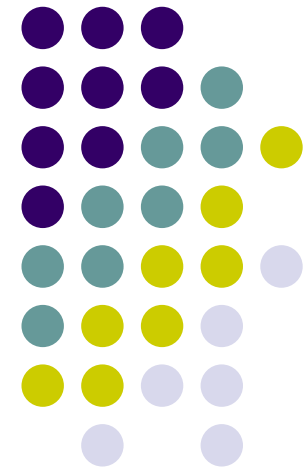


# Introduction to B2B

---

Simon Steinacker  
Patrick Mangesius

SS 2005





# Some Definitions

- Endpoint
  - Back-end application
  - Trading partners
  - Divisions within an enterprise
    - User interaction
- Close relationship between each other



# Some Definitions

- Hosted Trading Partner
  - Company that hosts server
    - hosted Trading partner
  - Manage communication with *other businesses*
    - Remote / non hosted trading partners
- Variants:
  - One Server
  - Multiple Server
  - Multiple hosted trading partners a server



# Some Definitions

- Data format and events
  - Enable communication of applications with different formats
  - B2B integration server acts as translator
  - Multiple data formats => server must be able to understand them all
- Received and sent messages are stored inside the server
  - Representation is called *events*



# Some Definitions

- Transformation
  - Applications that communicate have different data formats
  - Therefore data must be *transformed*
    - Transformation must not change semantics of the business data content
    - Semantic preserving transformation



# Some Definitions

- Domain value map
  - Often values have to be exchanged
  - E.g. shortcuts for expressions (Austria – AUT)
  - => Domain value mapping
- Huge datamaps
- Data may change frequently
- Often realized through a database look up

# Some Definitions

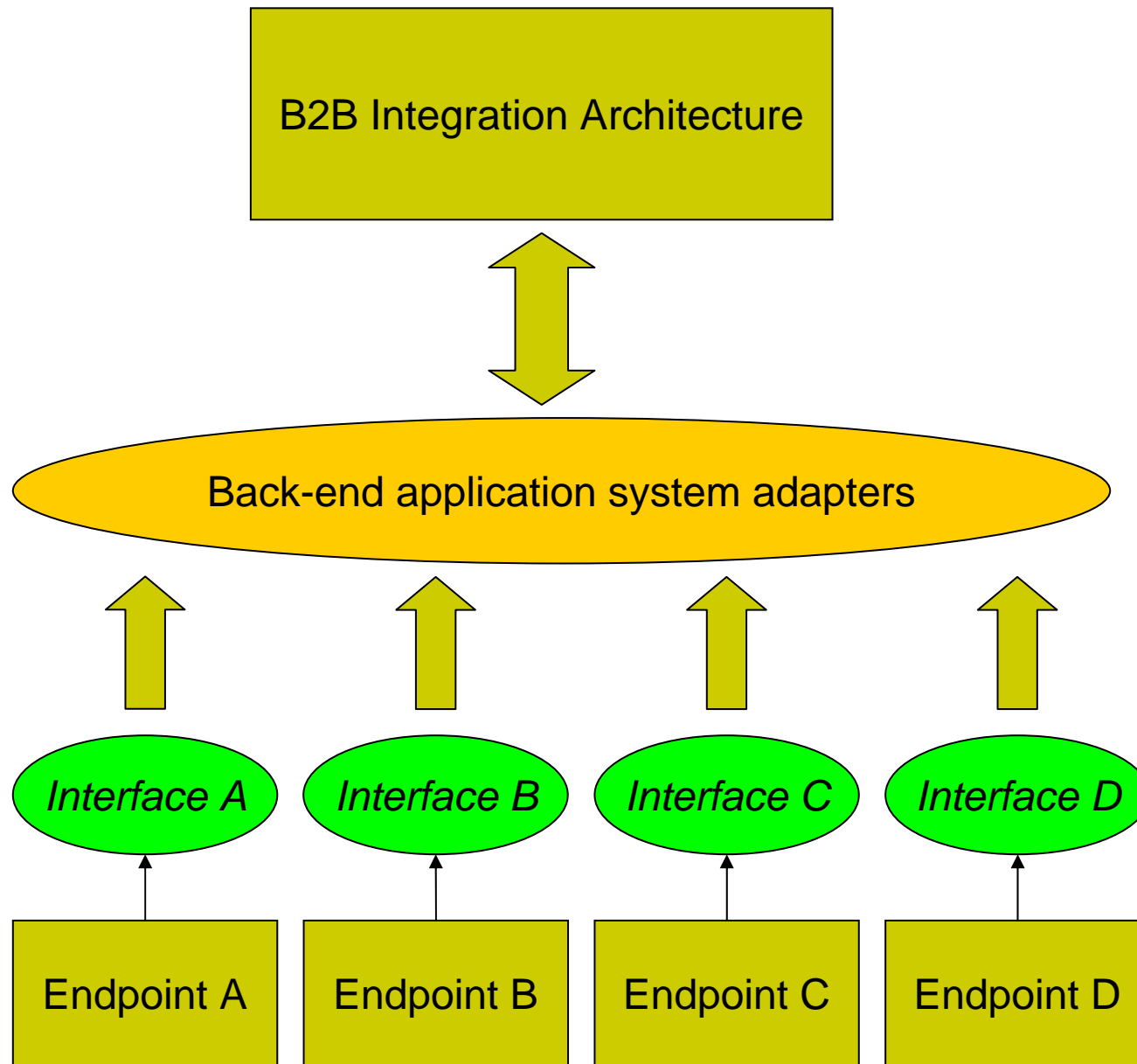


- B2B protocol
  - 2 approaches
    - Server implements one
    - Application implements the protocol as integration interface
  - There exist many protocols
  - Server must implement all needed for the conversation
  - Possibility to find out which protocol is supported by the endpoint



# Some Definitions

- Back-end application system adapters (1/3)
  - There must be some kind of gateway which enables communication
    - APIs, persistent queues, database tables, file system directories,...
  - Problems with the different functionalities of the different approaches
    - Heterogeneity
- Abstract interface (figure next slide)





# Some Definitions

- Back-end application system adapters (3/3)
  - Do NOT define behavior
  - 2 Types
    - Transactional
    - Nontransactional
  - Transactional: messages are sent once
  - Nontransactional: implement semantics like Protocols with retry limit, duplicate checks,...



# Some Definitions

- Business Process
  - Integration is not only transforming/forwarding data
  - Often complex rules have to be designed
    - E.g. Shipping example
  - Communication is not always one way
    - E.g. acknowledgements when purchasing



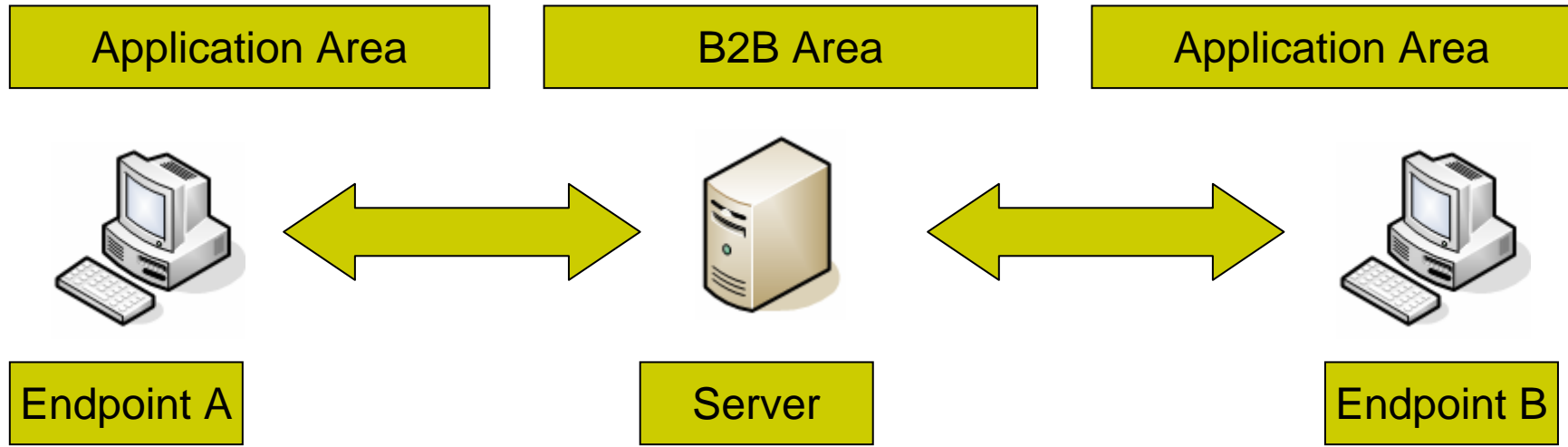
# Some Definitions

- History / Business Intelligence
  - Examples:
    - Logfiles
    - Which data is sent to which endpoint
  - Error detection/recovery
  - Auditing
    - E.g. Access logs for certain data (Healthcare)
  - Analyzing of future trends



