

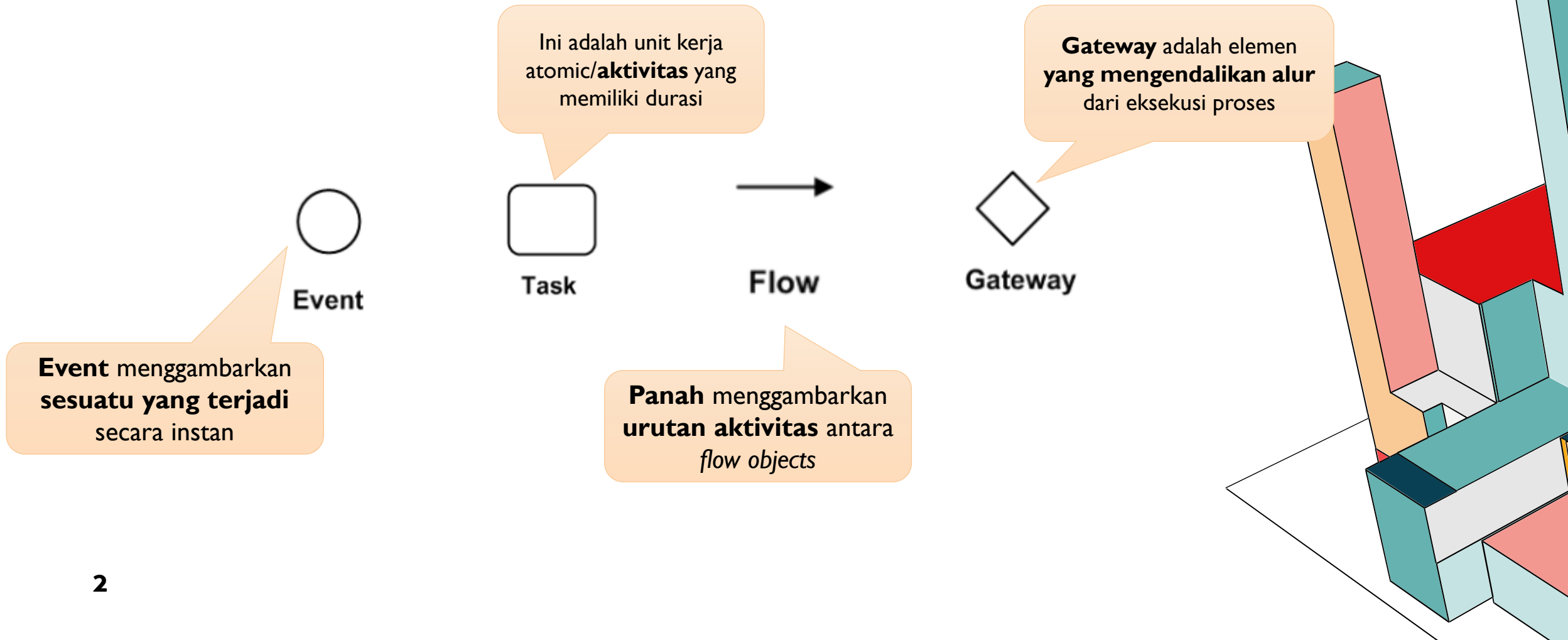


# **BASIC CONCEPTS OF BPMN**

Fahmi Yusuf, S.Kom, MMSI, Ph.D

# BPMN BASIC ELEMENTS

4 Basic Elements, yang biasanya memenuhi 80% kebutuhan pemodelan



# MENGHUBUNGKAN ANTAR AKTIVITAS

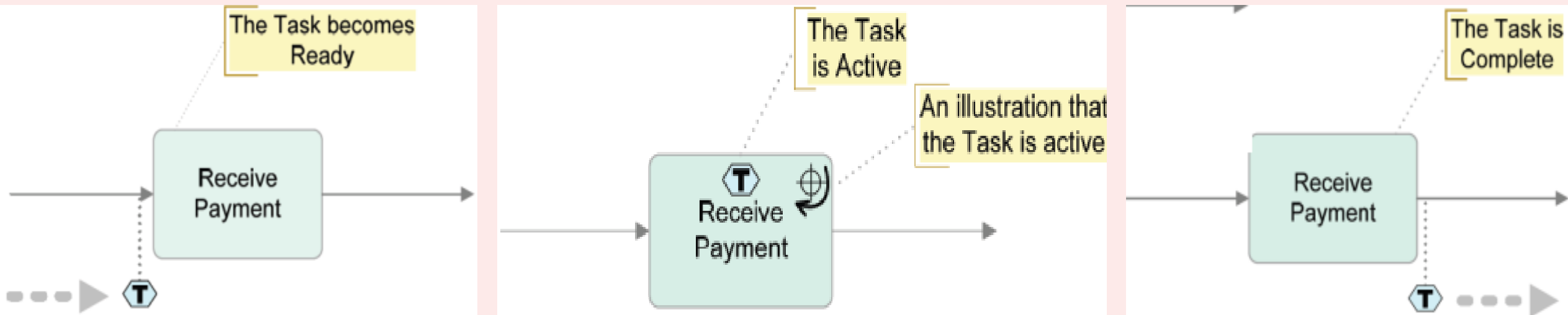
- **Sequence Flow (Tanda Panah)** menggambarkan urutan dari **Flow Objects** dalam sebuah proses. Setiap aktivitas dapat memiliki satu atau lebih sequence flow yang masuk dan juga satu atau lebih sequence flow yang keluar.



