KONSEP PEMODELAN DENGAN UML

KONSEP PEMODELAN

- Pada pengembangan sistem model digambarkan dalam bentuk fisik dan abstrak.
- Tim pengembangan sistem (software) membutuhkan model untuk mendapaat gambaran tentang sistem tersebut.
- Meskipun s/w dibangun oleh 1 orang model tetap diperlukan karena pengembangan sistem adalah kegiatan yang kompleks.

Apakah Model Itu?

Beberapa pemahaman tentang model

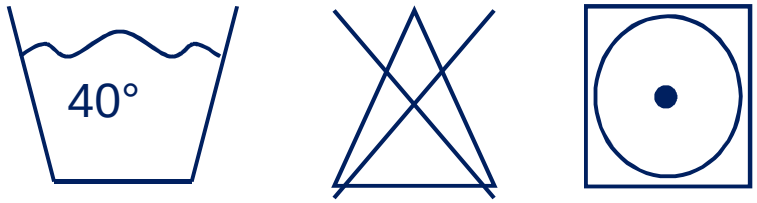
- Sebuah model harus cepat dan mudah untuk dibangun
- Sebuah model bisa digunakan untuk simulasi, mempelajari mengenai sesuatu yang akan direpresentasikan
- Sebuah model mampu mempelajari perkembangan dari suatu kegiatan atau masalah
- Kita bisa memilih secara rinci sebuah model
- Model bisa merepresentasikan sesuatu secara real atau tidak sebuah domain.

Apakah Diagram Itu?

- Analis dan perancang sistem menggunakan diagram untuk membuat model sebuah sistem.
- Fungsi diagram biasanya digunakan oleh analis dan designer untuk :
  - Mengkomunikasikan ide-ide
  - Mengenerate ide baru serta segala kemungkinan
  - Melakukan tes terhadap ide serta membuat prediksi
  - Memperlajari struktur dan hubungan suatu sistem

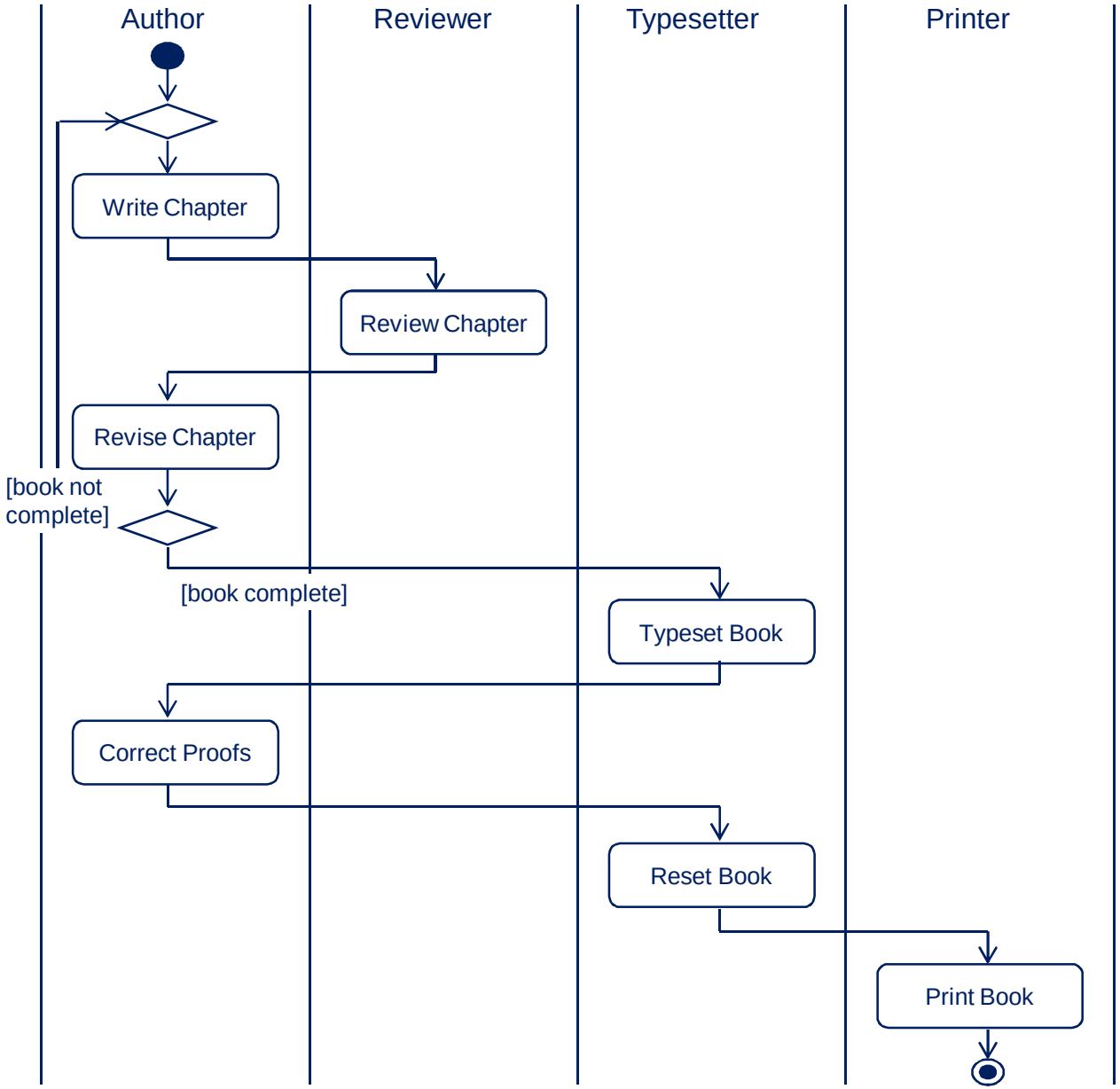
Apakah Diagram Itu?