# Integration concepts

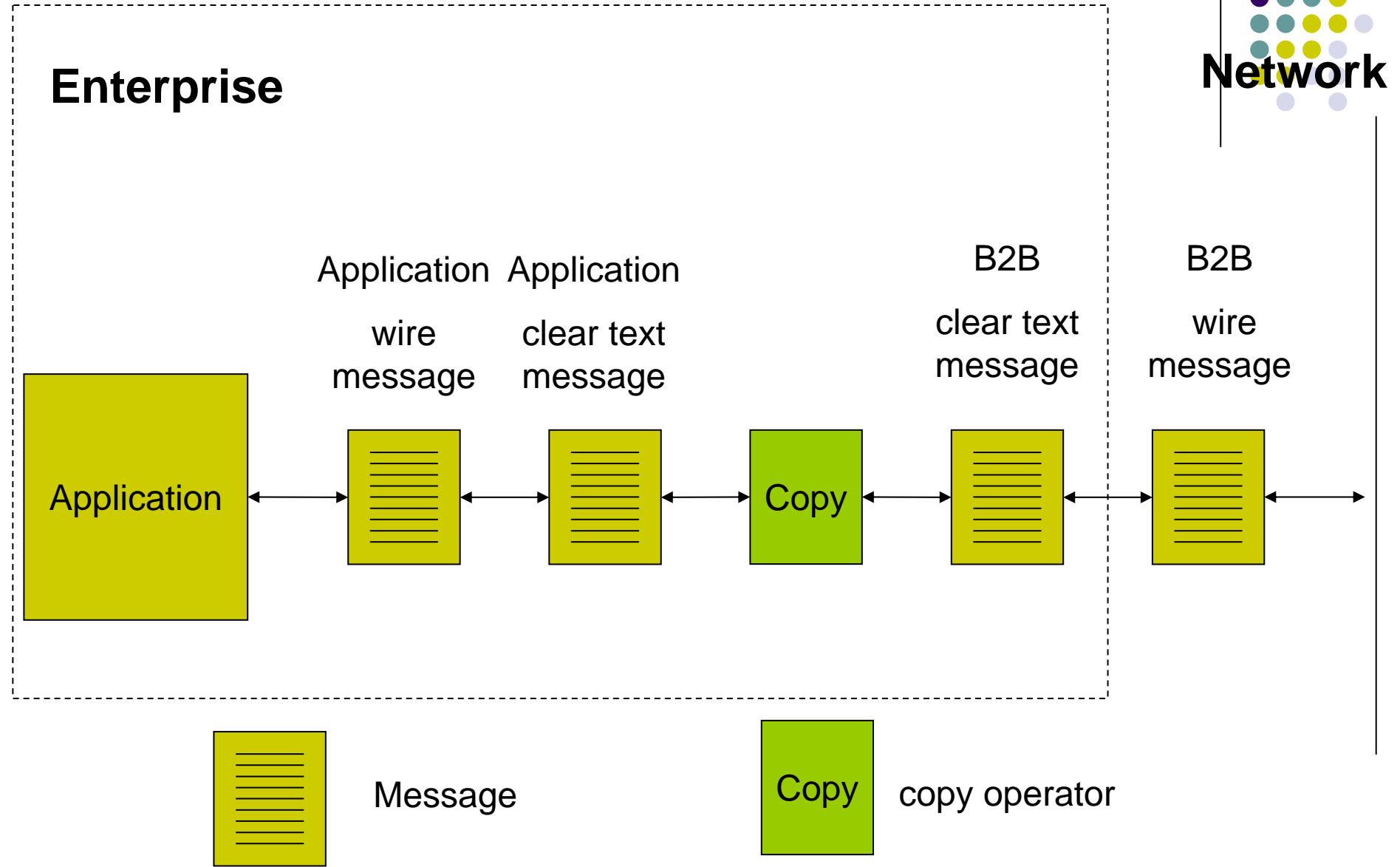
- Message
  - Stages similar to network layers
  - B2B wire message
  - B2B clear text message
  - Application clear text message
  - Application wire message





# Integration concepts

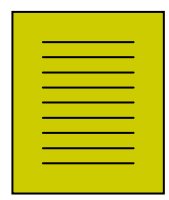
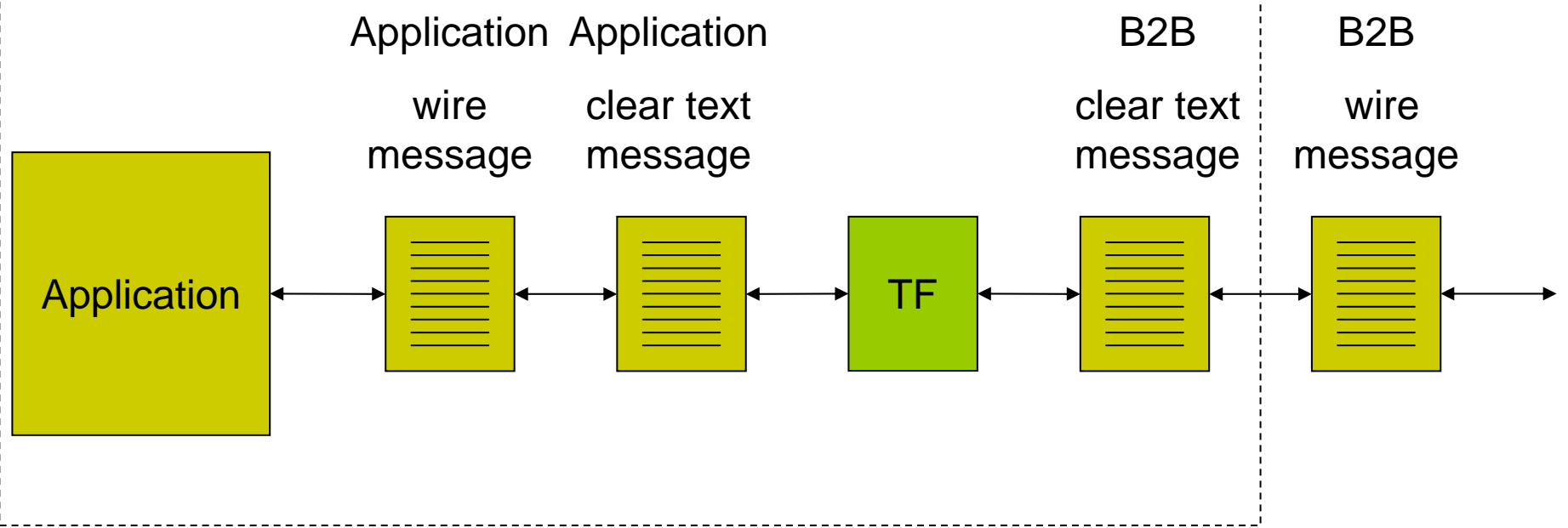
- Transformation between the messages
  - Two main differences
    - Application supports B2B protocol
    - Application uses a different protocol