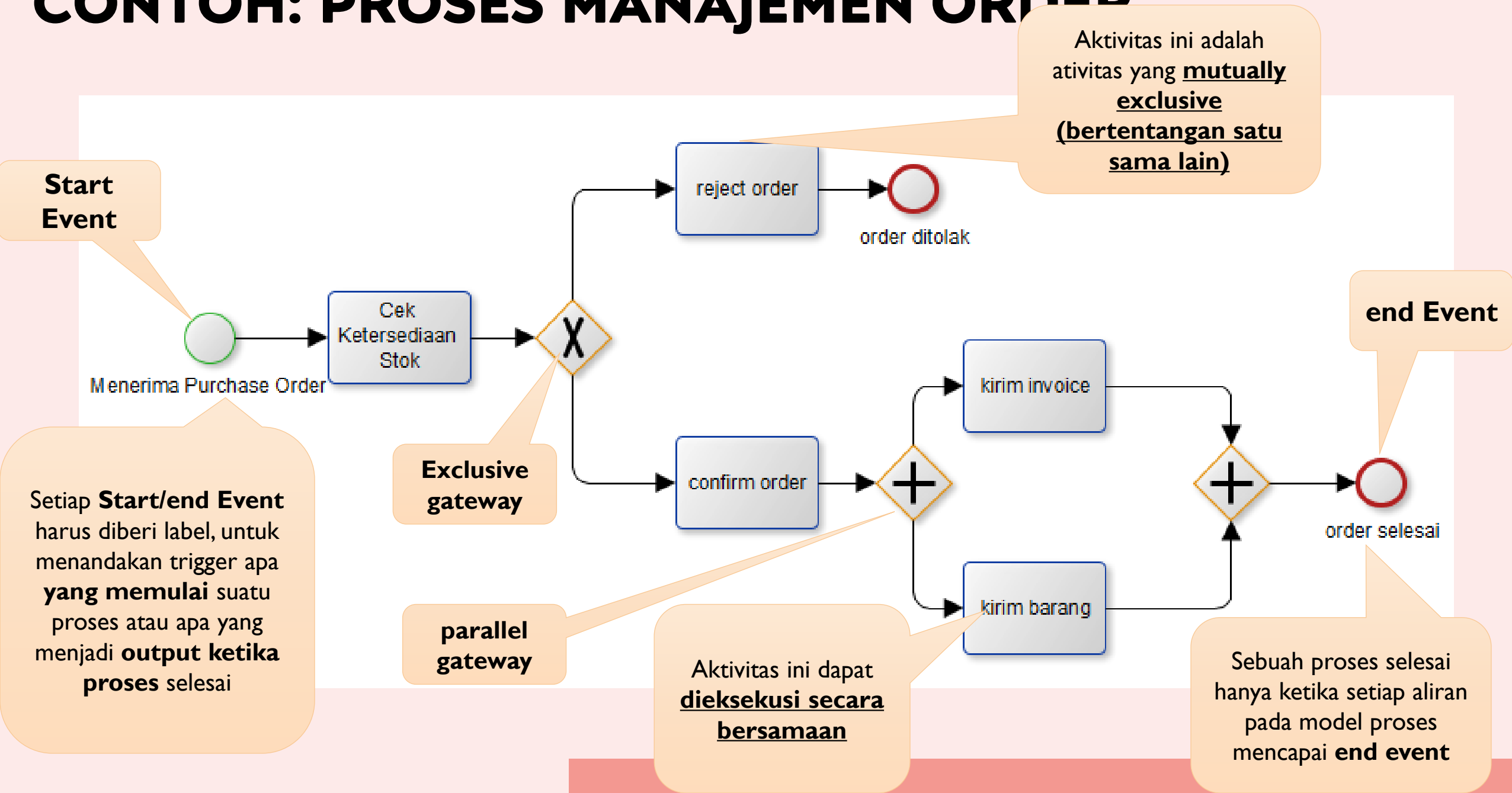
- Namun umumnya, sebuah aktivitas memiliki satu sequence flow yang masuk dan satu sequence flow yang keluar
- Setiap proses harus memiliki minimal satu Start Event (lingkaran dengan satu garis tipis), yang menunjukkan kapan dan dimana **proses dapat dimulai**, dan sebuah end event (lingkaran dengan border garis tebal/dobel), untuk menunjukkan kapan sebuah **proses dinyatakan selesai**.

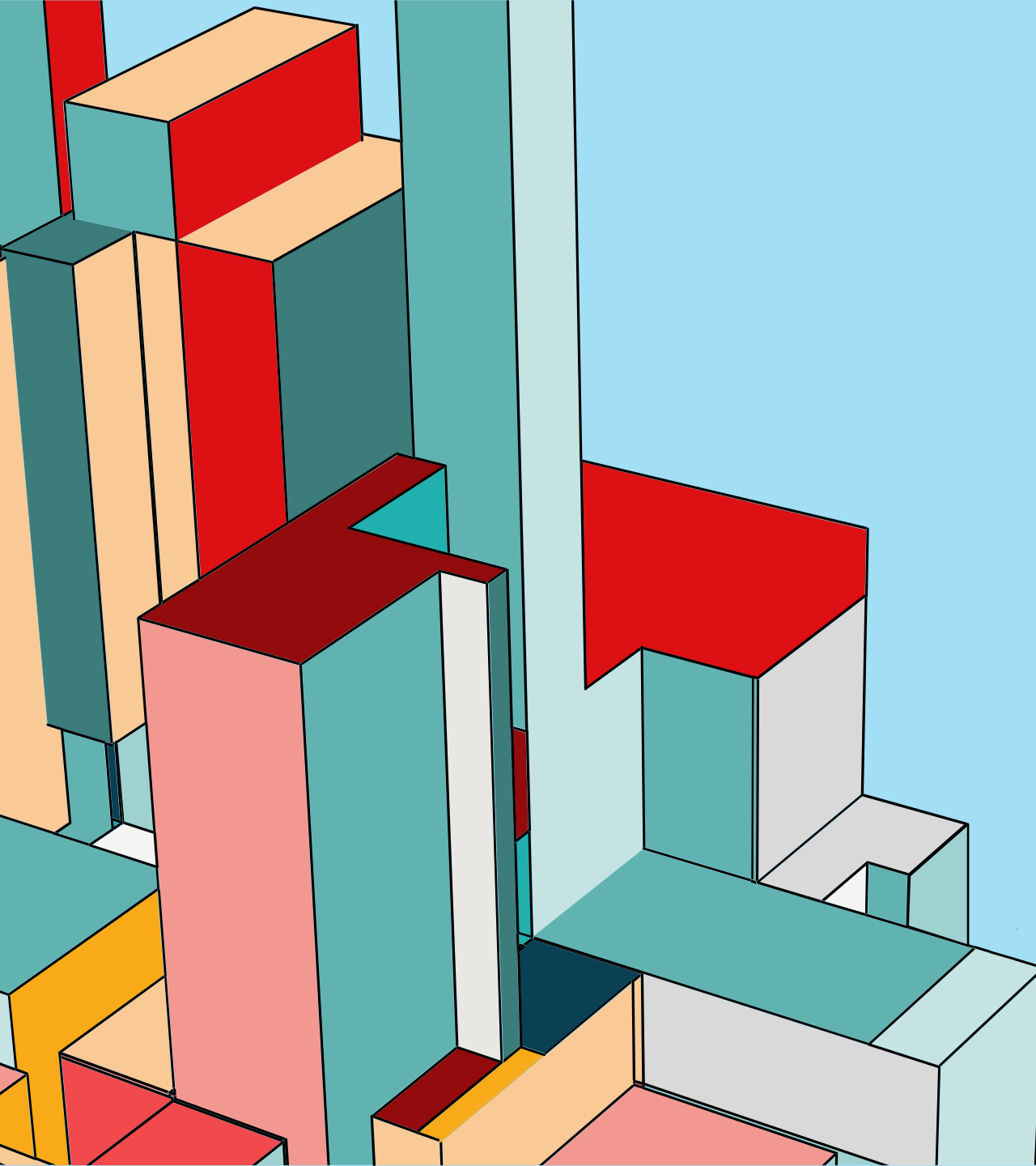
# ACTIVITY BEHAVIOUR

- Ketika proses dimulai (pada start event), kita menggunakan istilah token untuk menandakan progress (atau “status”) dari proses
- Token adalah objek teoritis yang digunakan untuk membuat “simulasi” deskriptif dari perilaku yang terkait dengan setiap elemen BPMN (namun token bukan bagian formal dari elemen BPMN)
- Token di-*create* pada start event, melintasi sequence flow dan dihancurkan pada end event. Artinya, tidak ada waktu yang terkait dengan token yang menyusuri *sequence flow*



# CONTOH: PROSES MANAJEMEN ORDER





# **SEDIKIT LEBIH LANJUT TENTANG GATEWAY**

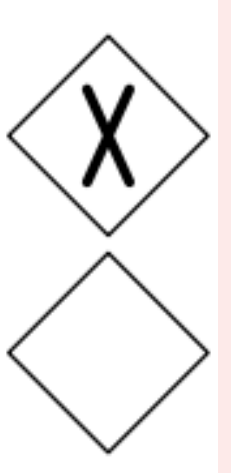
# SELANJUTNYA TENTANG GATEWAY

- Sebuah **gateway** menunjukkan bahwa ada mekanisme ‘gerbang’ yang dapat mengizinkan atau tidak mengizinkan sebuah token pesan melewati **gateway**
- Ketika token tiba di **gateway**, mereka dapat digabungkan menjadi satu, atau dipisah sebagai outputnya tergantung pada tipe gateway .
- **Split gateway** menggambarkan titik dimana proses dipecah, sedangkan **join gateway** menggambarkan titik dimana alur proses digabung.
  - **SPLIT** memiliki satu **incoming sequence flow** dan **multiple outgoing sequence flow** (menggambarkan bahwa pada percabangan terjadi percabangan)
  - **JOIN** memiliki **multiple incoming sequence flow** (menggambarkan cabang-cabang yang akan digabung) dan satu **outgoing sequence flow**