- Diagram mengikuti aturan atau standar.
- Contoh Diagram sederhana:

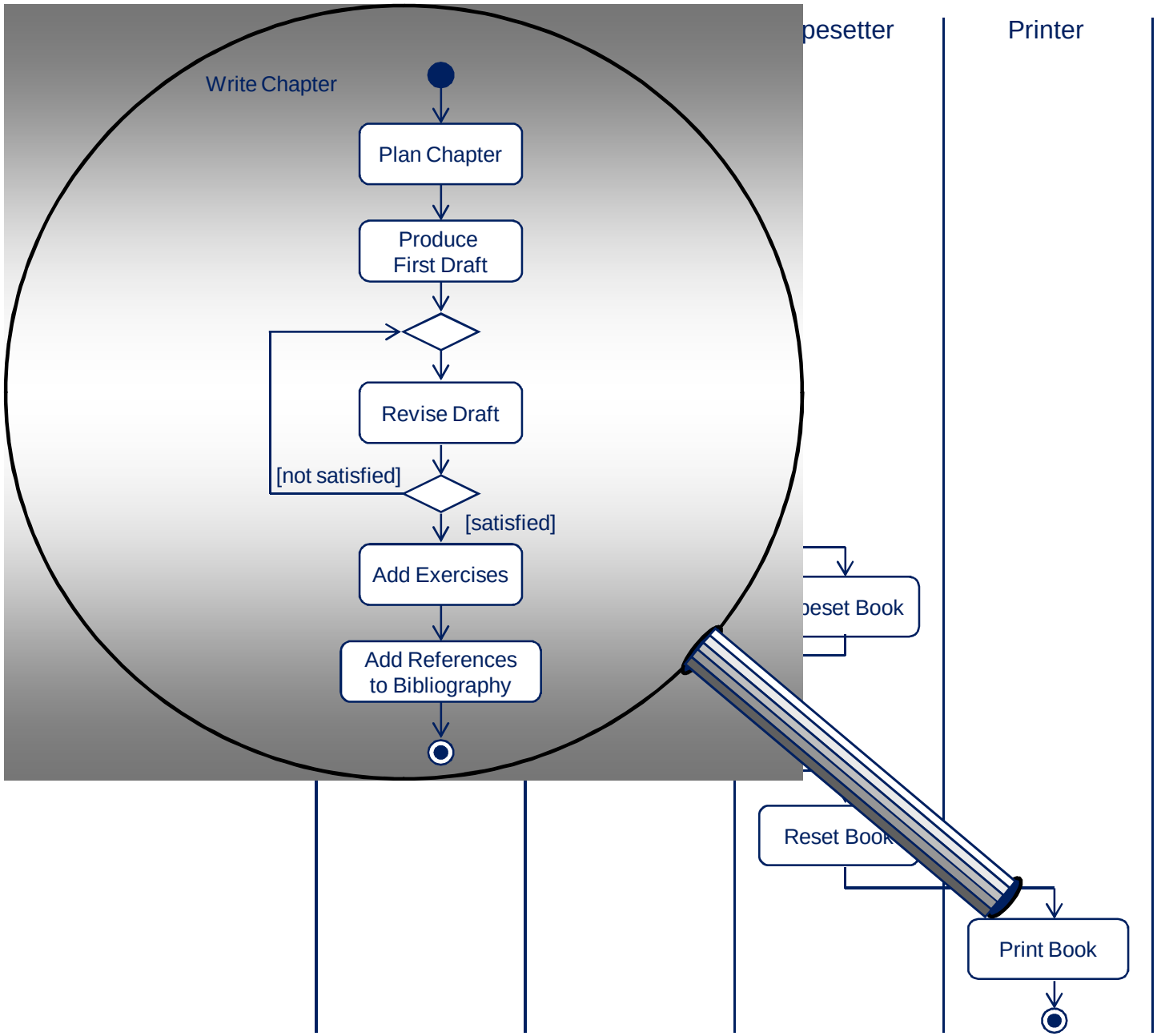


Apakah Diagram Itu?

- Contoh activity diagram untuk membuat buku:



Activity Diagram dengan detail yang tersembunyi


Write Chapter
Plan Chapter
Produce First Draft
Revise Draft
[not satisfied]
[satisfied]
Add Exercises
Add References to Bibliography
pesetter
Printer
beset Book
Reset Book
Print Book

Acuan dalam Merancang Sebuah Model

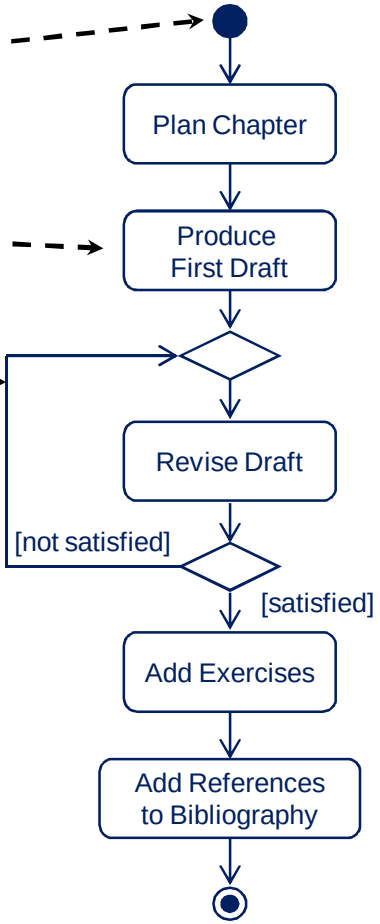
- **Simplicity representation** – hanya menggambarkan apa yang harus ditampilkan
- **Internal consistency** – pada sekumpulan diagram
- **Complentness** – menampilkan semua yang dibutuhkan
- **Hierarchical representation** – dapat diturunkan untuk melihat lebih detail pada level yang lebih rendah.

UML (Unified Modeling Language)

- UML banyak digunakan sebagai model standar dalam mengembangkan sebuah sistem informasi.
- UML memilik 4 elemen utama, yaitu :
  - Icons
  - Simbol dua dimensi (Two Dimensional)
  - Paths
  - String

UML (Unified Modeling Language)

- icons
- two-dimensional symbols
- paths
- Strings



Model vs. Diagram

(OMG, 2004b) mendefiniskan model sebagai berikut:

> “sebuah model menangkap kebutuhan sistem secara fisik. Merupakan abstraksi dari sistem secara fisik. Tujuannya adalah memasukkan apa yang harus dimasukkan dalam sistem dan seperti apa hubungannya. Model yang lengkap menggambarkan segala aspek sistem secara fisik pada tahapan yang lebih rinci.”