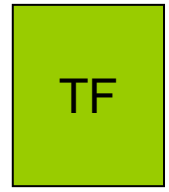


**Network**

# Enterprise



Message



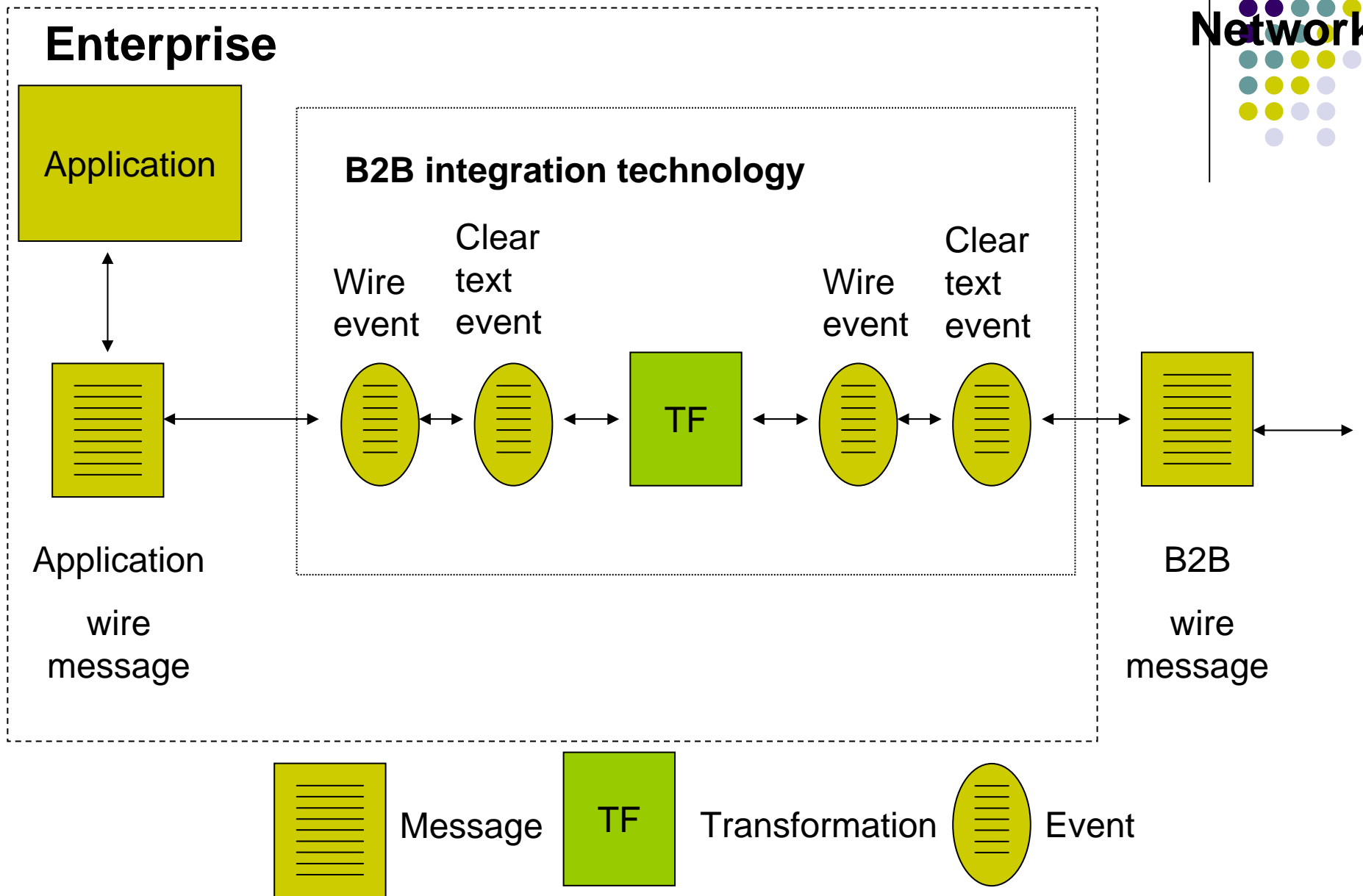
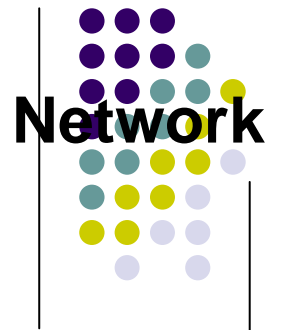
TF Transformation



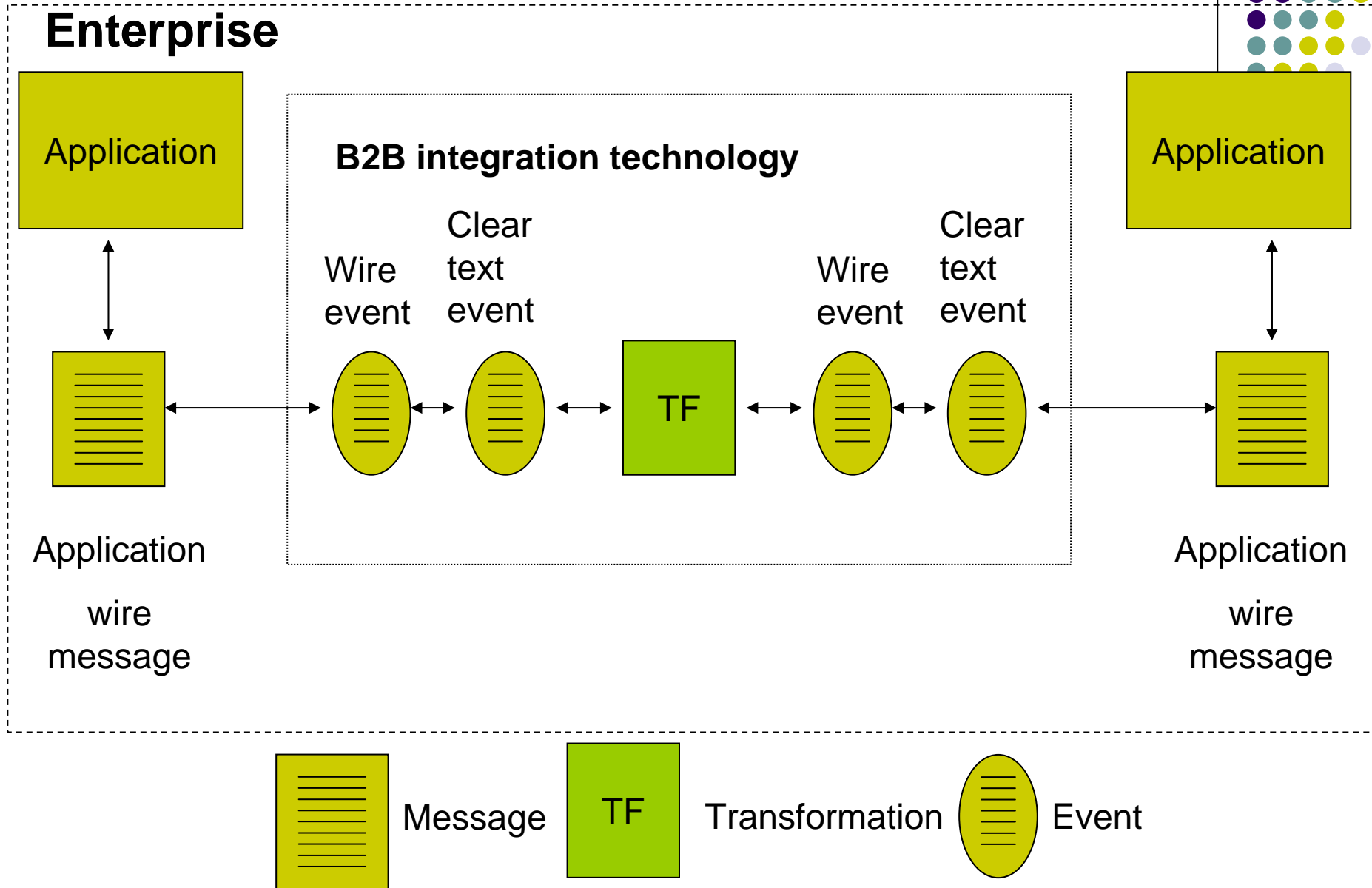
# Integration concepts

- Event
  - Include information
    - Data
    - Sender
    - Receiver, ...

# Applications talking over the network



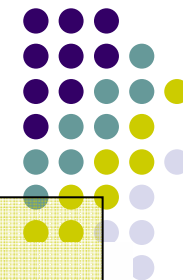
# Applications talking direct





# Integration concepts

- Transformation / Translation
  - Transformation
    - As mentioned transformation is often not only copying
    - E.g. Dataformat YYYY-MM-DD or 2005, March, 23
    - Transformation scripts
  - Translation
    - Eliminating different representations between incoming message and event syntax

