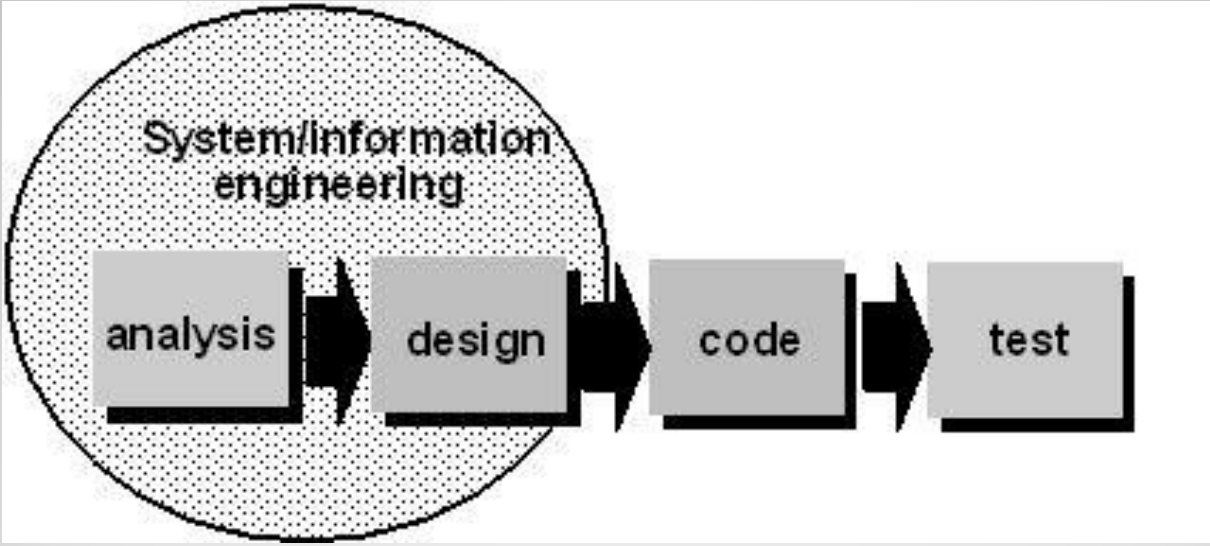


Metode Pengembangan PL

- Metode Systems Development Life Cycle, atau SDLC (Daur hidup pengembangan sistem) adalah proses yang digunakan oleh analis sistem untuk mengembangkan sistem , mulai dari penentuan kebutuhan, perancangan, validasi, sampai pelatihan dan penyerahan kepada konsumen.

- SDLC merupakan alur kerja baku yang biasa dipakai oleh perusahaan-perusahaan vendor software dalam mengembangkan software aplikasi produksinya. SDLC ini tidak hanya penting untuk proses produksi software saja, namun terlebih juga sangat penting untuk proses maintenance software itu sendiri,

- karena tanpa pengarsipan data-data development suatu software, maka akan sangat menyulitkan perusahaan dalam maintenance software tersebut dikemudian hari.



- ***System / information engineering*** **(rekayasa dan pemodelan sistem)** : menyangkut pengumpulan kebutuhan (*requirement gathering*) pada level sistem dengan sejumlah kecil analisis serta top desain.
- **Analisis** : kebutuhan PL, proses *requirement gathering* diintensifkan dan difokuskan, khususnya pada PL. Untuk memahami sifat program yang dibangun, analis harus memahami domain informasi, tingkah laku, unjuk kerja, dan *interface* yang diperlukan. Kebutuhan sistem maupun PL didokumentasikan dan di*review* bersama user.

- **Desain** : fokus pada 4 hal : desain *database*, arsitektur PL, *interface*, dan algoritma prosedural. Proses desain menerjemahkan kebutuhan ke dalam representasi PL sebelum dimulai *coding*.

- ***Coding*** : menerjemahkan desain ke dalam bahasa yang dimengerti mesin.