Model vs. Diagram

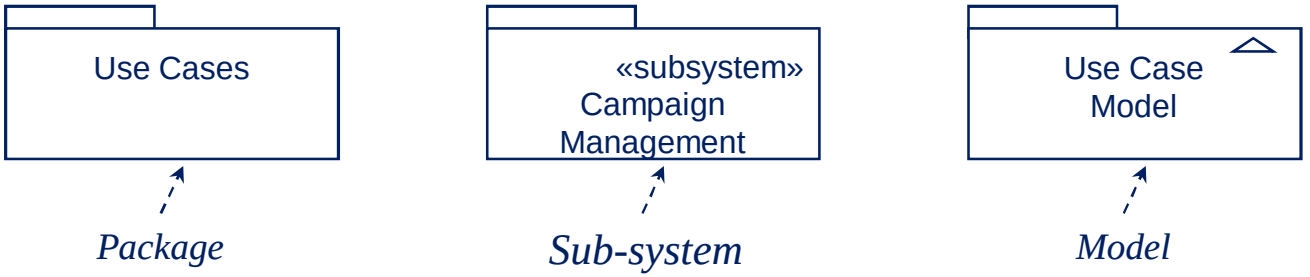
- Pada UML ada sekumpulan konsep yang digunakan untuk menggambarkan sistem serta bagaimana cara untuk membuat modelnya .
- Sistem adalah segala sesuatu yang akan dibuat modelnya.
- Sebuah subsistem adalah bagian dari sistem yang terdiri atas elemen-elemen pembentuk subsistem dan sistem tersebut.
- Sebuah model adalah abstraksi dari sistem dan subsistem pada sudut pandang tertentu.

Model vs. Diagram

- Diagram adalah representasi grafis dari sekumpulan elemen dalam model sebuah sistem.
- Sebuah model adalah abstraksi dari sistem dan subsistem pada sudut pandang tertentu.
- Model yang berbeda menggambarkan sudut pandang yang berbeda dari suatu sistem.
- Ada 5 sudut pandang yang digunakan dalam membuat model dengan UML yaitu : usecase view, design view, process view, implementation view, deployment view.
- UML menyediakan notasi untuk menggambarkan subsitem dalam bentuk *packages.*

Membangun Model

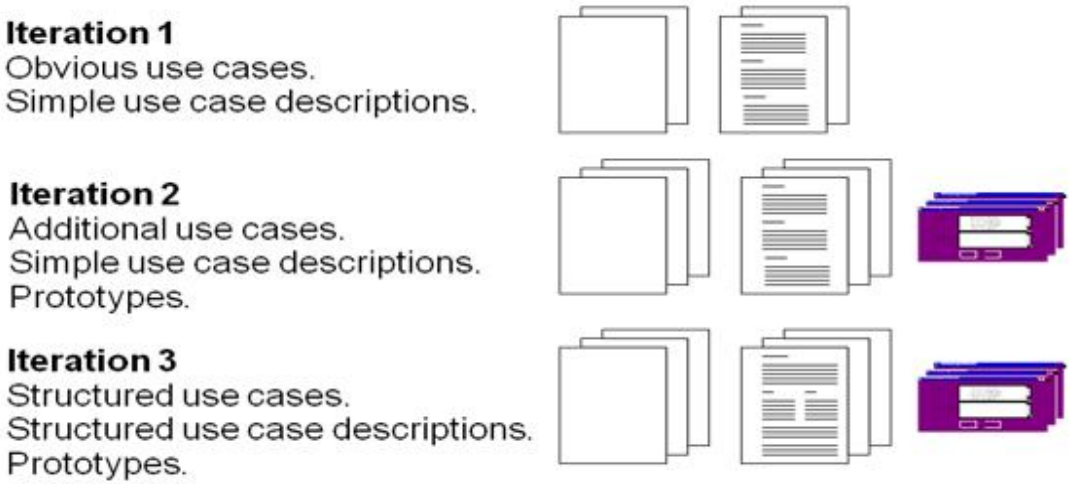
- Model yang dihasilkan dalam mengembangkan sistem selalu mengalami perubahan sesuai perkembangan proyek. Perubahan tersebut meliputi tiga dimensi utama, antara lain :
  - Abstraction(Abstraksi)
  - Formality(Formalitas)
  - Level of detail



Notasi UML untuk Package, SubSystem dan Model

Membangun Model

- Dalam proyek pengembangan sistem yang menggunakan siklus hidup iteratif, model yang berbeda mewakili pandangan yang sama dapat dikembangkan pada tingkat detail berbeda.