# TIPE GATEWAY

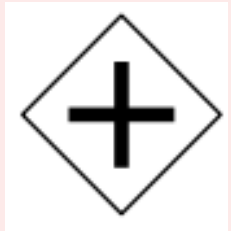
- Exclusive Gateways (XOR)

- Menunjukkan lokasi di dalam proses bisnis dimana *sequence flow* dapat mengambil dua atau lebih jalur alternative
- HANYA SATU jalur yang dapat diambil
- Disimbolkan dengan bentuk *diamond* yang dapat memiliki tanda seperti huruf “X”
- Kita menggunakan *XOR-Join* untuk menggabungkan dua atau lebih cabang alternative yang mungkin sebelumnya dipecah oleh *XOR-Split*



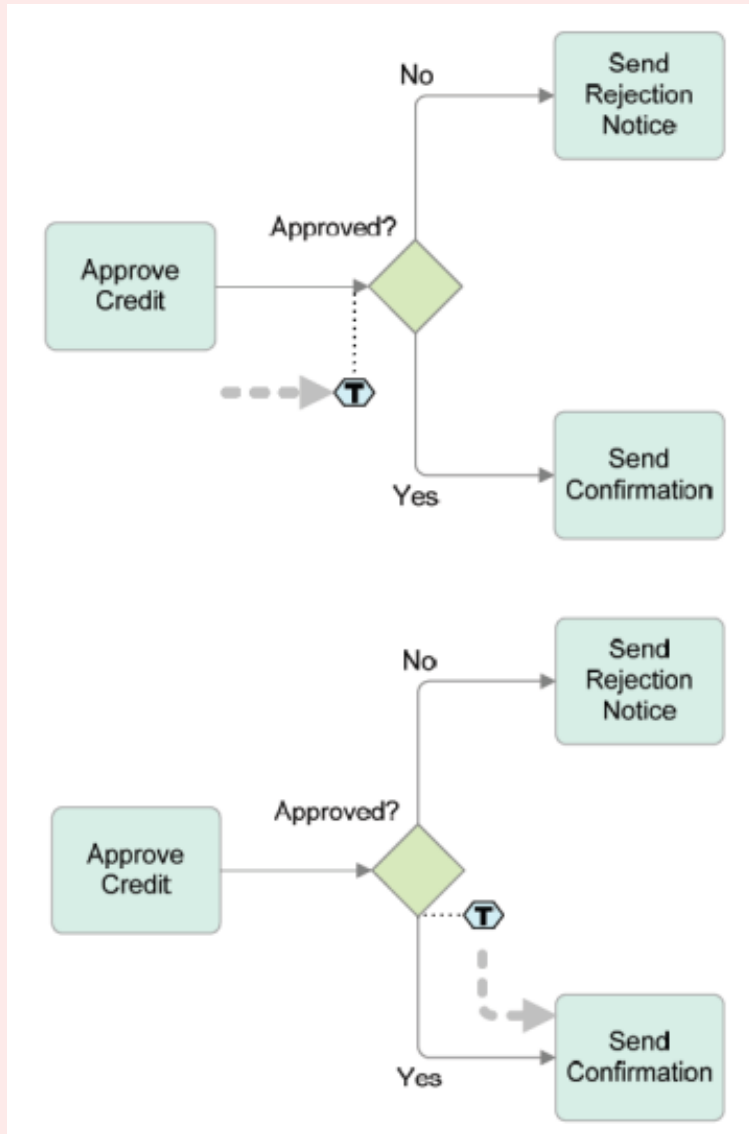
- Parallel Gateways (AND)

- Menyediakan mekanisme untuk mensinkronkan flow parallel (*AND-Join*) dan untuk membuat alur parallel (*AND-Split*), dengan aktivitas yang dapat dieksekusi secara bersamaan.
- Digambarkan oleh bentuk diamond yang harus memiliki tanda “+”