- ***Testing*** : fokus pada :
  Logika internal PL : memastikan bahwa semua *statement* telah diuji (white box)
  Fungsi eksternal : mengarahkan *testing* untuk menemukan kesalahan-kesalahan dan memastikan bahwa input yang diberikan akan menghasilkan output sesuai yang diinginkan (black box).
- ***Maintenance***

Kelemahan model :

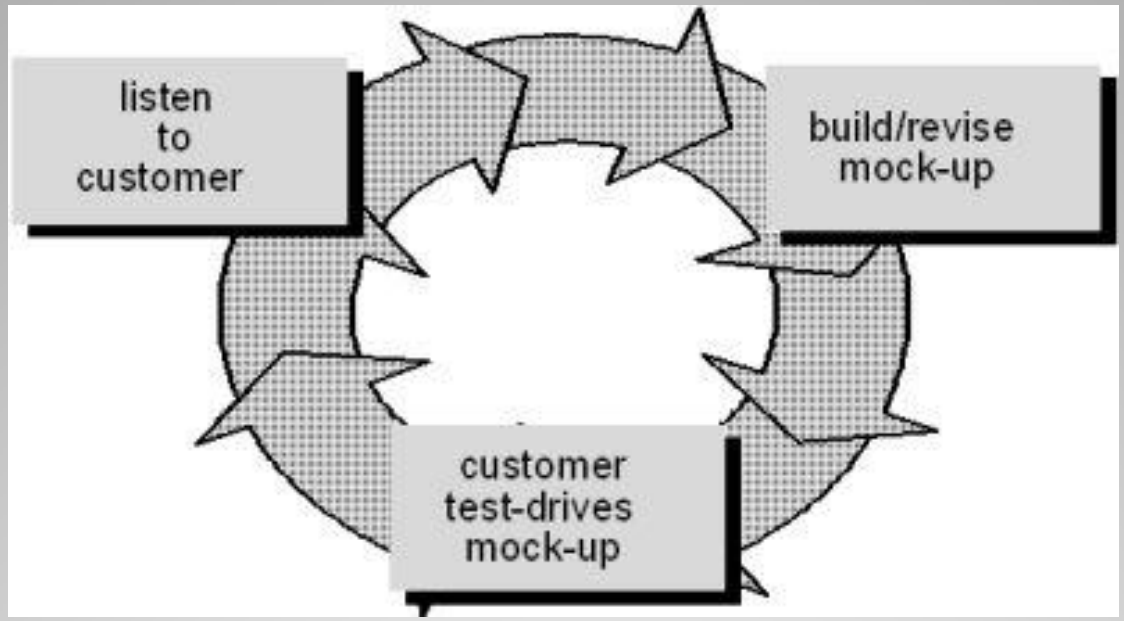
- Meskipun mengakomodasi iterasi, model linier melakukannya secara tidak langsung sehingga perubahan-perubahan dapat menyebabkan keraguan saat tim proyek berjalan
- Jika user sulit menyatakan semua kebutuhannya secara explisit, model ini sulit mengakomodasi ketidakpastian itu

- User harus bersabar karena hasil baru bisa dinikmati setelah *testing* (akhir waktu). Jika ada kesalahan besar yang tidak terdeteksi sampai program yang bekerja tersebut dikaji ulang, bisa menjadi petaka.
- Pengembang sering melakukan penundaan yang tidak perlu. Anggota tim harus menunggu anggota tim lain selesai mengerjakan tugasnya yang mempunyai keterkaitan dan ketergantungan tinggi.

- Meski memiliki kelemahan, model ini masih lebih baik daripada pendekatan yang sembrono / sembarangan.
- Contoh : **model Waterfall** terdiri dari fase investigasi -> analisis -> desain -> implementasi (*coding*) -> testing -> maintenance.

Prototyping Model

- Berfungsi sebagai mekanisme pendefinisian kebutuhan. Pertama, *developer* menggali semua kebutuhan user secara cepat kemudian membangun *prototipe* yang sesuai dengan yang diinginkan dengan cepat pula dan ditunjukkan ke user, baru dibuat PL yang sesungguhnya berdasarkan komentar user terhadap *prototipe*.