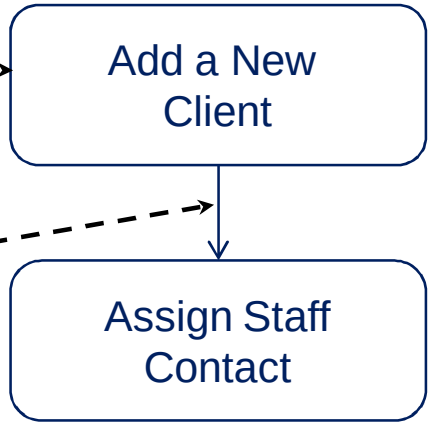


Activity Diagram

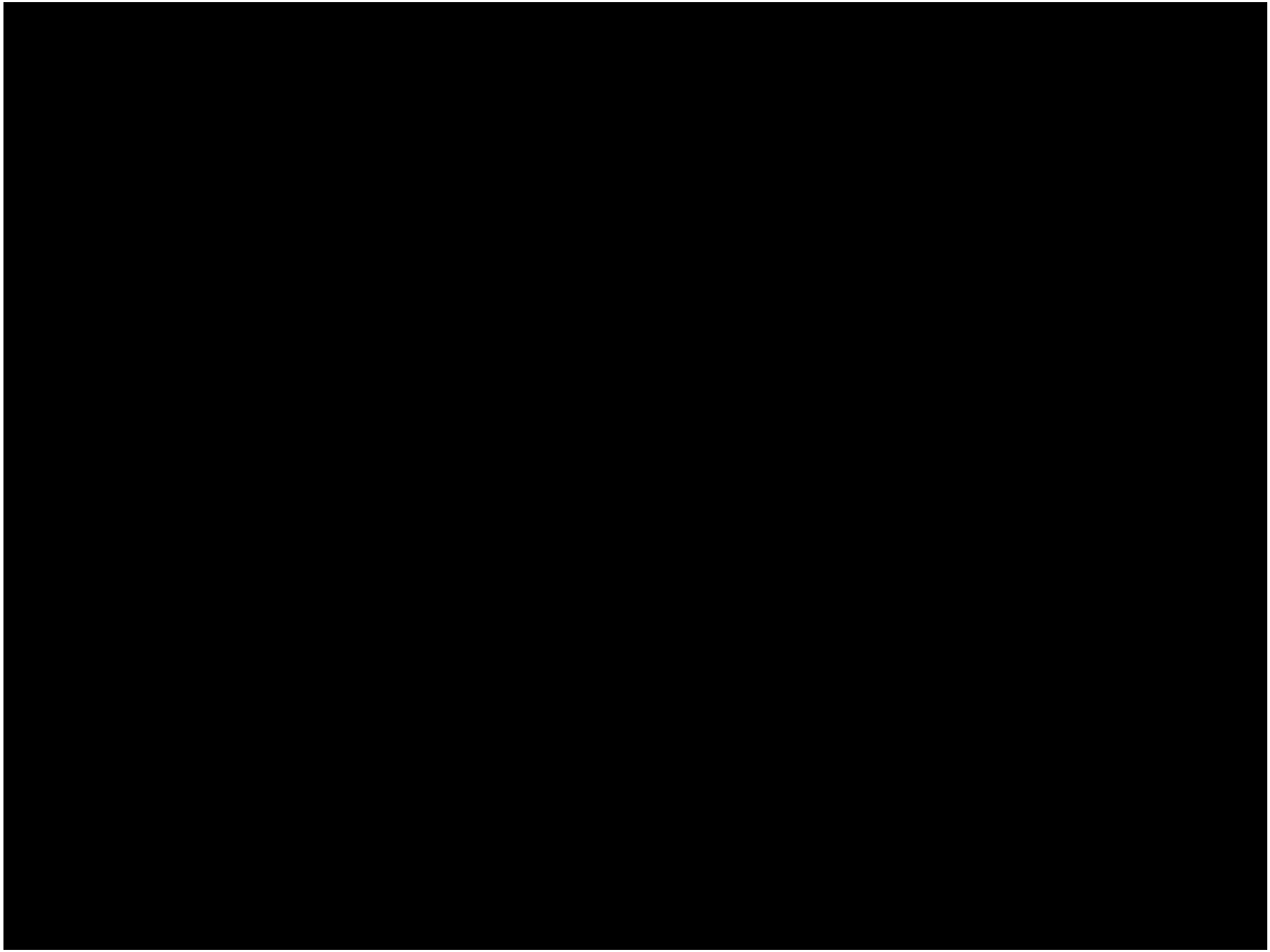
- Digunakan untuk memodelkan beberapa aspek dari sistem.
- Pada level yang lebih tinggi digunakan untuk memodelkan aktivitas bisnis yang ada atau potensial pada sistem.
- Umumnya activity diagram digunakan untuk beberapa tujuan, antara lain :
  - Memodelkan proses atau task
  - Mengambarkan fungsi sistem yang direpresentasikan oleh usecase
  - Pada spesifikasi operasional digunakan untuk menggambarkan logika operasi
  - Pada USDP (Unified Software Development Process) untuk memodelkan aktifitas yang membentuk siklus hidup (lifecycle)

Notasi pada Activity Diagram

- Actions
  - rectangle with rounded corners
  - meaningful name
- Control flows
  - arrows with open arrowheads

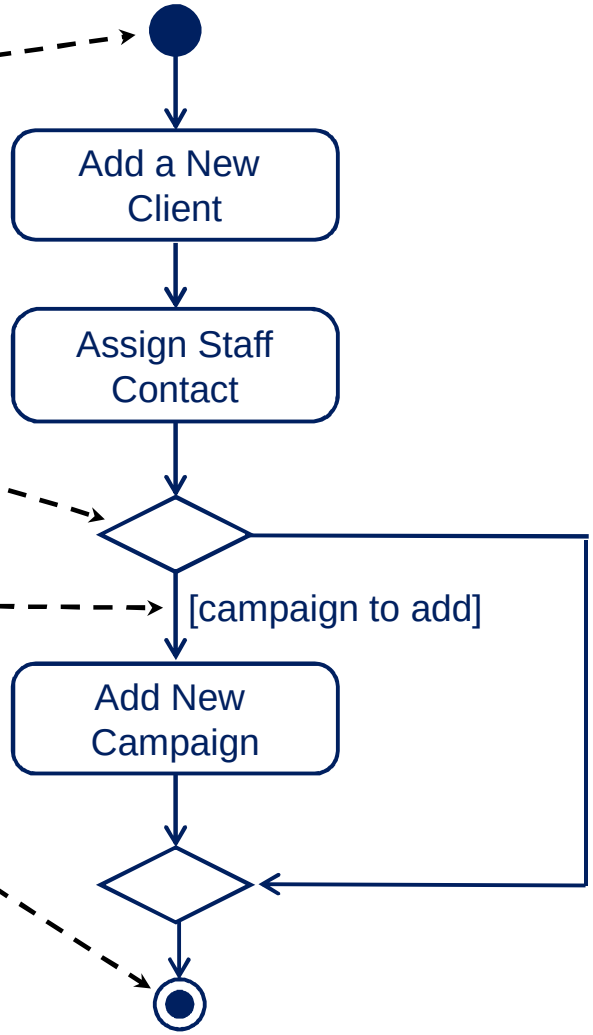


Perhatikan Video tentang Activity Diagram sebagai pengantar materi . Klik Video pada slide halaman berikutnya.