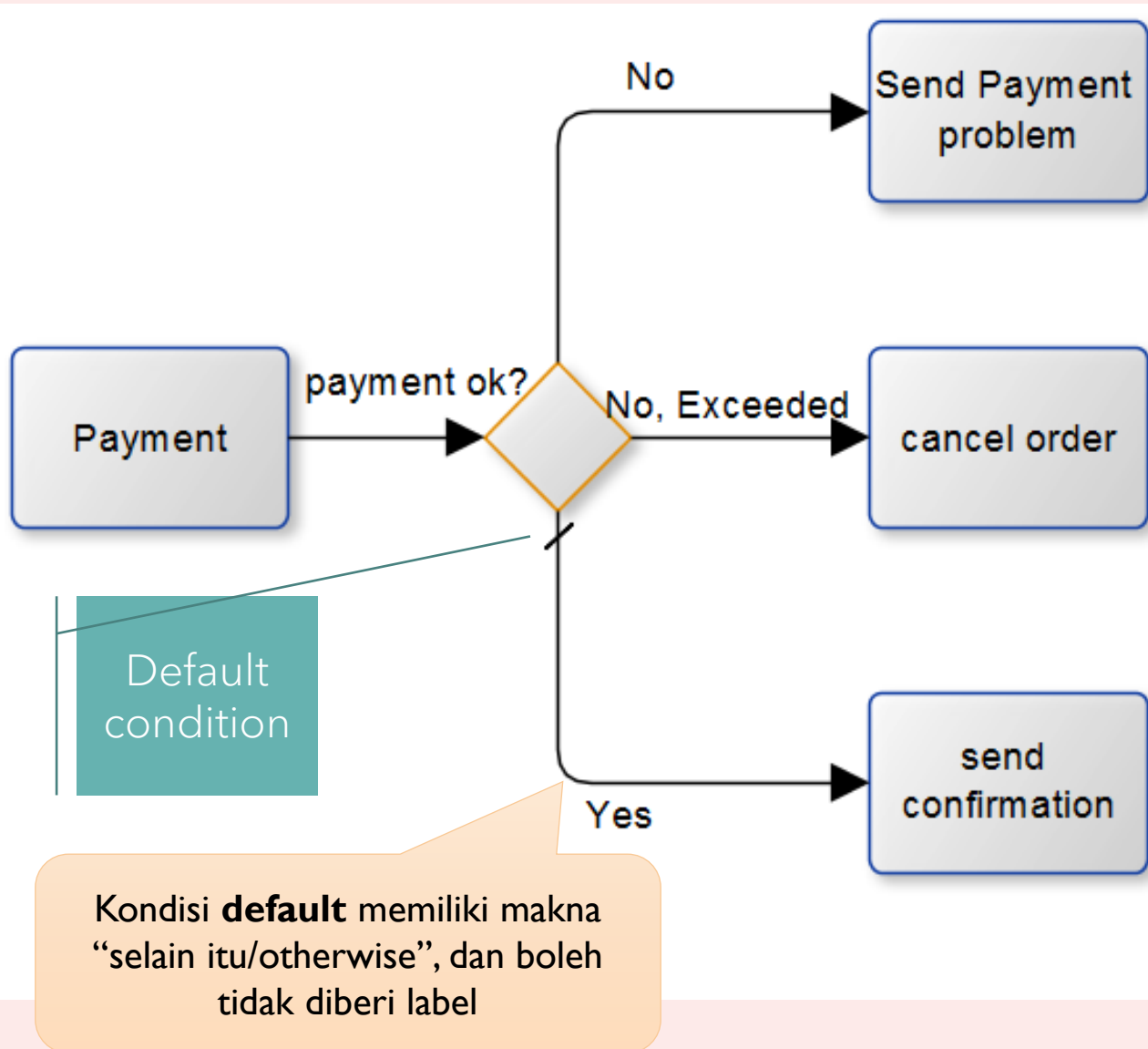


# EXCLUSIVE GATEWAY – SPLITTING BEHAVIOR



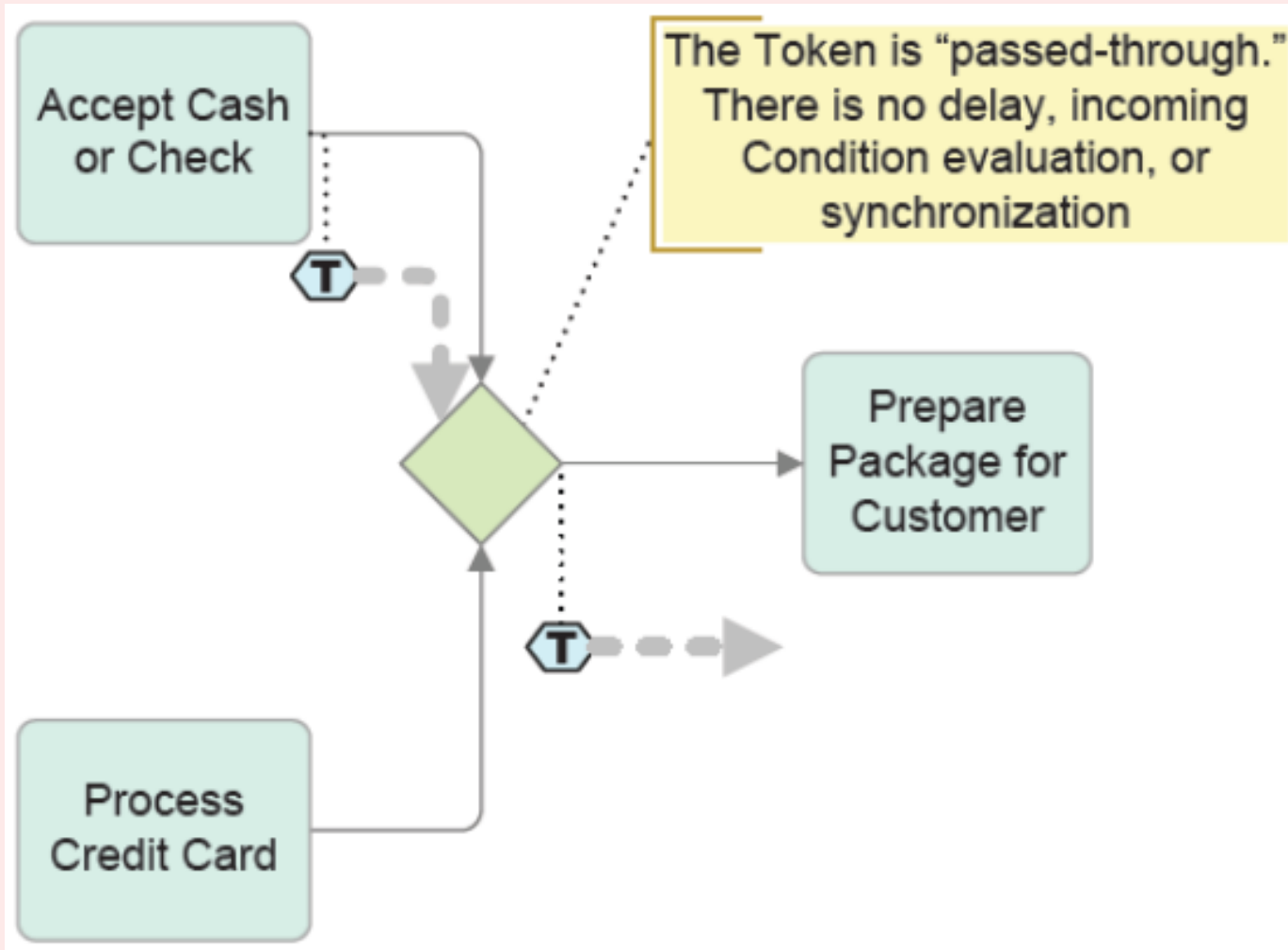
- **Exclusive gateway** adalah lokasi didalam proses dimana terdapat dua atau lebih jalur alternative
- **Kriteria** untuk keputusan yang diambil, muncul sebagai kondisi pada setiap cabang *outgoing sequence flow*
- Ketika **token** tiba pada exclusive gateway, seketika itu terjadi evaluasi dari kondisi yang akan keluar dari gateway. Salah satu dari kondisi yang keluar tersebut harus selalu kondisi jika hasil evaluasi bernilai "benar"

# DEFAULT CONDITION



- Satu cara untuk memastikan bahwa proses tidak terjebak pada exclusive gateway adalah dengan menggunakan **default condition** pada satu dari sequence flow yang keluar.
- **Default condition** dapat melengkapi sekumpulan kondisi standar untuk menyediakan **automatic escape mechanism** jika semua kondisi standar memiliki hasil evaluasi “false/salah”
- **Default** dipilih jika seluruh kondisi yang lain salah.

# EXCLUSIVE GATEWAY – MERGING BEHAVIOR

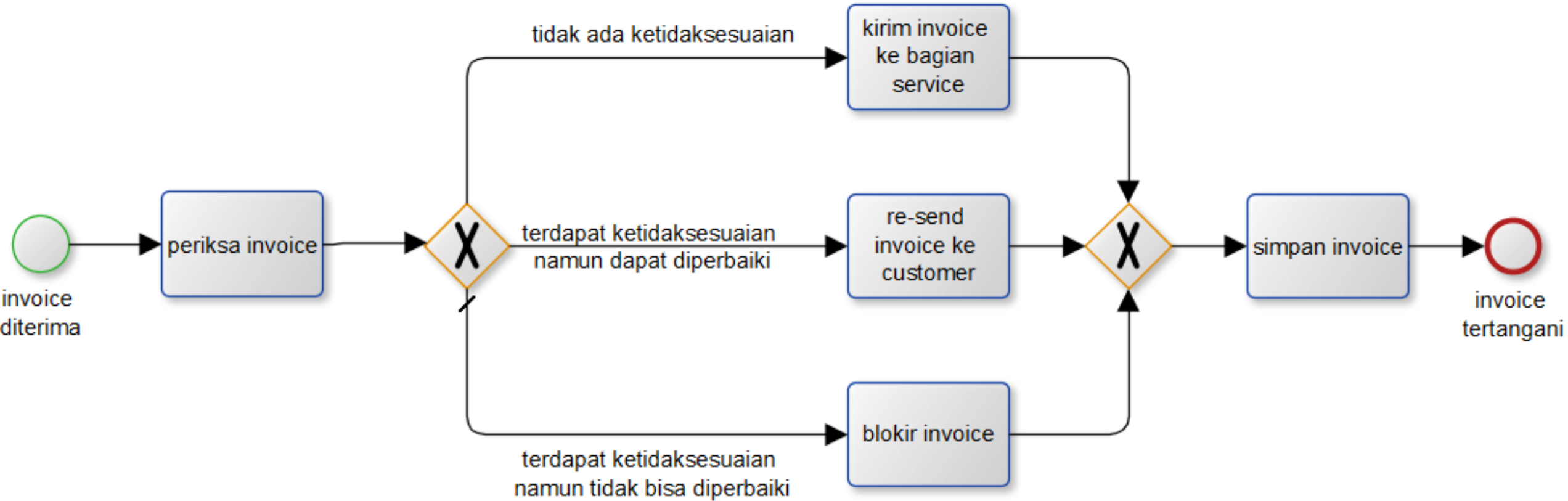


- Ketika token tiba pada exclusive gateway, tidak ada evaluasi dari kondisi (pada sequence flow yang datang/masuk), dan langsung bergerak menjadi sequence flow yang keluar/outgoing