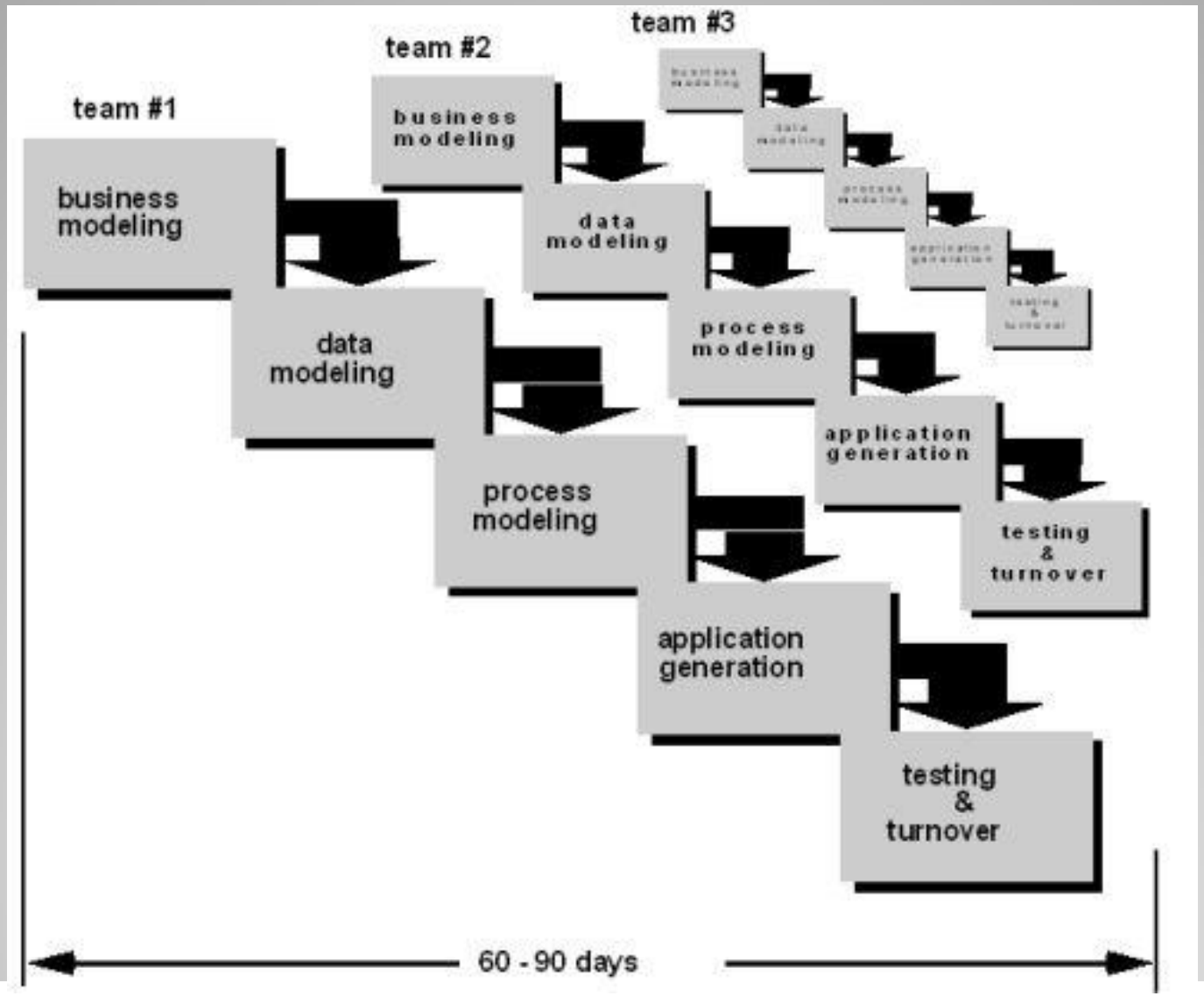
Kelebihan model prototipe : user dapat langsung melihat wujud PL yang akan dibangun meskipun sederhana dan dari sana dapat digali kebutuhan yang lebih dalam sebagai bahan penyusunan PL berikutnya.

