High Level Execution Primitives in MDA

Eugene Wong

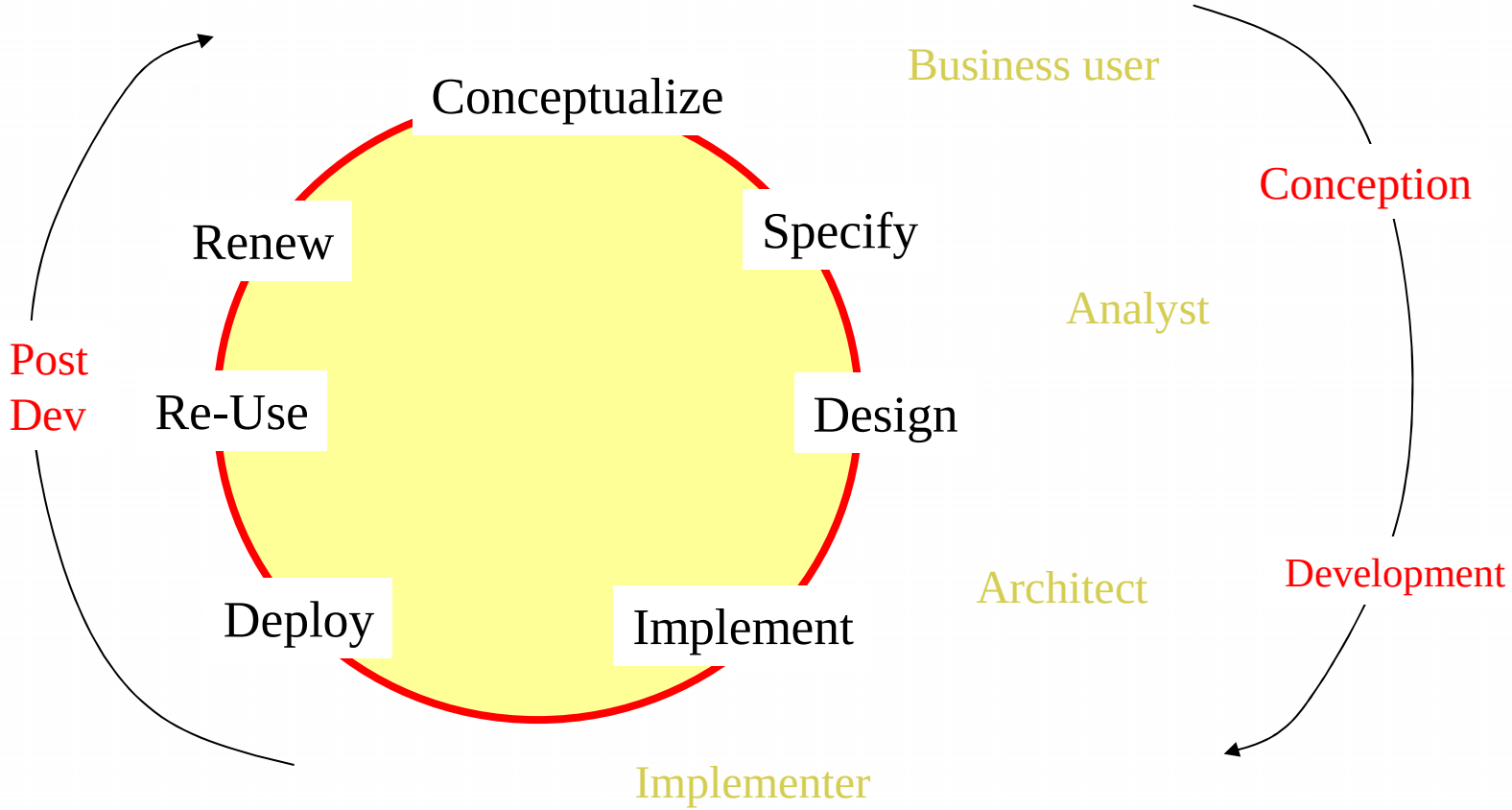
Chief Scientist

Agenda

- Motivation and Goal
- Objectives
- Modeling Paradigm
- High level primitives
  - Transaction
  - Presentation and display

Motivation for MDA

Application Lifecycle Productivity



Goal of Executable MDA

- **Model is the Application**

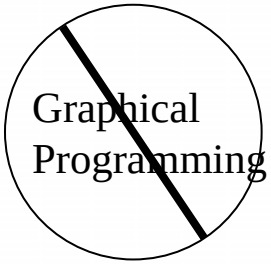
  Modeling = replacement for, not supplement to, development