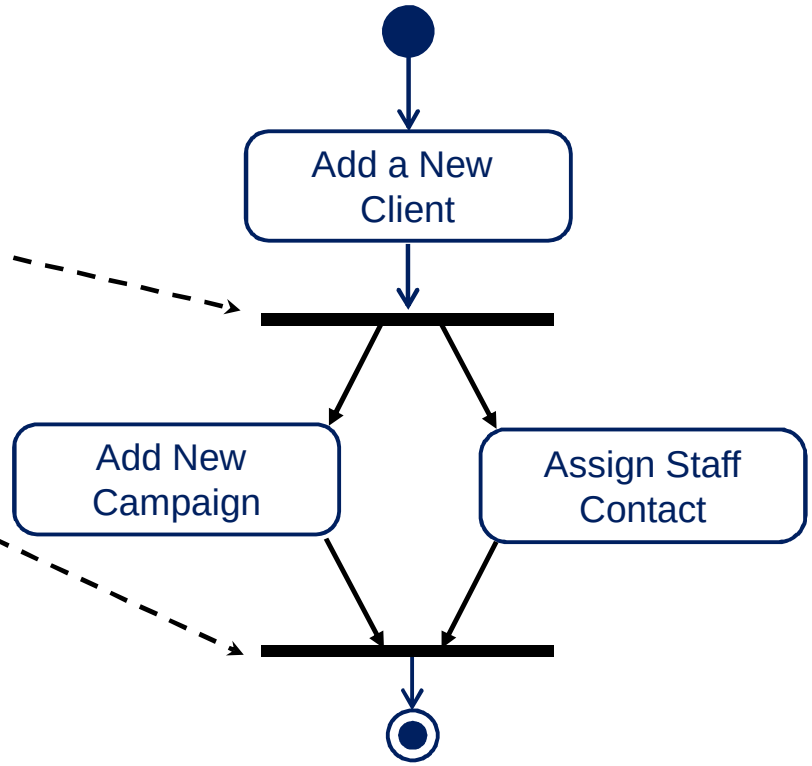
Notasi pada Activity Diagram

- Initial node
  - black circle
- Decision nodes (and merge nodes)
  - diamond
- Guard conditions
  - in square brackets
- Final node
  - black circle in white circle



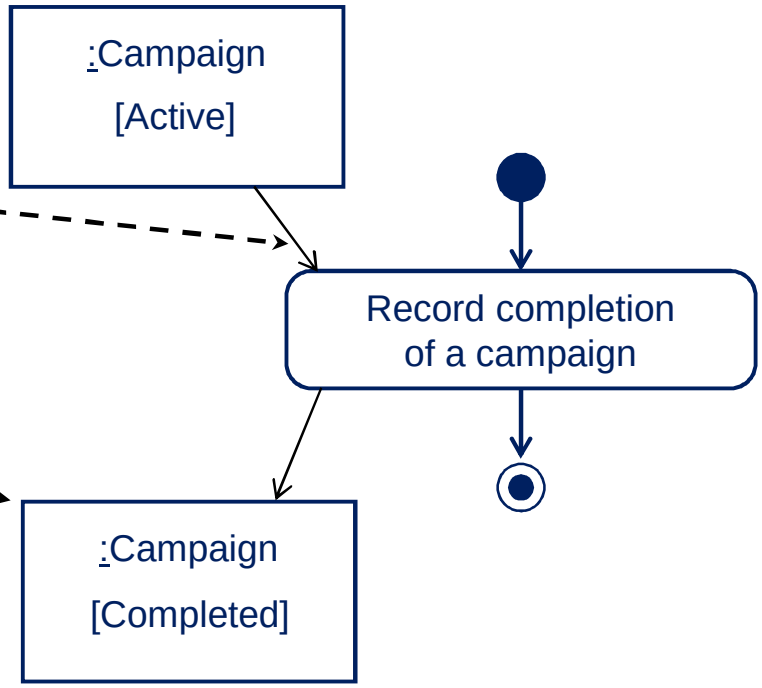
Notasi pada Activity Diagram

- Fork nodes and join nodes
  - thick bar
- Actions carried out in parallel



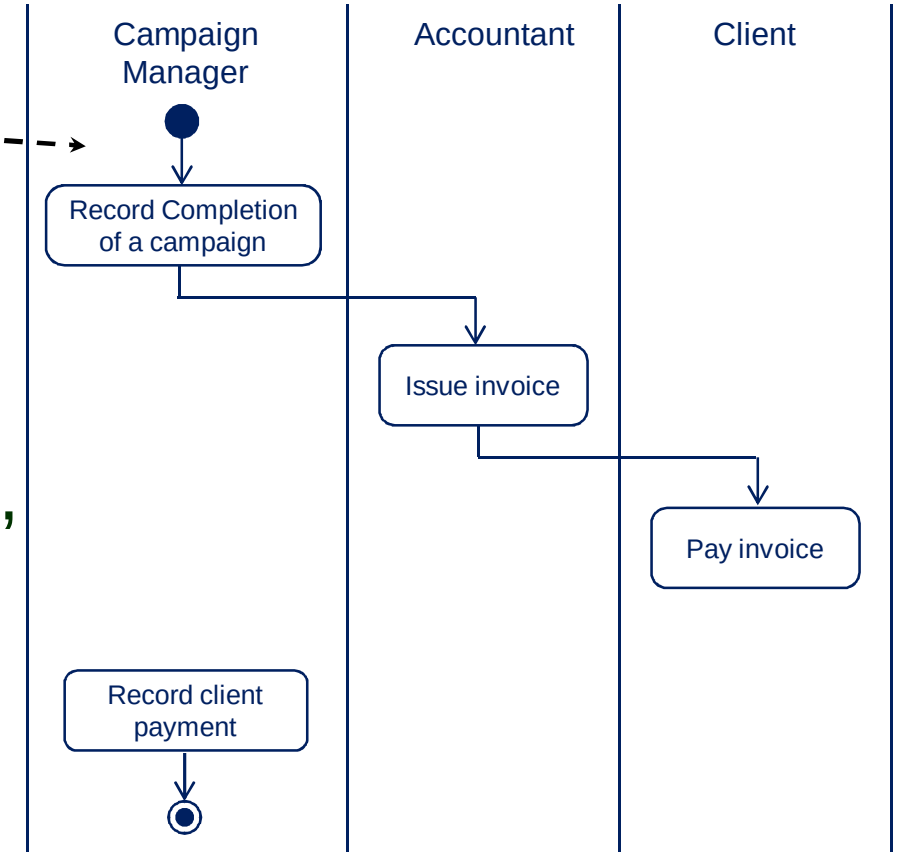
Notasi pada Activity Diagram

- Object flows
  - open arrow
- Objects
  - rectangle
  - optionally shows the state of the object in square brackets



Notasi pada Activity Diagram

- Activity Partitions (Swimlanes)
  - vertical columns
  - labelled with the person, organisation, department or system responsible for the activities in that column