Masalah yg muncul :

- User merasa *prototipe* merupakan PL yang sesungguhnya, padahal ketika membuat *prototipe* belum disertakan kualitas PL secara keseluruhan / kemampuan pemeliharaan untuk jangka panjang
- *Developer* sering membuat kompromi-kompromi implementasi untuk membuat *prototipe* bekerja dengan cepat sehingga akan ditemui ketidakcocokan pada *prototipe* ketika *prototipe* dibangun dengan bahasa yang sederhana
- Program dibuat ulang / *prototipe* selalu baru

Rapid Application Development (RAD) Model

- Model proses perkembangan PL sekuensial linier yang menekankan siklus perkembangan yang sangat pendek. Menekankan perkembangan komponen program yang bisa dipakai lagi sehingga mendasari konsep *Object-Oriented*.


team #1
business modeling
data modeling
process modeling
application generation
testing & turnover
team #2
business modeling
data modeling
process modeling
application generation
testing & turnover
team #3
60 - 90 days

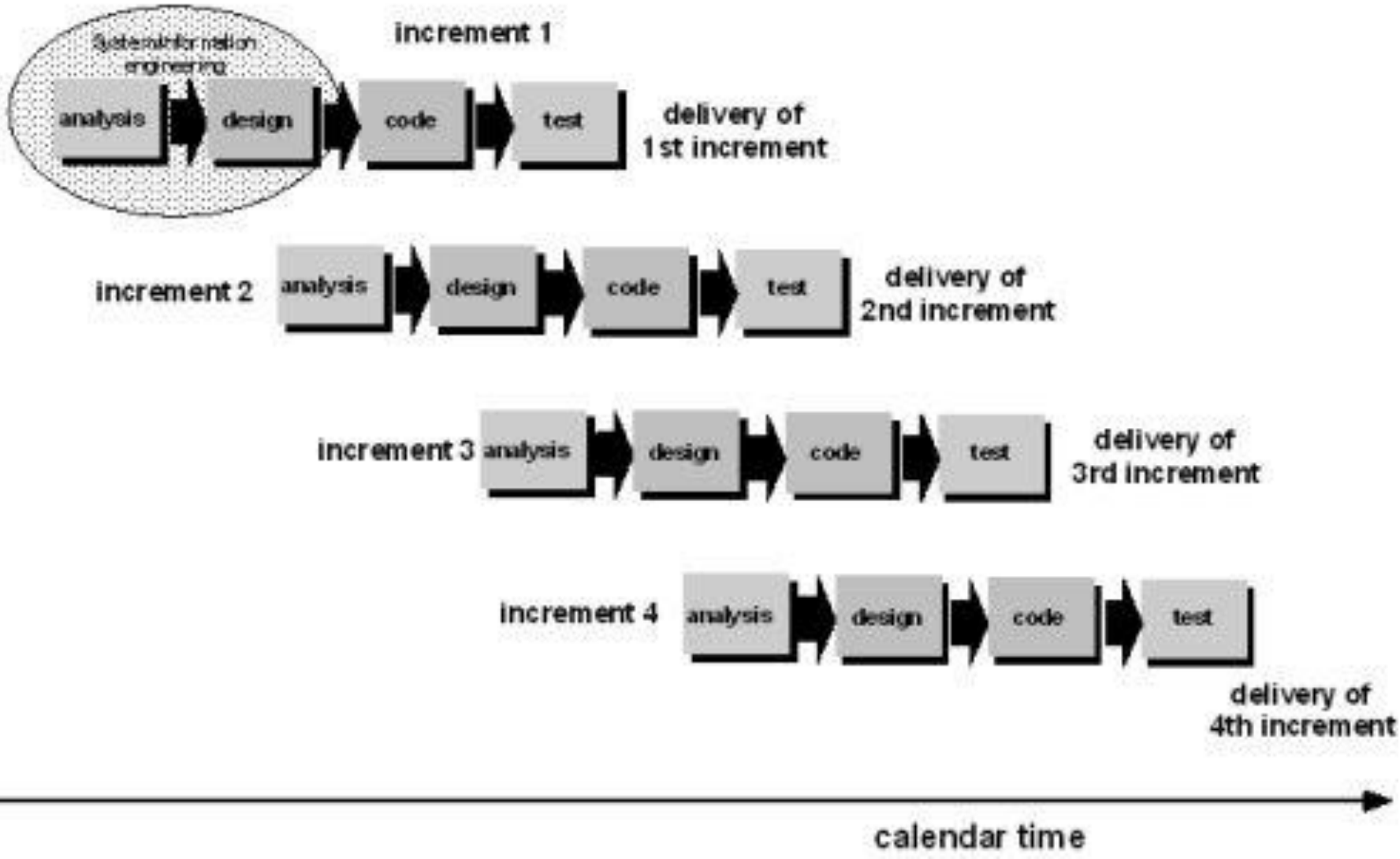
Fase2 pendekatan RAD :