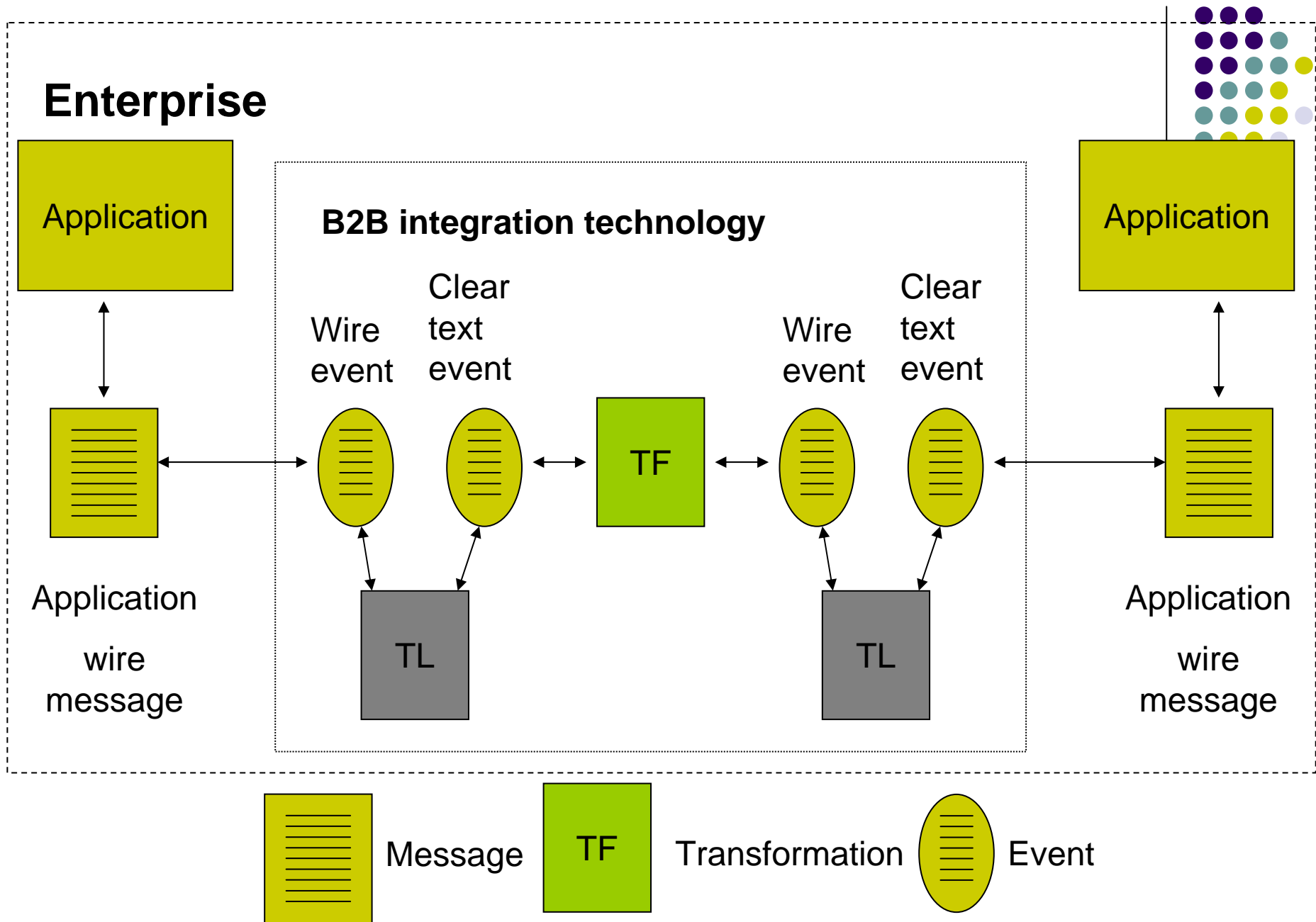


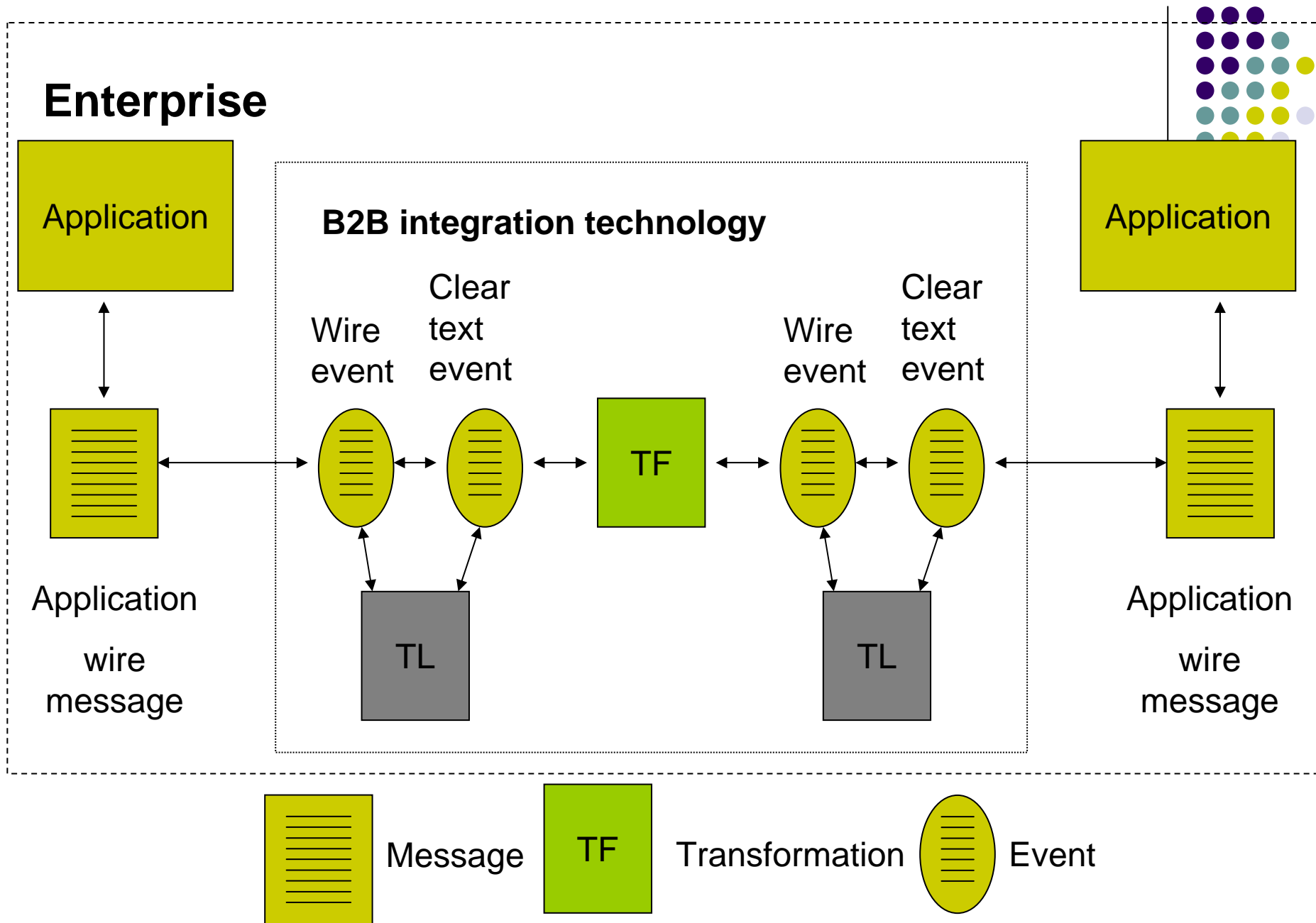
```
*Patrick*Mangesius*  
*Term*6*  
*0215407*  
*csae1493@uibk.ac.at*
```

## Message Syntax

```
<name>  
  <firstname>Patrick</firstname>  
  <lastname>Mangesius</lastname>  
</name>  
<student_information>  
  <term number="6"/>  
  <matrn value="0215407"/>  
  <email value="csae1493@uibk.ac.at"/>  
</student_information>
```