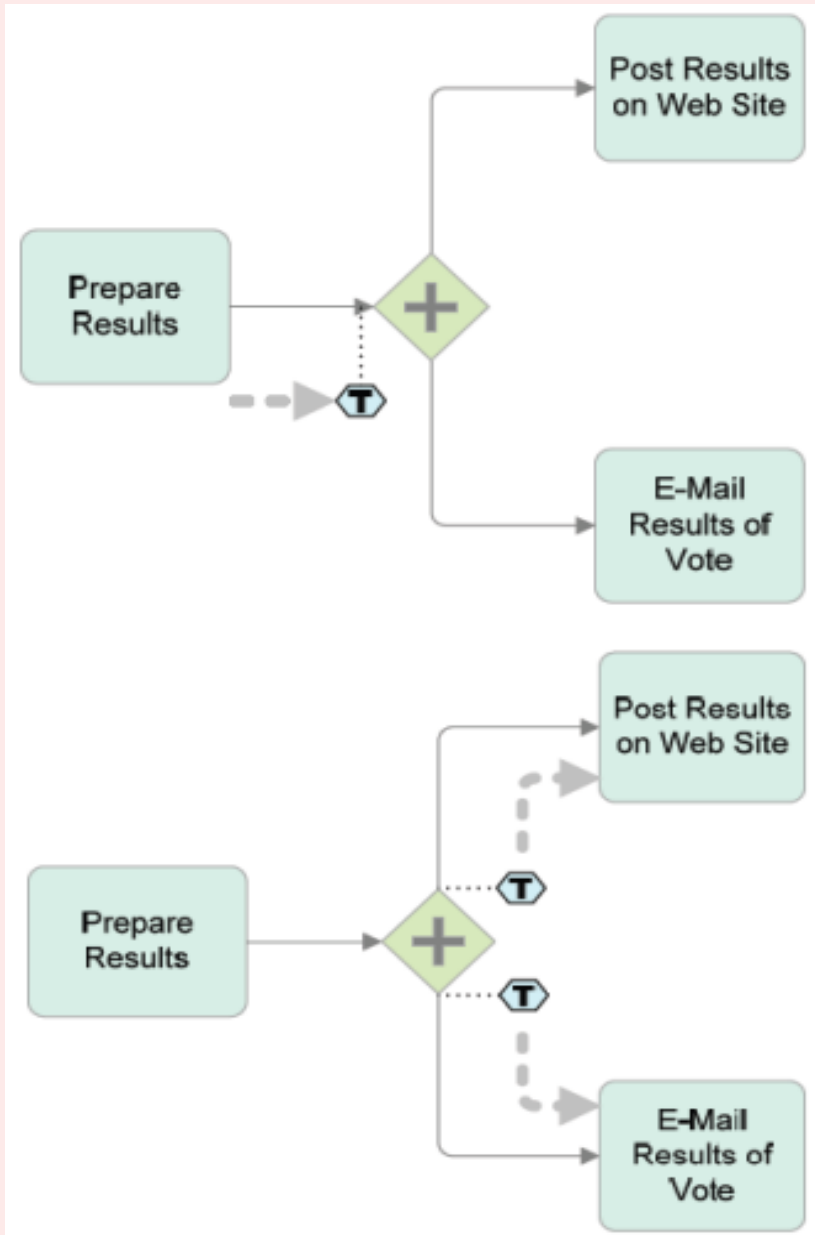
# LATIHAN 1

- Ketika invoice diterima dari customer, invoice tersebut harus diperiksa kesesuaiannya
- Hasil pemeriksaannya mungkin akan menghasilkan salah satu kondisi dari tiga kemungkinan berikut:
  1. Tidak ada ketidaksesuaian, kemudian invoice dikirim ke bagian yang menangani
  2. Ada ketidaksesuaian namun dapat diperbaiki, dalam kasus ini, invoice dikirim kembali ke customer untuk diperbaiki
  3. Ada ketidaksesuaian namun tidak dapat diperbaiki, dalam kasus ini invoice diblokir
- Ketika satu dari tiga kemungkinan aktivitas diatas dilakukan, invoice diletakkan ditempat yang seharusnya dan proses pun selesai

# SOLUSI



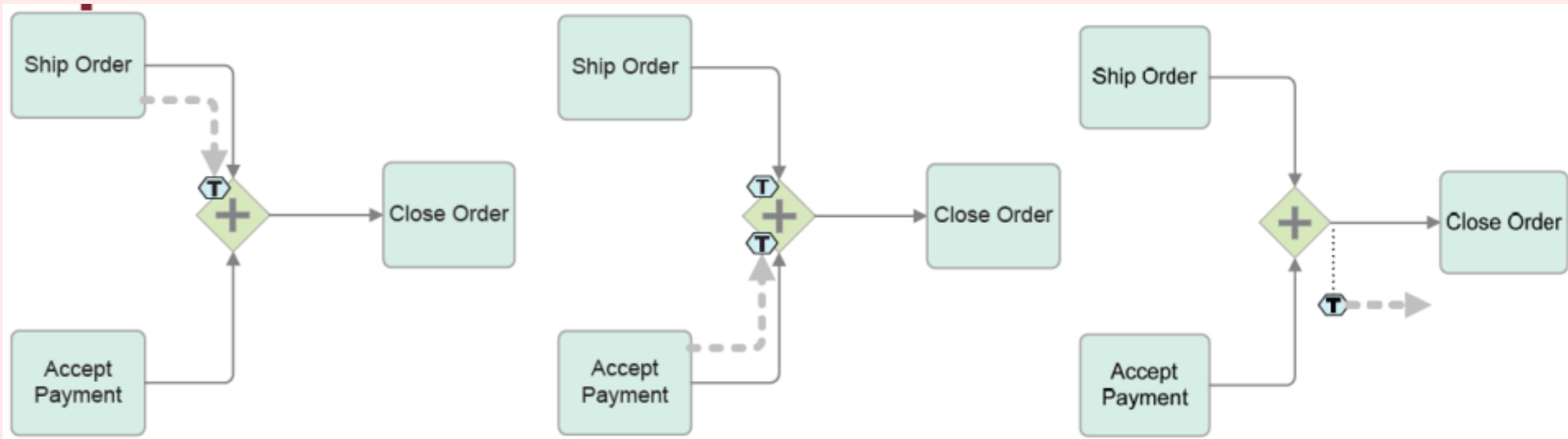
# PARALLEL GATEWAY – SPLITTING BEHAVIOR



- Ketika token tiba pada parallel gateway, tidak ada evaluasi dari semua kondisi pada outgoing sequence flow.
- Parallel gateway akan menciptakan jalur yang parallel
- Hal ini berarti bahwa gateway akan menciptakan jumlah token yang sama dengan jumlah outgoing sequence flow. Satu token akan menyusuri setiap sequence flow yang keluar (outgoing sequence flow)
- Hal ini menyebabkan aktivitas parallel dapat dilaksanakan secara bersamaan