- ***Bussines modeling***
- ***Data modeling***
- ***Proses modeling***
- ***Application generation*** : RAD mengasumsikan pemakaian teknik 4G (generasi keempat). Selain menciptakan PL dengan bahasa pemrograman generasi ketiga yang konvensional, RAD lebih banyak memproses kerja untuk memakai lagi komponen program atau menciptakan komponen yang bias dipakai lagi.

- ***Testing and Turn Over*** : karena menekankan pada *reusability*, banyak komponen program yang telah diuji sehingga mengurangi keseluruhan waktu pengujian. Tapi komponen baru harus diuji dan semua *interface* harus dilatih secara penuh.

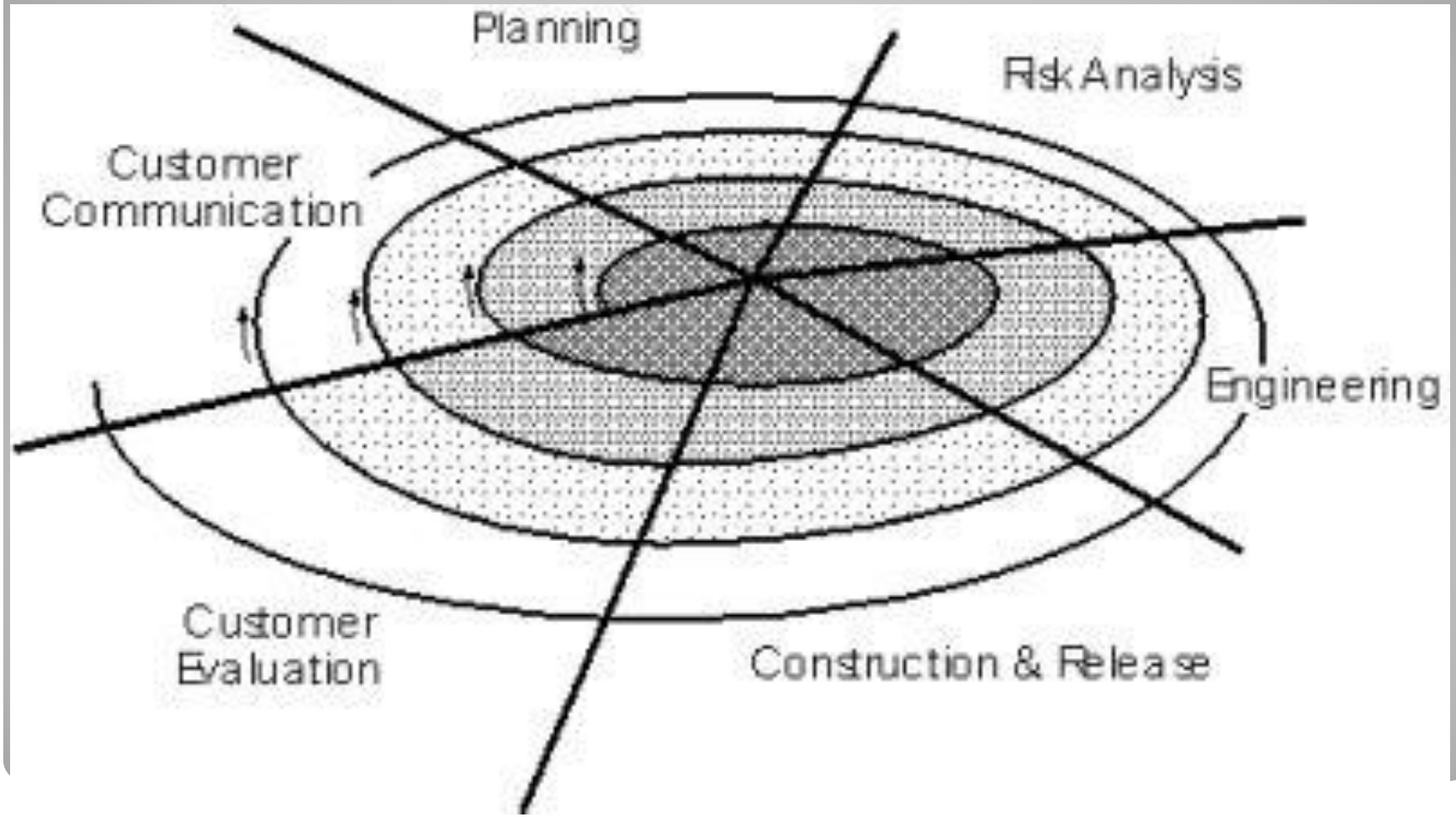
The Incremental Model

- Mengadopsi model sekuensial linier dan model *prototipe*. Fungsi dasar sama, tapi ada tambahan asesoris (contoh : pada M.Word 1997, 2000). Fungsi tambahan ditambahkan terus untuk membuat sistem menjadi lebih baik. Pada *increment* pertama PL yang jadi, mengakomodasi kebutuhan inti. Baru pada tahap berikutnya ditambahkan kemampuan baru.
- Contoh : pengembangan *microsoft word*.
- Increment 1 : hanya memberi fungsi inti –> hanya bisa mengetik saja
- Increment 2 : bisa *word art*, *spelling*, dll
- Kelebihan model : cocok untuk produksi masal.


System/information engineering
increment 1
analysis
design
code
test
delivery of 1st increment
increment 2
analysis