## Event Syntax





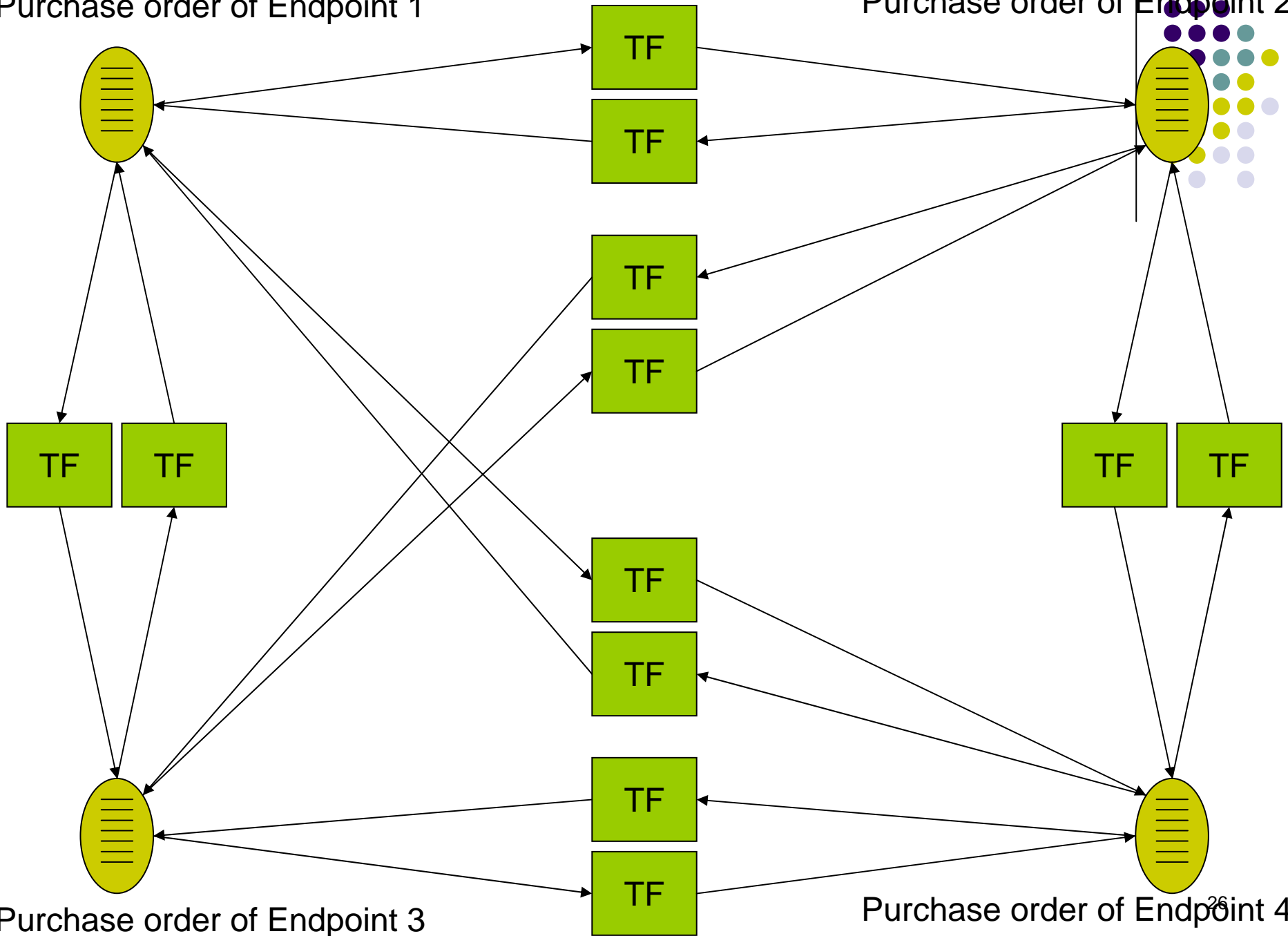


# Integration concepts

- Business event
  - Until now point to point integration
  - Problems:
    - Transformations increase rapidly with number of participants
  - Example
    - 4 companies purchasing
    - Each different protocol
      - This means: 12 transformations

Purchase order of Endpoint 1

Purchase order of Endpoint 2

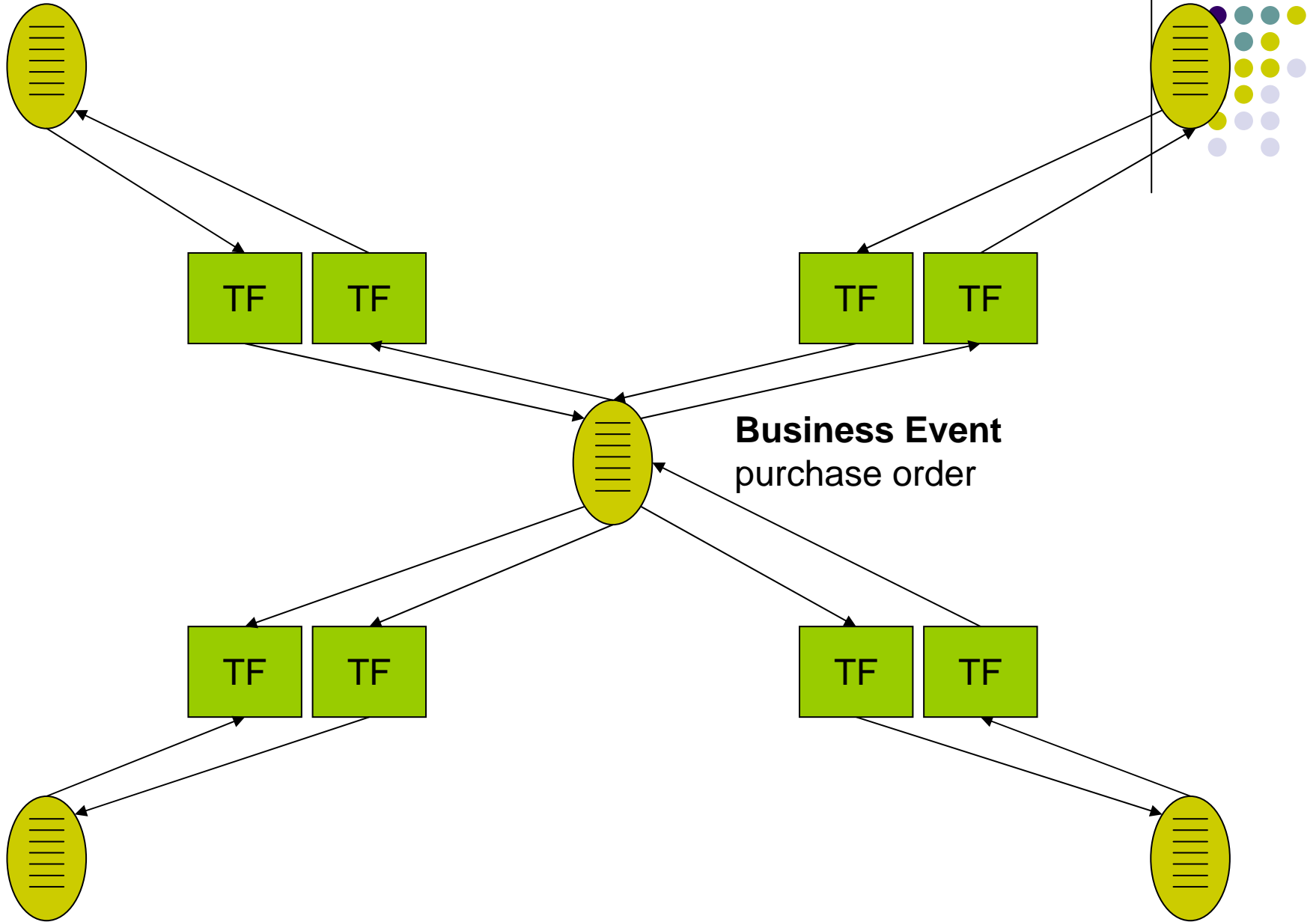


Purchase order of Endpoint 3

Purchase order of Endpoint 4

Purchase order of Endpoint 1

Purchase order of Endpoint 2



Purchase order of Endpoint 3

Purchase order of Endpoint 4



# Integration concepts

- Business event
  - Business event can be used to avoid costly transformations
    - Without we had  $n \cdot (n-1) = 4 \cdot 3 = 12$  Transformations
    - With there are only  $2 \cdot n = 2 \cdot 4 = 8$  Transformations

# Business Process & Endpoint Behavior



- Usually messages exchange between back-end app systems and trading partners are logically related.
- This relation can be expressed by defining behavior.