# PARALLEL GATEWAY – MERGING BEHAVIOR

- Untuk menyinkronkan flow, parallel gateway akan menunggu setiap token tiba dari setiap *incoming sequence flow*
- Ketika token pertama tiba, tidak ada evaluasi terhadap kondisi *incoming sequence flow*, namun token tersebut "*ditahan*" di gateway dan tidak lanjut
- Ketika semua token tiba, kemudian mereka disatukan dan menjadi satu token yang mengalir menuju *outgoing sequence flow*



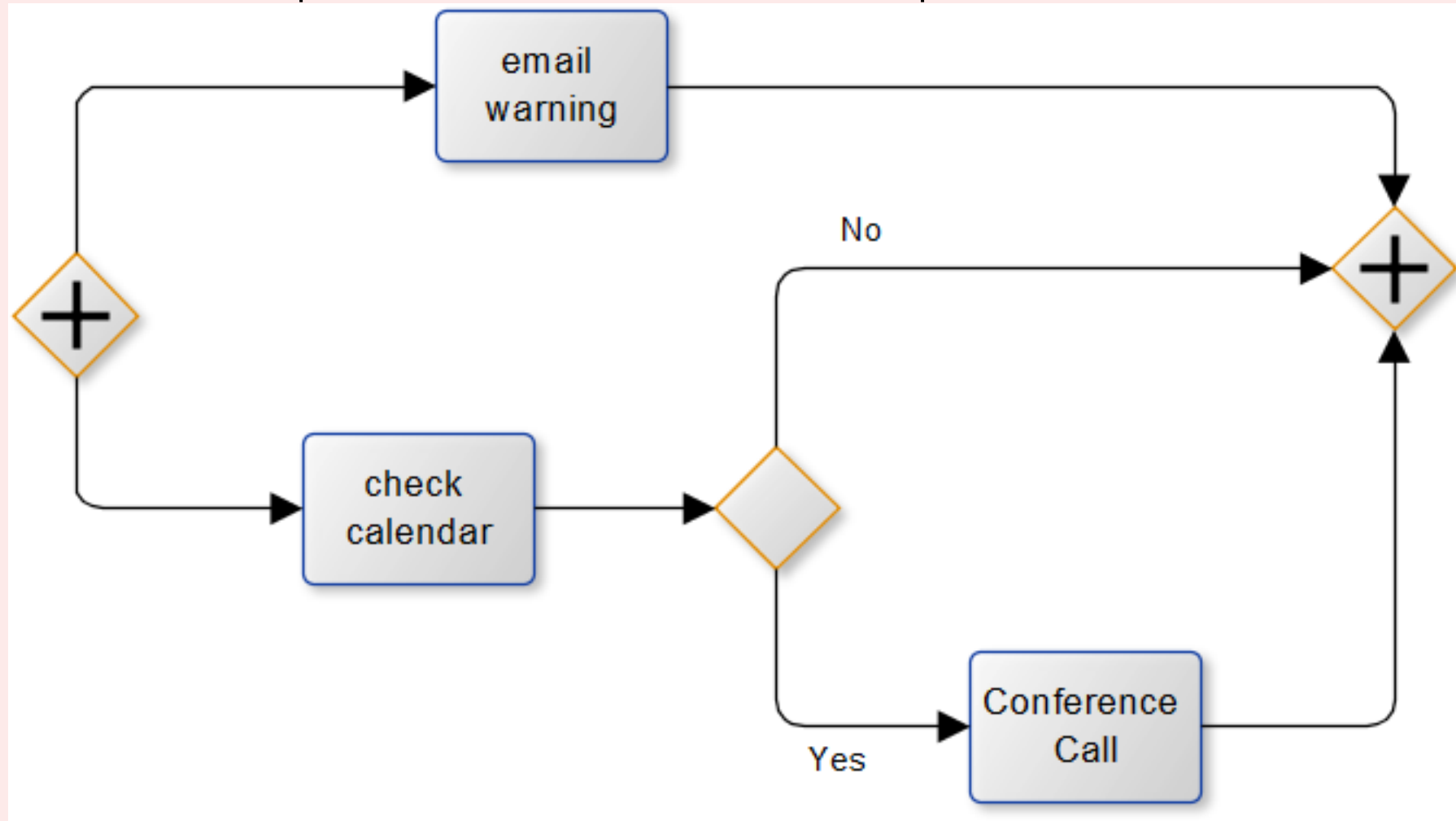
# LATIHAN 2

- Ketika penumpang telah menerima boarding pass, penumpang masuk ke ruang tunggu pesawat dengan melewati *security check point*. Disini mereka harus menjalani pemeriksaan keamanan untuk personal dan juga untuk barang bagasi yang mereka bawa. Setelah itu mereka dapat masuk ke ruang tunggu keberangkatan (*departure level*)



# LATIHAN 3

- Jelaskan perilaku/behavior dari proses berikut



Proses ini menggambarkan **pengecekan jadwal** untuk menentukan apakah perlu dilakukan **conference call**.

Terdapat dua gateway utama:

1. **Parallel Gateway (+)** – digunakan untuk menjalankan dua aktivitas secara bersamaan.
2. **Exclusive Gateway (berbentuk wajik dengan panah "Yes/No")** – digunakan untuk pengambilan keputusan berdasarkan hasil pemeriksaan kalender.

### Alur Proses

#### 1. Start → Parallel Gateway (+)

Proses dimulai dan langsung bercabang menjadi dua alur paralel:

1. Alur pertama menuju aktivitas **"email warning"**.
2. Alur kedua menuju aktivitas **"check calendar"**.

#### 2. email warning

Aktivitas ini mengirimkan peringatan melalui email – bisa jadi untuk mengingatkan tentang rapat yang akan datang atau jadwal penting lainnya.

#### 3. Check calendar → Decision Gateway (Yes/No)

Setelah memeriksa kalender, sistem atau pengguna menentukan apakah **conference call** diperlukan.

1. Jika **Yes** → lanjut ke aktivitas **"Conference Call"**.
2. Jika **No** → langsung menuju titik penyatuan (join).

#### 4. Conference Call

Jika diperlukan, dilakukan konferensi panggilan (meeting online). Setelah selesai, alurnya juga menuju titik penyatuan.

#### 5. Join Parallel Gateway (+)

Kedua jalur (hasil dari **"email warning"** dan hasil keputusan dari **"check calendar"**) bertemu di sini.

Karena ini adalah **parallel join**, proses akan **menunggu kedua aktivitas selesai** sebelum lanjut ke langkah berikutnya (kemungkinan menuju *end event*).

## Perilaku/Behavior Proses

1. Proses dimulai **secara paralel**, artinya "*email warning*" dan "*check calendar*" dilakukan **bersamaan**.
2. Setelah *check calendar* selesai, keputusan dibuat:
  1. Jika rapat diperlukan, dilakukan *conference call*.
  2. Jika tidak, langsung lanjut ke *join gateway*.
3. Proses **baru dapat selesai** setelah kedua alur paralel (*email warning* dan hasil dari *check calendar/conference call*) selesai.
4. Ini memastikan bahwa **notifikasi selalu dikirim**, dan **pengecekan kalender serta kemungkinan rapat selalu dipertimbangkan** secara bersamaan.