design
code
test
delivery of 2nd increment
increment 3
analysis
design
code
test
delivery of 3rd increment
increment 4
analysis
design
code
test
delivery of 4th increment
calendar time

- Penomoran dimulai dari arsiran paling dalam. Proses yang terjadi :
- **Proyek pengembangan konsep**
- **Proyek pengembangan produk baru**
- **Proyek perbaikan produk**
- **Proyek pemeliharaan produk**

Evolutionary (Spiral) Model



- Semua proyek tersebut dibangun dengan mengikuti model spiral melalui tahap :
- ***Customer communication***
- ***Planning***
- ***Risk analysis***
- ***Engineering Construction and release***
- ***Customer evaluation***

Latihan kasus

- Anda ditunjuk sebagai manejer yang mendapatkan proyek sebuah perangkat lunak pengolah database yang sangat sederhana, tetapi waktu yang ditentukan oleh pelanggan sangat ketat .

  Pertanyaan :

a. model pengembangan/proses PL mana yang anda pilih ?Mengapa ??.

- Anda dipilih menjadi seorang manajer proyek pada sebuah organisasi sistem informasi yang mendapat proyek membangun sebuah aplikasi yang sangat mirip dengan aplikasi lain yang sudah pernah dibangun sebelumnya, meskipun yang satu ini lebih besar dan kompleks, syarat-syaratnya sudah didokumentasikan dengan teliti oleh pemakai.

- Pertanyaan :
  model pengembangan/proses PL mana yang anda pilih ? Mengapa ??.