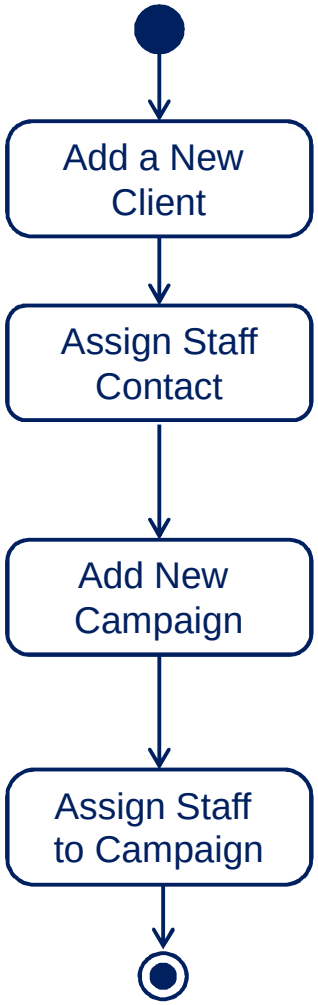


Contoh Kasus: **System Agate**

Langkah-langkah :

1. Tentukan Tujuan, pada sistem Agate, tujuannya adalah untuk menampilkan beberapa aktifitas yang ada pada sistem tersebut.
2. Apa saja yang akan ditampilkan pada diagram, dalam hal ini nama dari proses bisnis, usecase atau operasinya
3. Sampai level detail mana proses tersebut dibutuhkan apakah hanya global saja atau lebih rinci
4. Identifikasi setiap action/aksi, pada sistem Agate aksi yang dikerjakan adalah:
   - Add a New Client
   - Assign Staff Contact
   - Add New Campaign
   - Assign Staff to Campaign
5. Organisasikan setiap aksi dalam bentuk aliran data

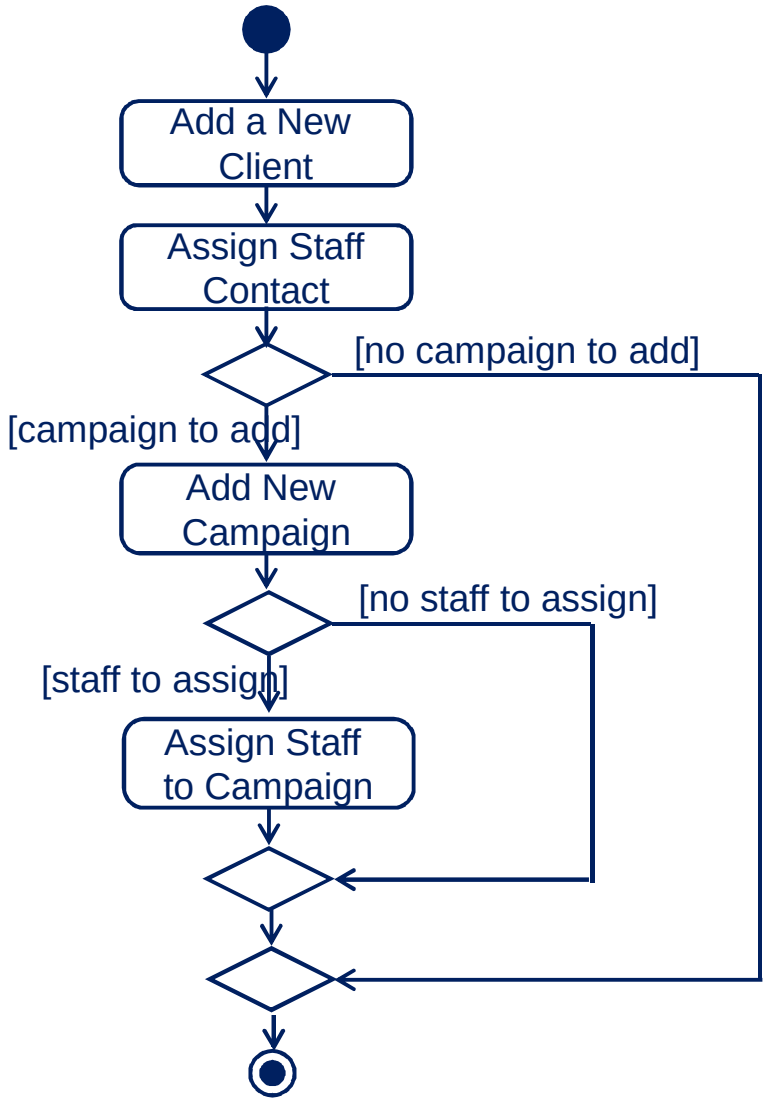
Contoh Kasus: **System Agate**



Contoh Kasus: **System Agate**

Langkah-langkah :