Challenge

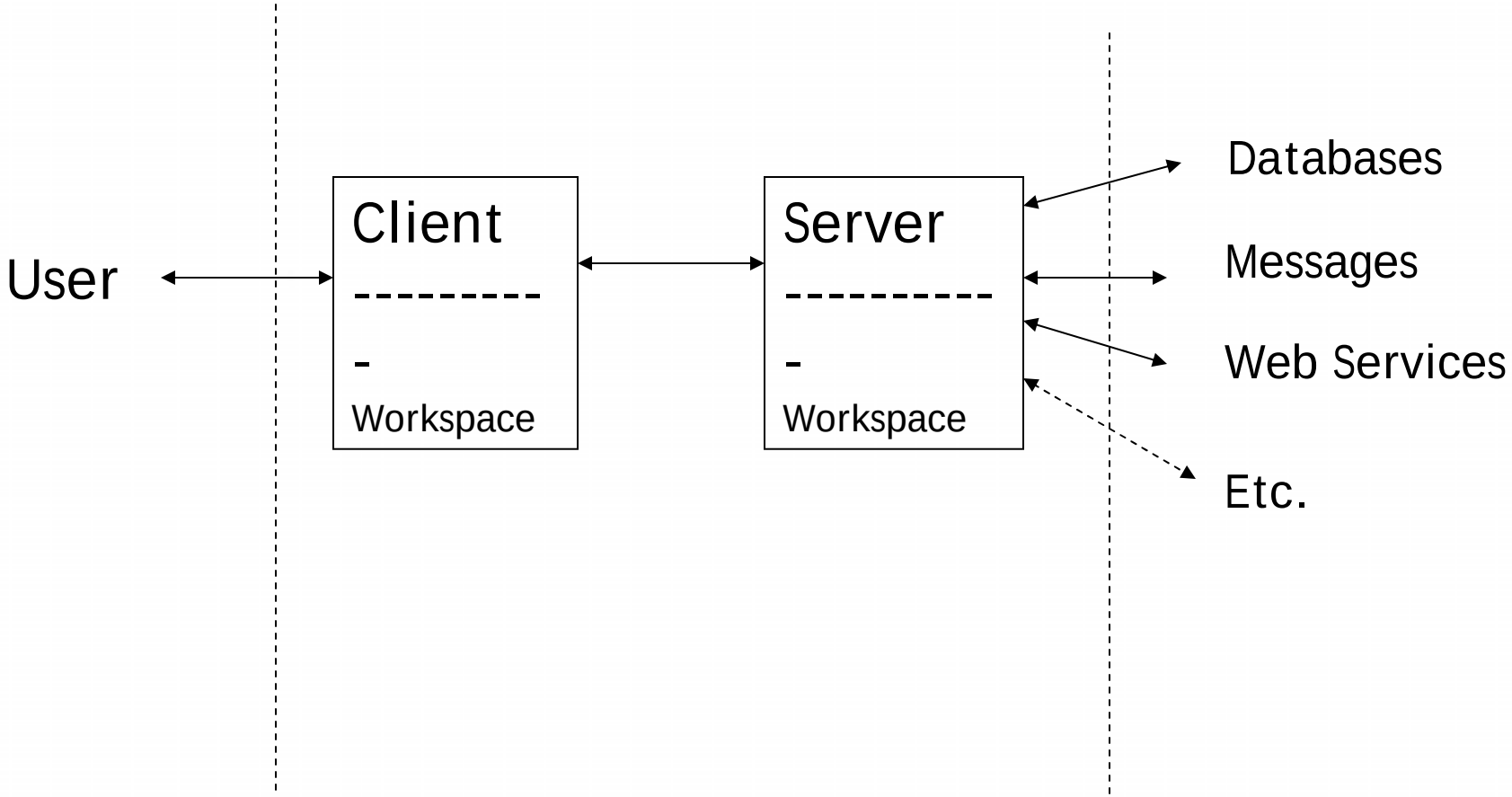
- The model must provide
  - Complete representation
  - Expressive power
    - Declarative
    - Economy of expression
  - Execution efficiency

Objectives

- Multiple levels of abstraction (and detail)
  - Layered and nested
- Simplicity of modeling environment
  - Self-similarity
- Coarse-grain execution ← **Our focus**