## Kesimpulan

Behavior proses ini menunjukkan pola kerja **paralel dengan kondisi**:

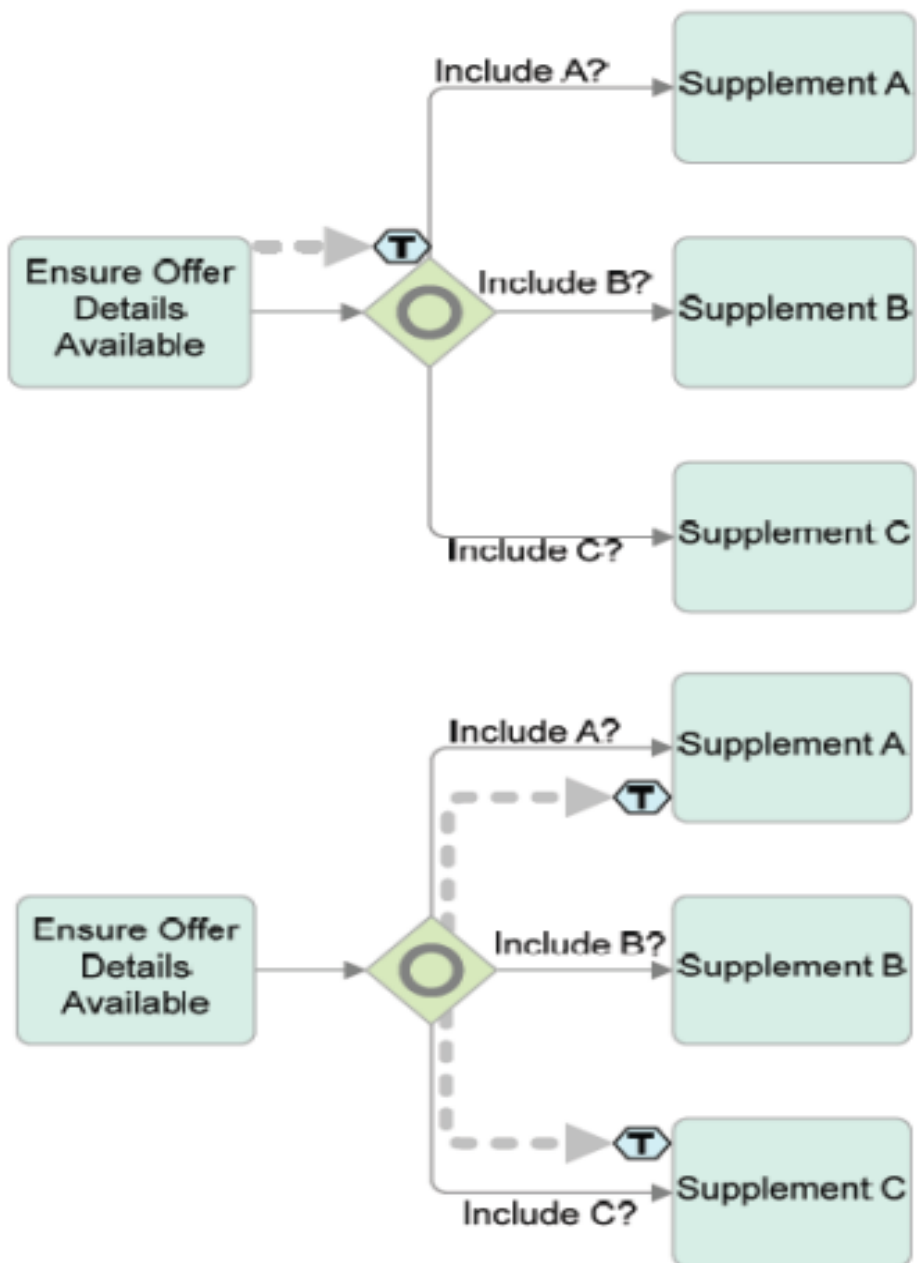
1. Dua aktivitas berjalan serentak di awal.
2. Salah satunya memiliki percabangan keputusan (Yes/No).
3. Setelah semua aktivitas selesai, proses disinkronisasi kembali.
4. Dengan kata lain:
5. "Sistem mengirimkan email peringatan sambil memeriksa kalender; jika perlu diadakan conference call, proses dilakukan, dan setelah keduanya selesai, proses berakhir."

# INCLUSIVE GATEWAY (OR)

- Terkadang mungkin dibutuhkan untuk melakukan satu atau lebih cabang setelah membuat suatu keputusan
- Untuk memodelkan situasi dimana bisa ada satu atau lebih pilihan keputusan yang harus diambil pada saat bersamaan, kita perlu menggunakan *inclusive (OR) split gateway*
- Sebuah *OR-Split Gateway* mirip dengan *XOR-Split gateway*, tetapi kondisi pada jalur yang keluar *tidak harus berlawanan*. Jadi, satu atau lebih diantara jalur yang keluar dapat saja bernilai benar pada saat yang bersamaan
- Ketika kita menggunakan *OR-Split*, kita akan melakukan satu atau lebih cabang, tergantung pada kondisi mana yang bernilai benar.

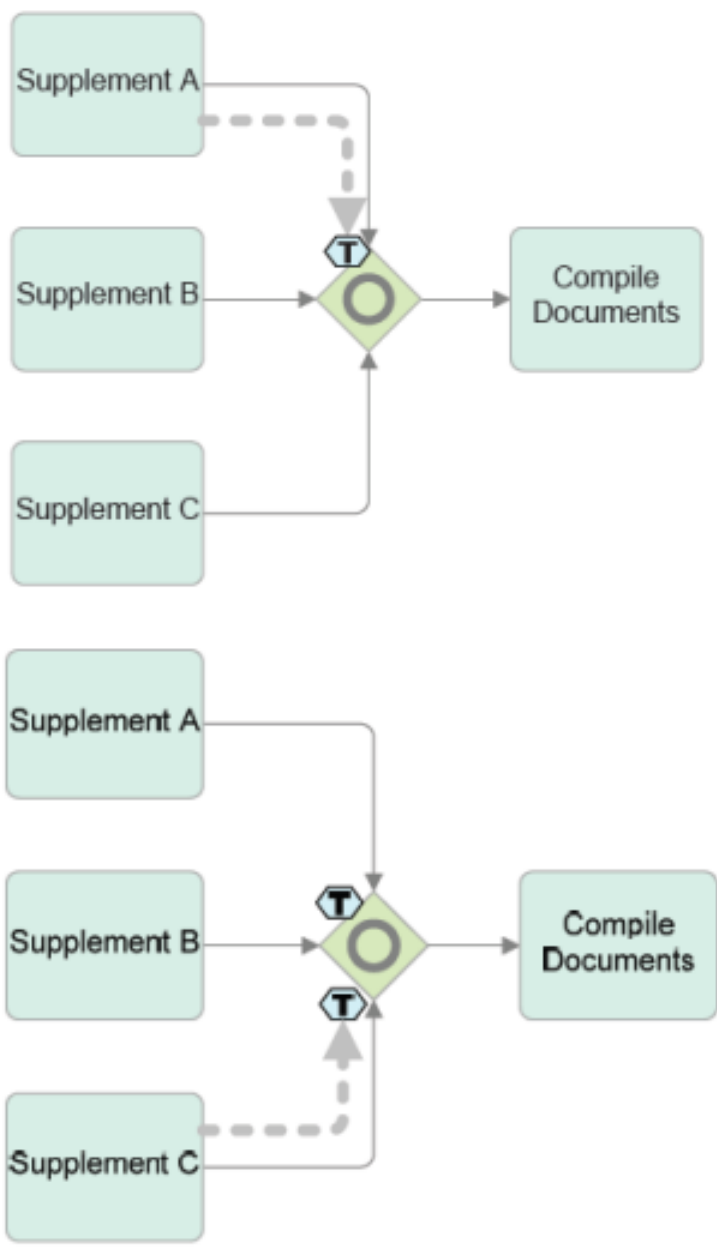


# INCLUSIVE GATEWAY (OR) – SPLITTING BEHAVIOR



- Inclusive Gateway mensupport keputusan dimana terdapat kemungkinan outcome lebih dari satu.
- Inclusive gateway yang memiliki outgoing sequence flow menghasilkan satu atau lebih jalur tergantung pada kondisi pada sequence flow yang terkait
- Paling sedikit satu dari semua kondisi bernilai benar.