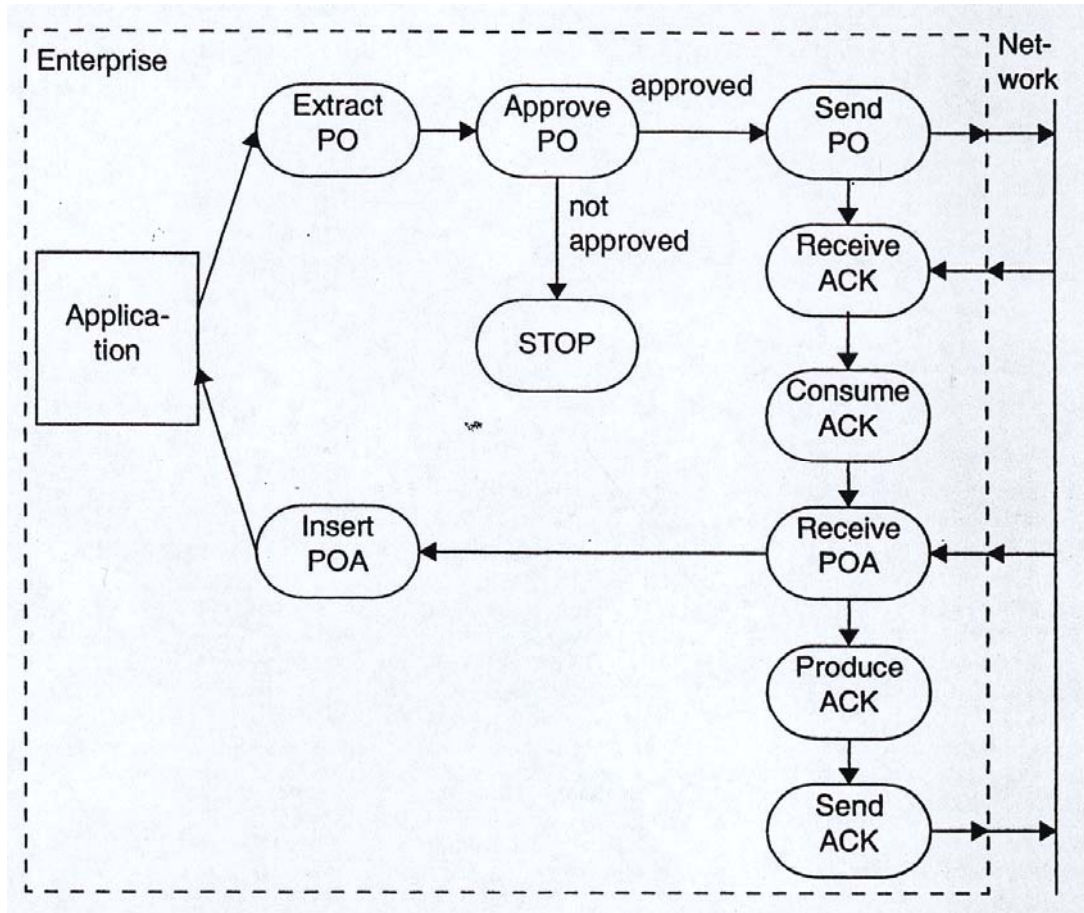


# Defining behavior

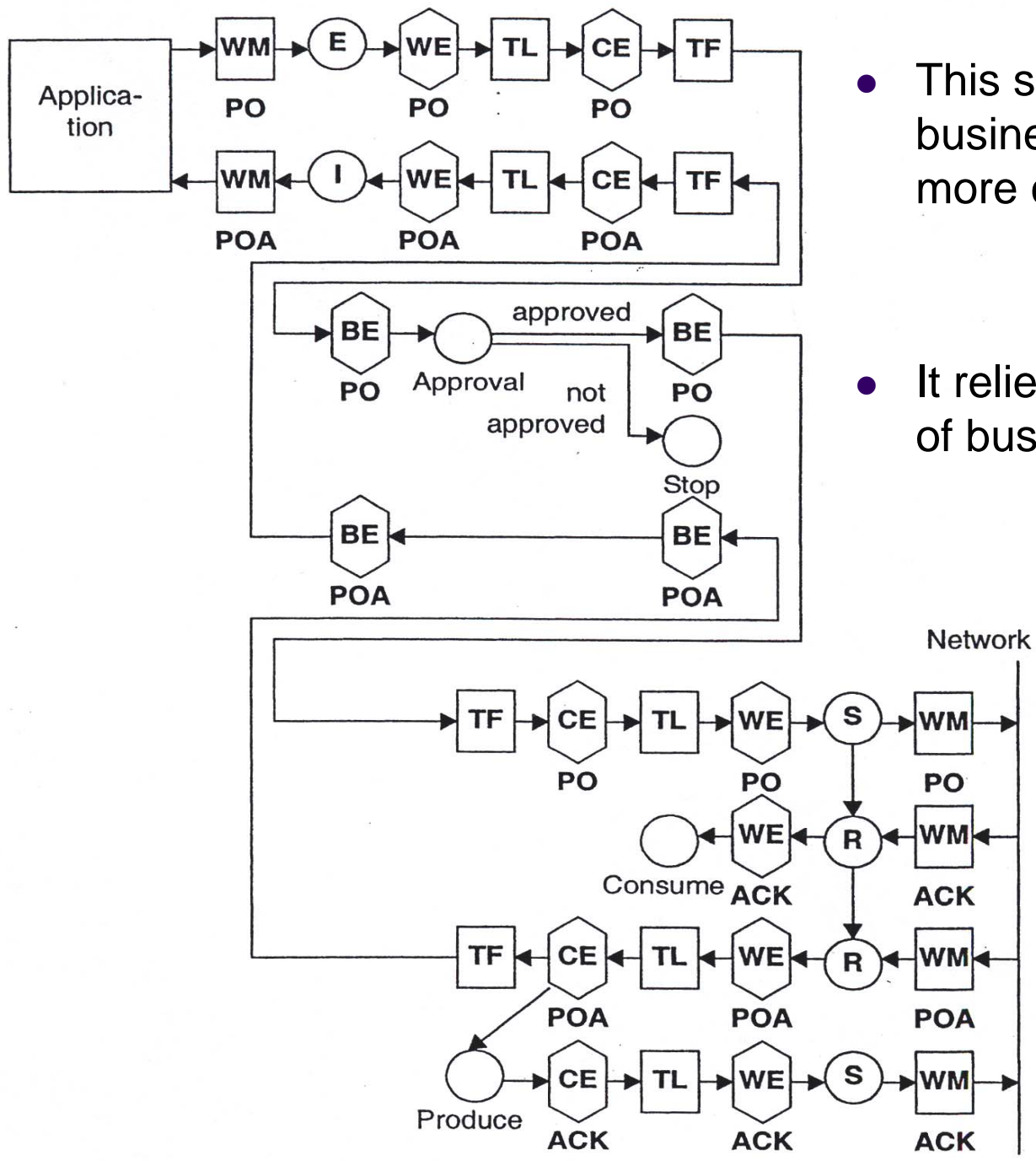
- One way to define behavior is to define a business process.
- And through that the complete message and event behavior is defined.
- The following slides will deal with that...



# Extended example of 2 endpoints



- 2 endpoints exchanging purchase order messages,
- Purchase order messages have to ack'ed
- Each message received by the first endpoint (back-end) has to be ack'ed.
- See figure on the left



- This slide shows the business process in a more concrete way.

- It relies on the concept of business events.



## Three Subdivisions ...

- The representation on the former slide is informally divided in three subdivisions:
  - The upper third concerns the first endpoint (back-end application system)
  - The middle third is about the business logic
  - And the lower third deals with messages and events concerning the second endpoint (trading partner)



# Deficiencies

- While it is certainly possible to define the whole process in one outlet, there arise several disadvantages of this approach:
  - Different layers of functionality
  - Cannot be reused



# A better approach...

- These deficiencies justify the search for a more flexible approach.

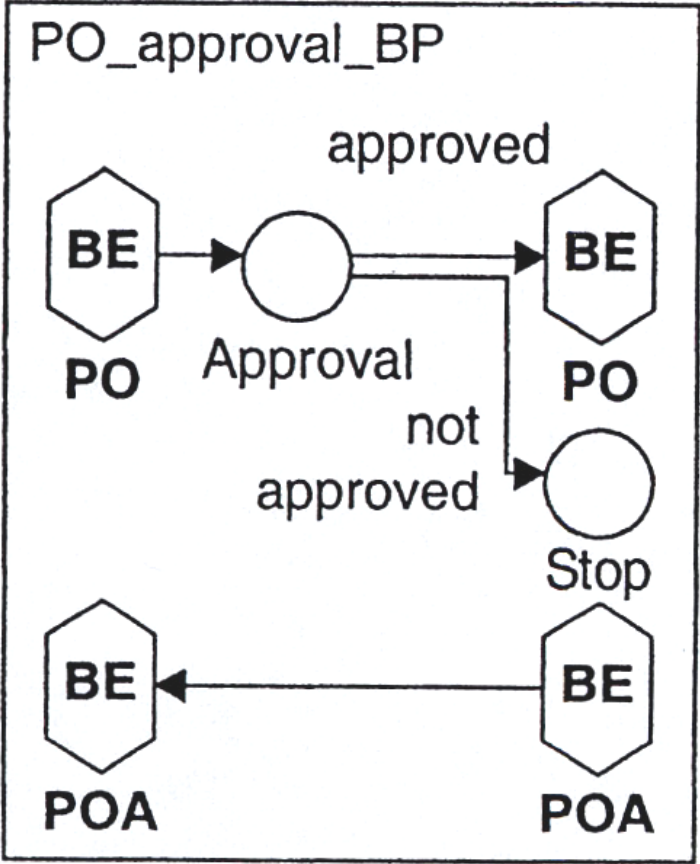
→ **Divide and Conquer**

# Divide



- First the original all-in-one process is divided into 3 sub-processes:
  - The business logic process (from now on called ‘business process’)
  - One processes for each endpoint (called ‘interface process’)