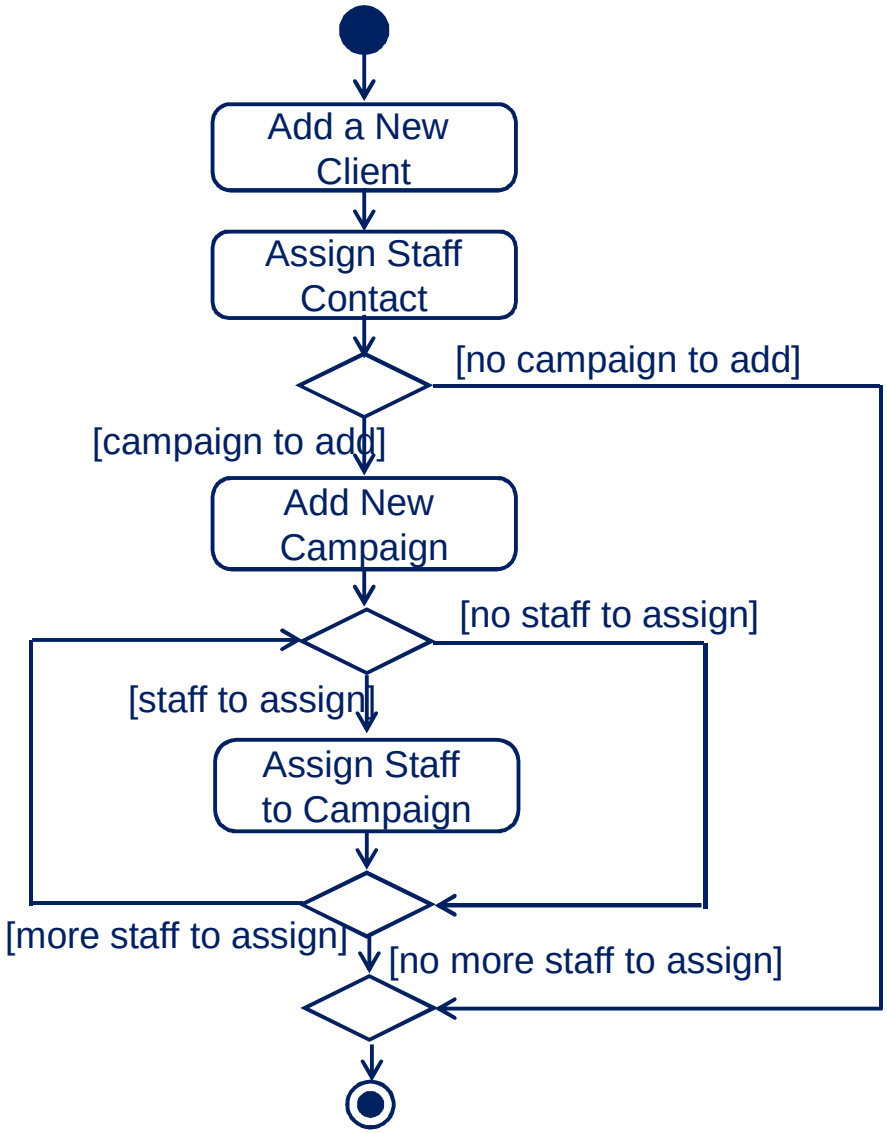
6. Identifikasi altenatif aliran data untuk setiap kondisi
7. Tambahkan node decision jika diperlukan



Contoh Kasus: **System Agate**

Langkah-langkah :

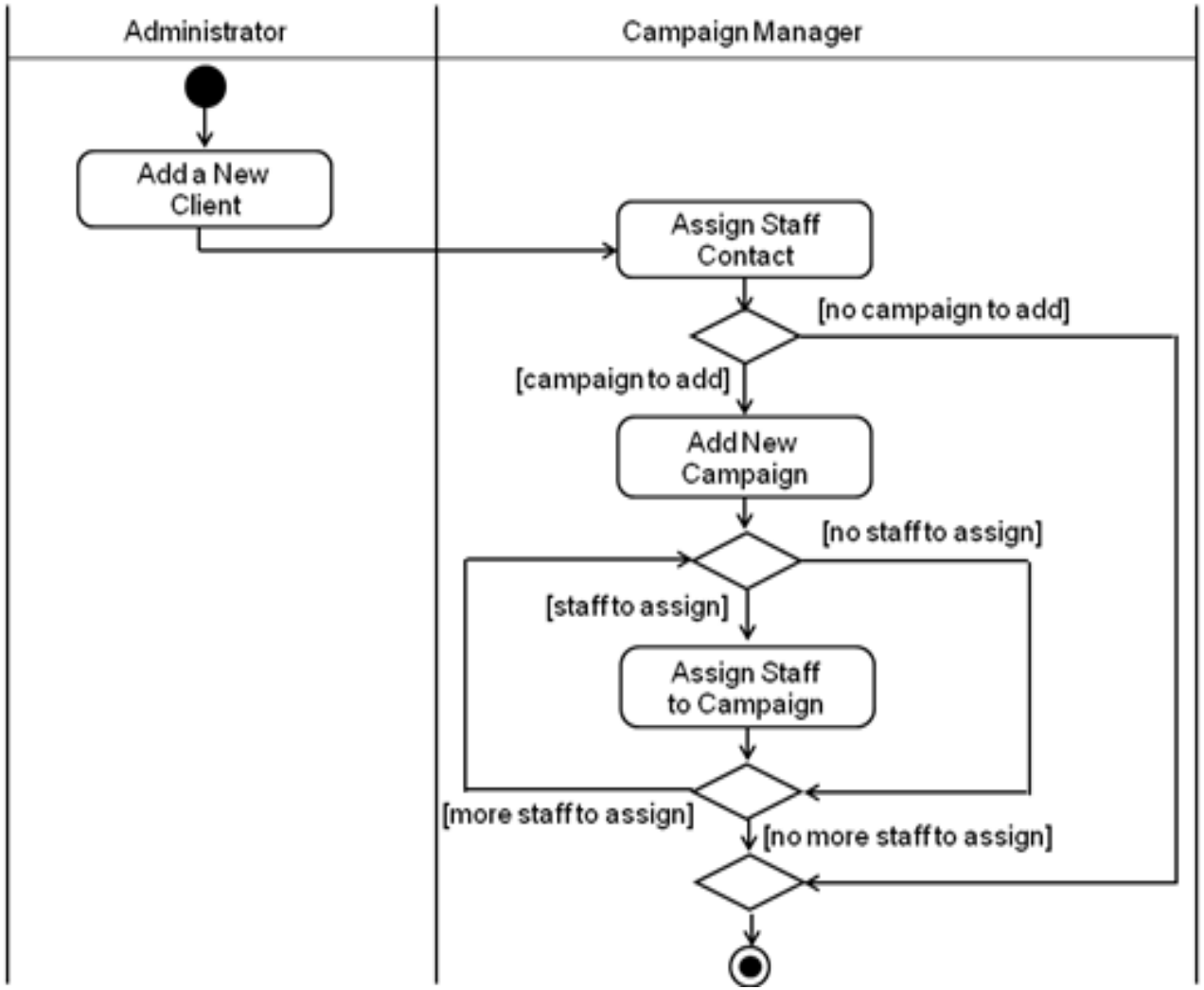
8. Identifikasi aksi yang berjalan secara pararel
9. Tambahkan notasi fork atau Join jika diperlukan
10. Identifikasi proses yang berulang



Contoh Kasus: **System Agate**

Langkah-langkah :