## INCLUSIVE GATEWAY (OR) – MERGING BEHAVIOR

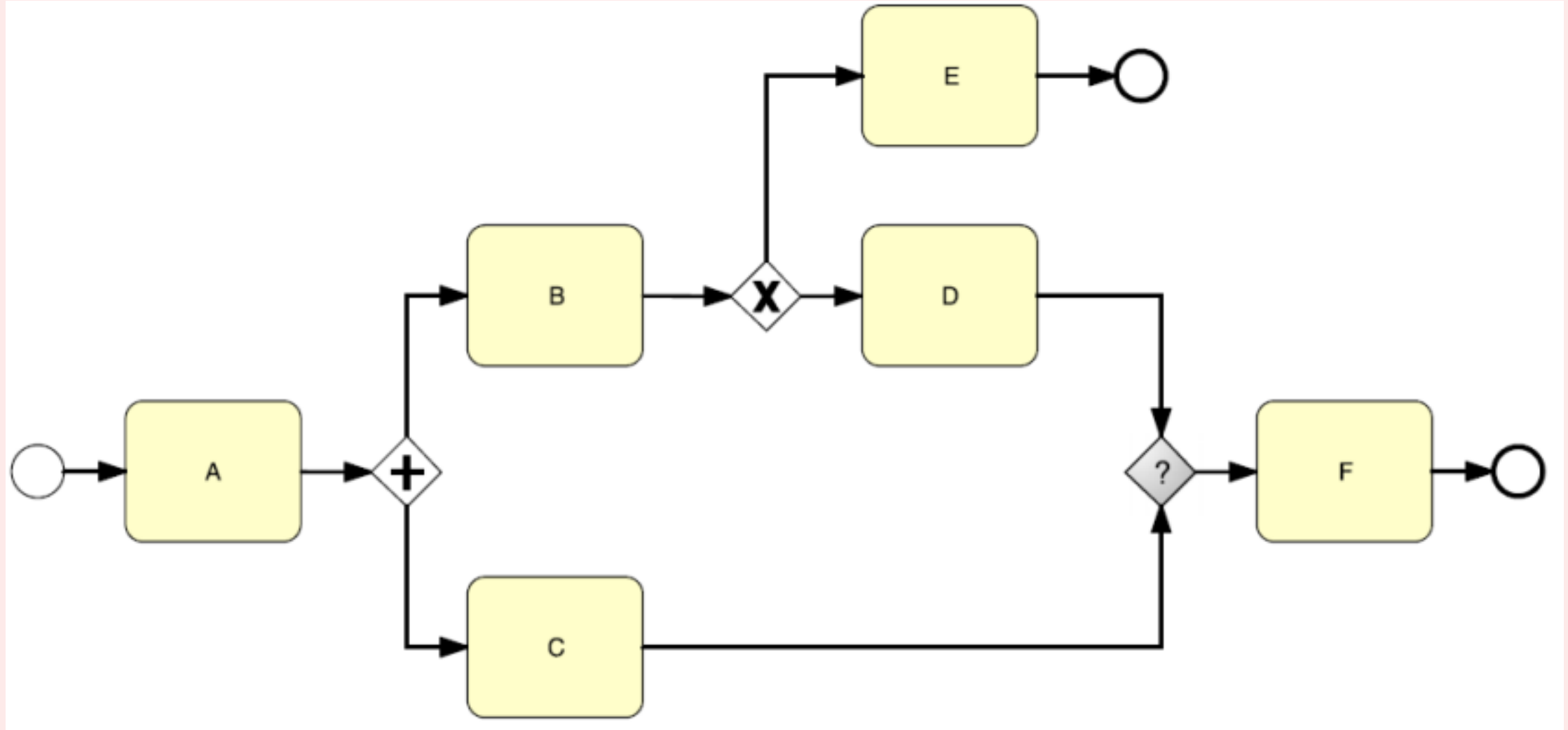


- Ketika token pertama tiba pada gateway, gateway akan mencari ke atas dari setiap incoming sequence flow untuk melihat apakah ada token yang akan tiba selanjutnya
- Sehingga, gateway akan menahan token pertama dari path yang atas hingga token dari path dibawahnya tiba
- Ketika semua token tiba di gateway, alur proses akan disinkronisasi (incoming token digabungkan) dan kemudian token mengalir pada outgoing sequence flow

# LATIHAN 4

- Sebuah perusahaan memiliki dua buah gudang yang menyimpan produk yang berbeda. Lokasi gudang di: Amsterdam dan Hamburg
- Ketika order diterima, order tersebut di distribusikan ke dua gudang tersebut dengan kondisi: jika produk yang terkait disimpan di Amsterdam, maka sub-order dikirim kesana, dan begitu juga jika produk yang terkait berada di Hamburg, sub-order pun dikirim kesana
- Setelah itu, order telah terdaftar dan proses selesai

## TIPE GATEWAY APA YANG SEHARUSNYA KITA GUNAKAN PADA JOIN BERIKUT?





## Penjelasan:

Mari kita analisis alurnya satu per satu:

### 1. Dari awal:

1. Aktivitas **A** dijalankan terlebih dahulu.
2. Setelah itu, proses bercabang melalui **parallel gateway (+)** ke dua jalur:
  1. Jalur 1 → **B**
  2. Jalur 2 → **C**
3. Artinya **B** dan **C** dilakukan secara paralel.

### 2. Setelah B:

1. Dari **B**, alur menuju ke gateway **X (exclusive)** yang akan menentukan apakah menuju ke **D** atau **E**.
2. Jika menuju **E**, proses berakhir (karena ada end event setelah **E**).
3. Jika menuju **D**, proses berlanjut ke bawah.

### 3. Setelah D:

1. Aktivitas **D** dilanjutkan ke gateway **?**, yang menggabungkan dua aliran: dari **D** dan dari **C**.
2. Setelah bergabung, proses masuk ke **F**, lalu berakhir (end event).

### Mengapa harus *exclusive gateway (XOR)*?

- Tujuan gateway ini adalah **menggabungkan dua aliran proses (merge)** dari **C** dan **D**.
- Namun, dalam model ini **tidak ada indikasi bahwa kedua jalur (C dan D) akan selesai secara bersamaan**, karena:
  - Jalur **D** hanya terjadi jika kondisi tertentu di gateway **X** terpenuhi, sementara jalur **C** selalu berjalan paralel dari awal.
  - Maka, pada titik gabung, **salah satu atau keduanya bisa mencapai titik tersebut pada waktu yang berbeda**.
- Jika kita menggunakan **parallel gateway (AND)** di sini, proses akan menunggu kedua alur selesai secara bersamaan — yang bisa menyebabkan *deadlock* jika salah satu alur (misalnya **E**) sudah selesai terlebih dahulu dan tidak pernah sampai ke gateway tersebut.
- Dengan **exclusive gateway (XOR)**, proses akan **melanjutkan ke F segera setelah salah satu alur (C atau D) mencapai gateway tersebut tanpa menunggu yang lain**.

# THANK YOU

Fahmi Yusuf, S.Kom, MMSI, Ph.D

fahmionline@uniku.ac.id