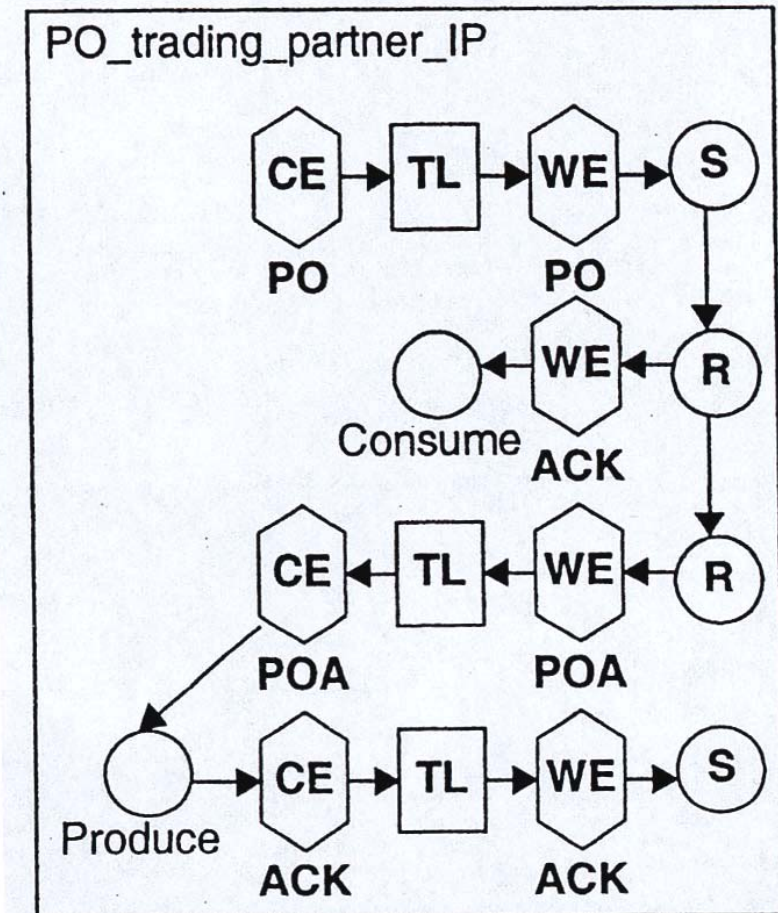
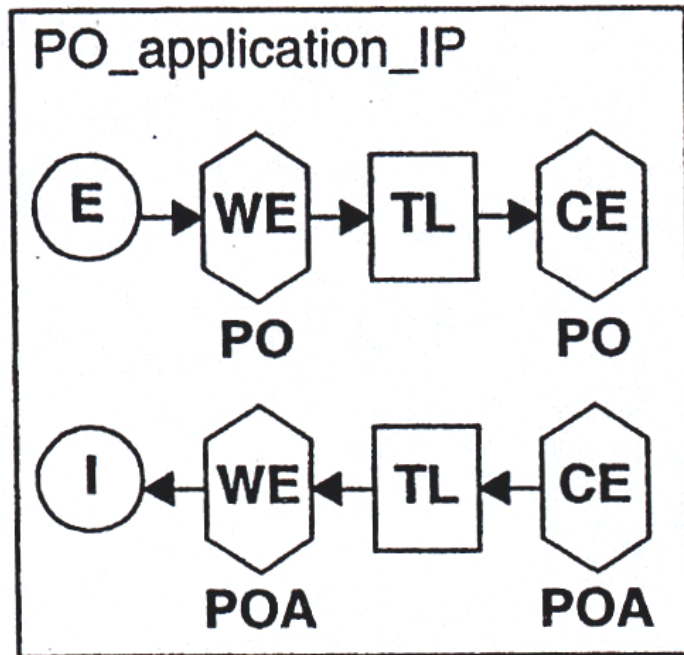
# Business Process



- This process describes the business logic and represents the 'middle part' of the previous example



# Interface Processes



These two interface processes implement the same behavior as the 'upper part' and the 'lower part' in the previous example.

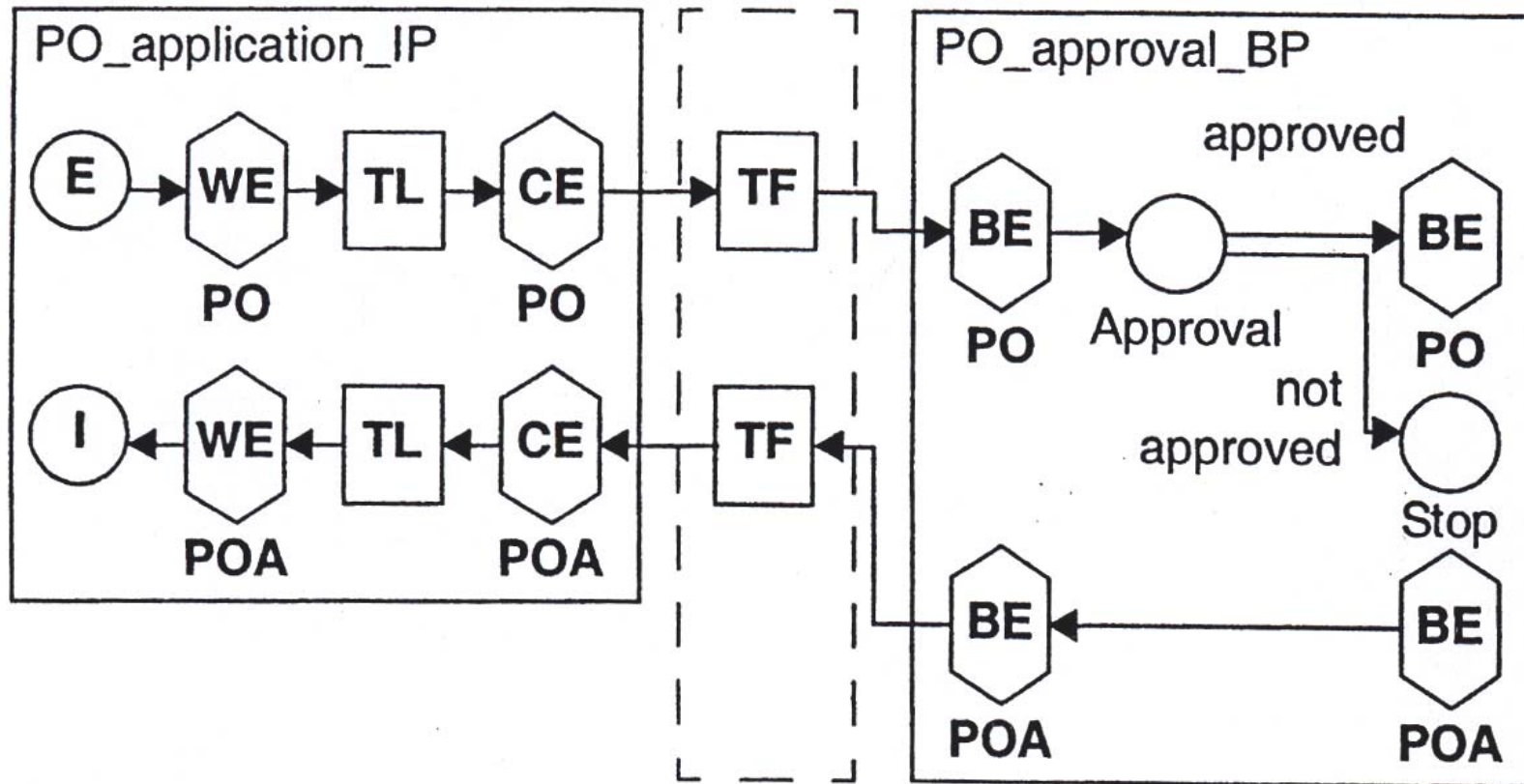


# Conquer

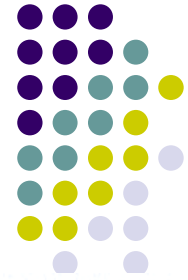
- Now that we have ‘divided’ we need to ‘conquer’.
- Conquer is meant in the sense of tying the divided components together.
  - Since none of the processes from above contained a transformation, this is done in the Conquer part.