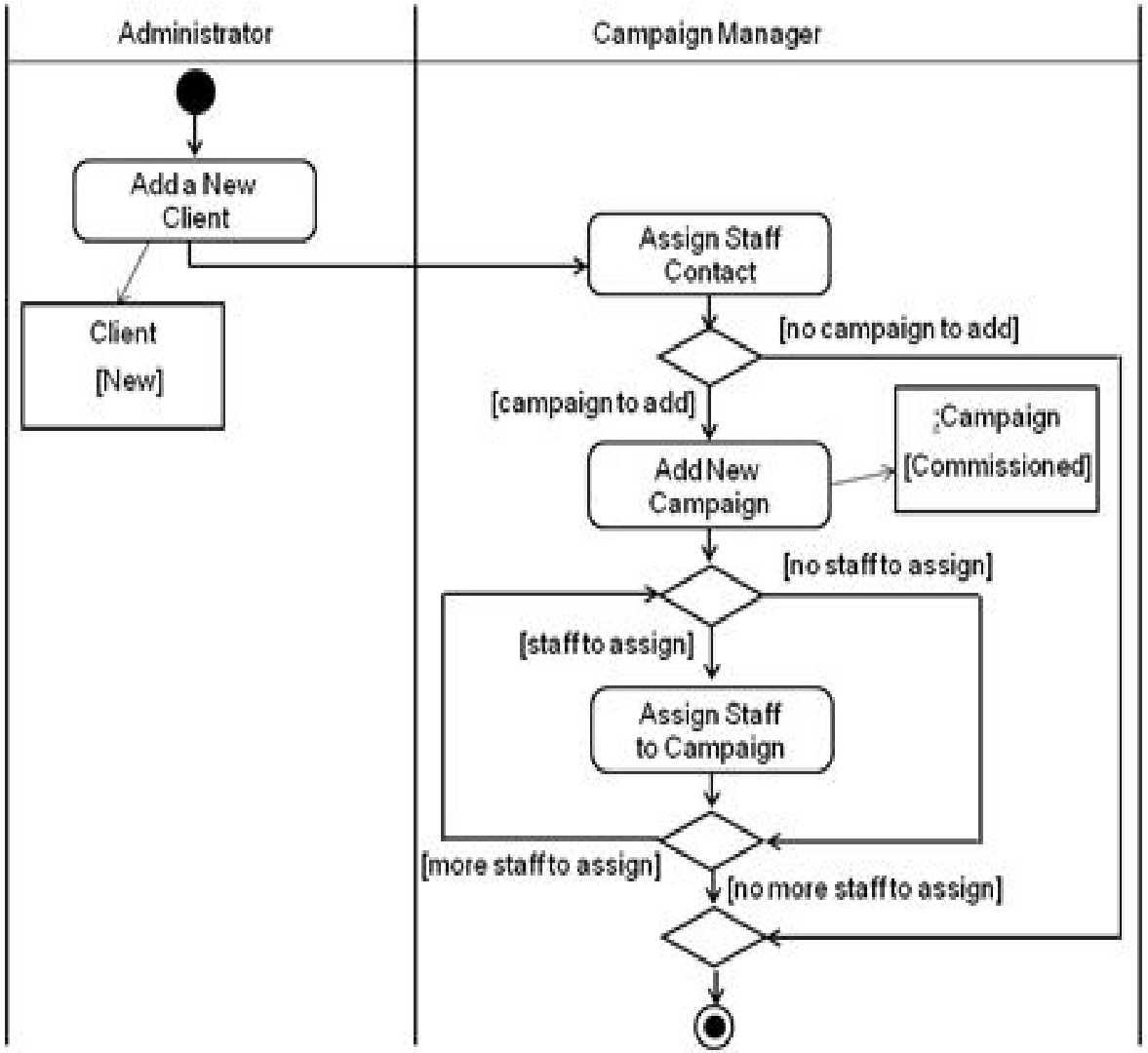
11. Tambahkan swimline untuk mengidentifikasi siapa yang melakukan aktifitas.



Contoh Kasus: **System Agate**

Langkah-langkah:

12. Masukkan object Flow dan object yang diperlukan



Tinjauan UML

Sejarah UML (Unified Modelling Language)

- Muncul pada mid 1970s dan 1980s.
- Berbagai metodologi digunakan untuk melakukan analisa dan perancangan.
- Jumlah pemodelan diidentifikasi meningkat selama periode antara 1989-1994.
- Banyak pengguna metode berorientasi object mengalami kesulitan menemukan satu
  Pemodelan → memicu "perang metode."
- Mid 1990s, iterasi baru dari metode ini mulai muncul dan metode ini mulai menggabungkan teknik masing-masing dari beberapa tokoh.

Tinjauan UML

- Akhir 1994 Grady Booch dan Jim Rumbaugh dari Rational Software Corporation mempersatukan model Booch dan OMT (Object Modeling Technique).
- 1995, IvarJacobson dan perusahaan Objectory nya menggabungkan Rasional dalam upaya unifikasi, penggabungan ini membentuk metode OOSE (Object-Oriented Software Engineering) .
- Sebagai penulis utama dari Booch, OMT, dan metode OOSE, Grady Booch, Jim Rumbaugh , dan Ivar Jacobson termotivasi untuk menciptakan sebuah bahasa pemodelan terpadu yang pada akhirnya pada oktober 1996 mumcullah UML versi 0.9
- Selama tahun 1996 UML mulai dilirik sebagai bagian dari OMG (Object Management Group) dan mulai january 1997 mulai dimasukkan dalam RFP (A Request for Proposal) sebagai bagian dari OMG.

Artifak UML

Terdapat beberapa artifak utama dalam UML, yaitu :

- Use Case Diagram, diagram yang menggambarkan actor, use case dan relasinya
- Class Diagram, diagram untuk menggambarkan kelas dan relasi diantara kelas-kelas tersebut
- Behaviour Diagram, yang terdiri dari :
  - Activity Diagram, menggambarkan aktifitas-aktifitas, objek, state, transisi state dan event
  - Collaboration Diagram, menggambarkan objek dan relasinya, termasuk struktur perubahannya yang disebabkan oleh adanya suatu message

Artifak UML

- Sequence Diagram, menggambarkan objek dan relasinya termasuk kronologi (urutan) perubahan secara logis setelah menerima sebuah message
- Statechart Diagram, menggambarkan state, transisi state dan event

- Implementation Diagram, terdiri dari :
  - Component Diagram, menggambarkan komponen dan relasi antara komponen tersebut
  - Deployment Diagram, menggambarkan komponen, titik awal dan relasi antara komponen tersebut