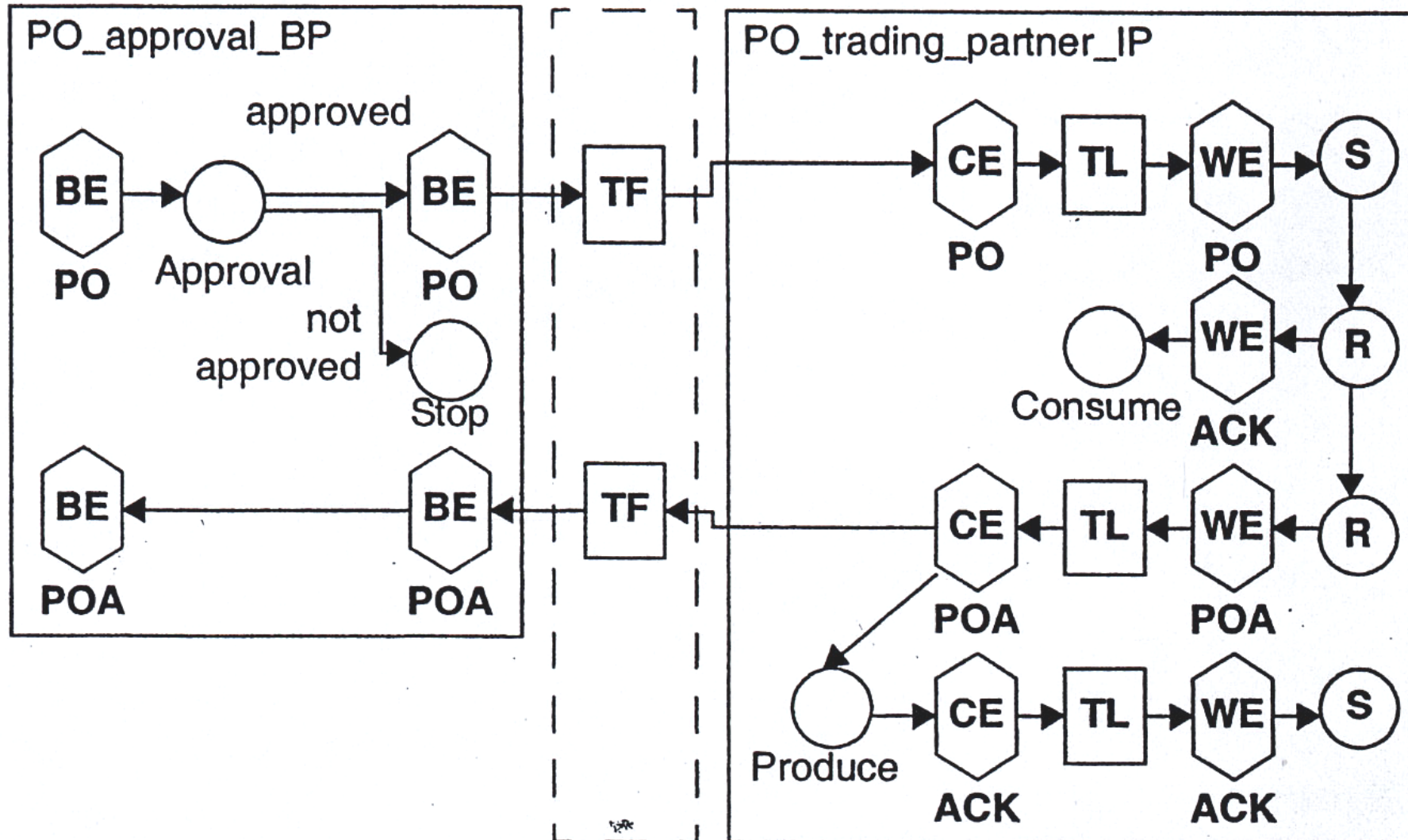
# Process Binding

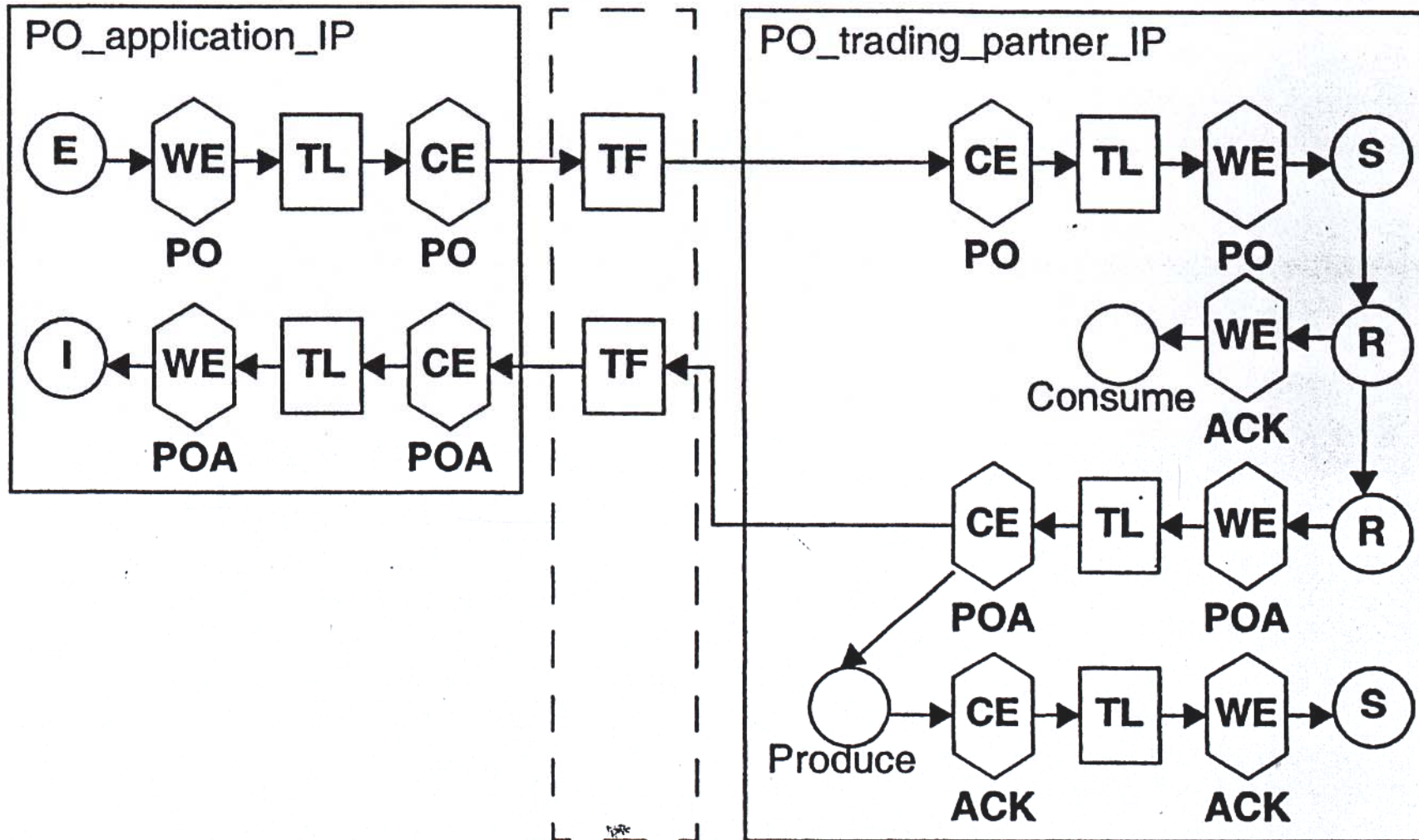


[ ] Process binding



# Process Binding (2)





Interface Processes can also be tied together directly, if no business logic is necessary in between.



# Endpoint Attributes

- Endpoints need identifying arguments like:
  - Names
  - Unique identifiers
  
- Other data like:
  - Public / Private keys
  - etc.



## Endpoint Attributes (2)

- Additionally, endpoint type-specific information has to be kept:
  - Credit ranking
  - Payment history
  - Several business addresses
- It must be possible to add data and new categories at runtime

# Endpoint Capabilities



- This section describes the capabilities of an endpoint in terms of B2B integration.
- For that data has to be stored:
  - Supported B2B protocols
  - Modifications to standards

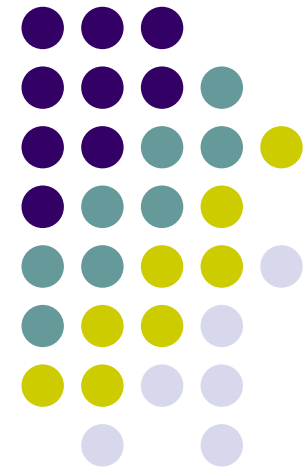


# Endpoint agreements

- For security reasons, not all endpoints should be enabled to send messages to each other automatically.
- For that the construct of Agreements has been introduced.
  - If an endpoint wants to communicate with another endpoint, an Endpoint Agreement is necessary between those two endpoints.

# Integration Technology Architecture

---



## B2B integration technology server

**User interface**

**Integration logic**

**Connectivity**

**Persistence**

- 4 different layers  
In the architecture

- Strict Top-Down



## User Interface

**Modeling**

**Testing**

**Monitoring**

**Analysis**

**Administration**

**Endpoint  
management**

**Error handling**

**Worklist**



## Logic layer

**Event  
Management**

**Process  
management**

**Transformation**

**Translation**

**Data type  
management**



## Connectivity layer

**Back-end application adapters**

**B2B protocol engine**

**Endpoint management**

**Security**

**Packaging**

**Transport**



## Persistence layer

**Database  
system**

**Persistent  
Queuing system**

**File system**

# Conclusion



- B2B Integration technology offers great and flexible possibilities to fulfill most industries dreams of flexibility, adaptability and compatibility.

**Thanks for your attention**