General Application Architecture



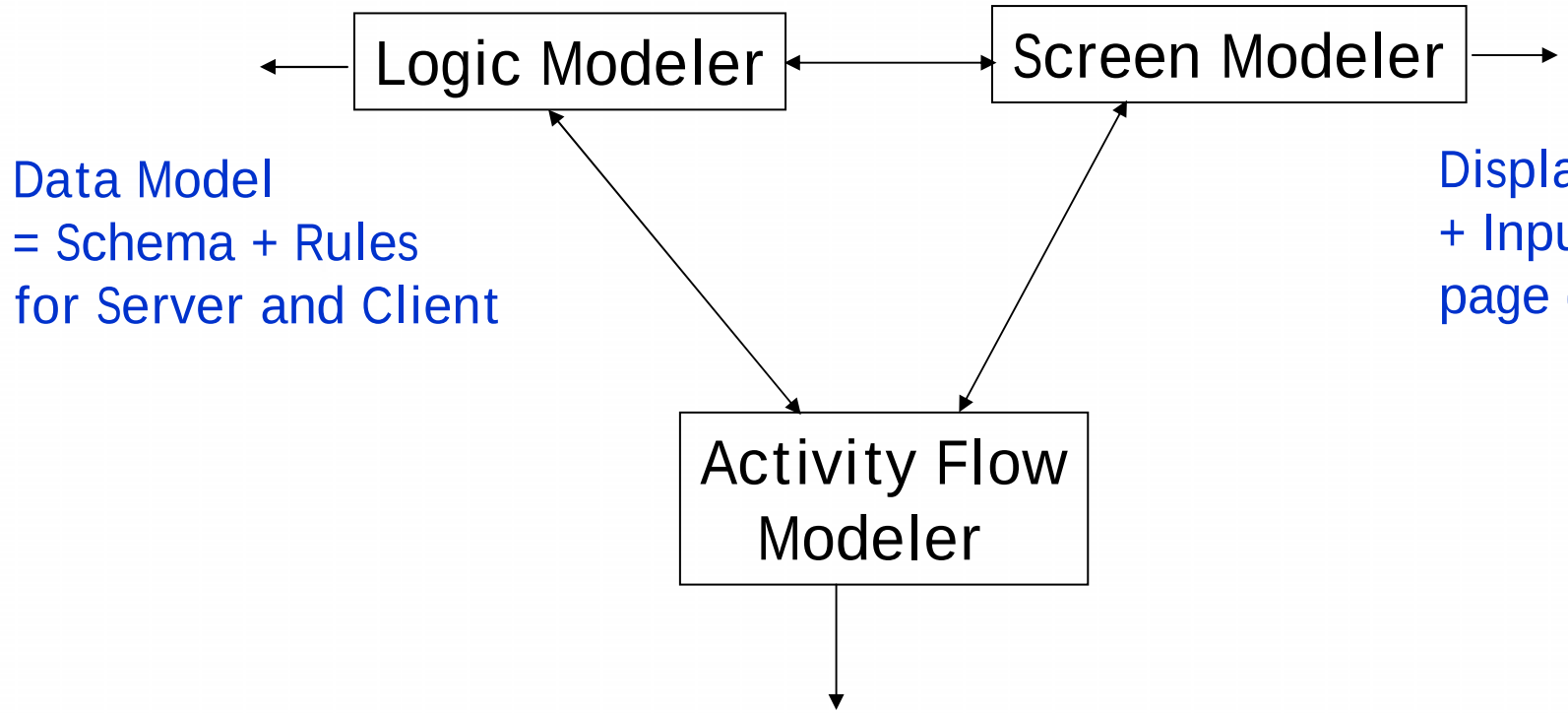
Application Modeling

- Nested Layers of Activity Flow
  - Nesting: decomposing activities
- Stop when primitives are reached

Primitives

- Decisions
- Rules Automated Transactions
- Presentation and display
- Communications
- Pre-built components and services
- Hard Core Indecompsables (hand coding)
  - Must be transaction and communication free

Modeling Studio Organization

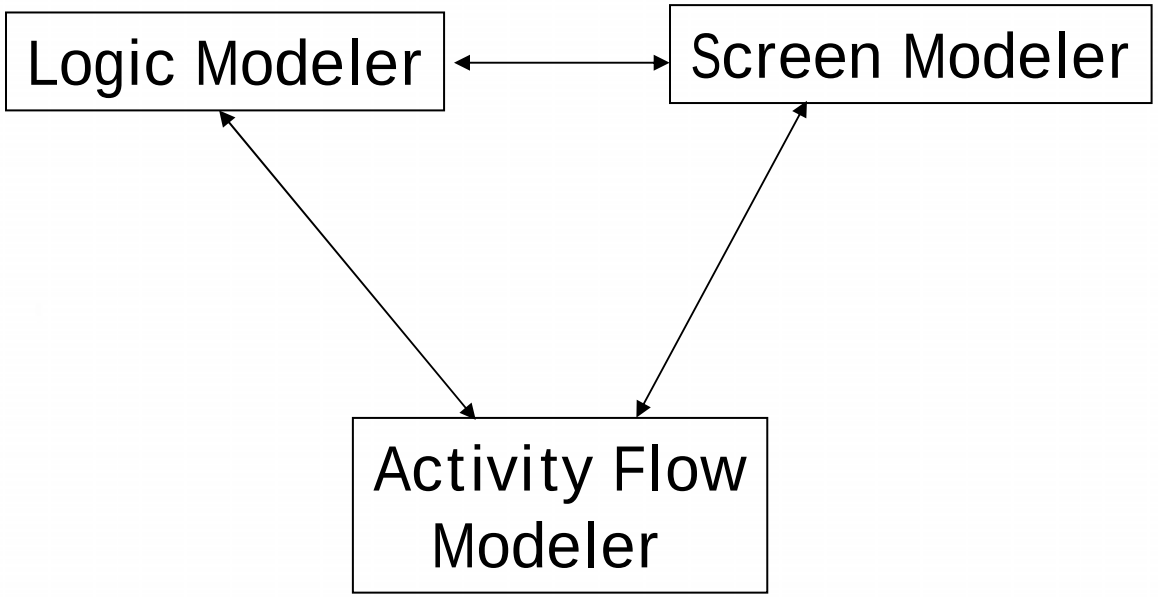


Data Model
= Schema + Rules
for Server and Client

Display model
+ Input to
page generator

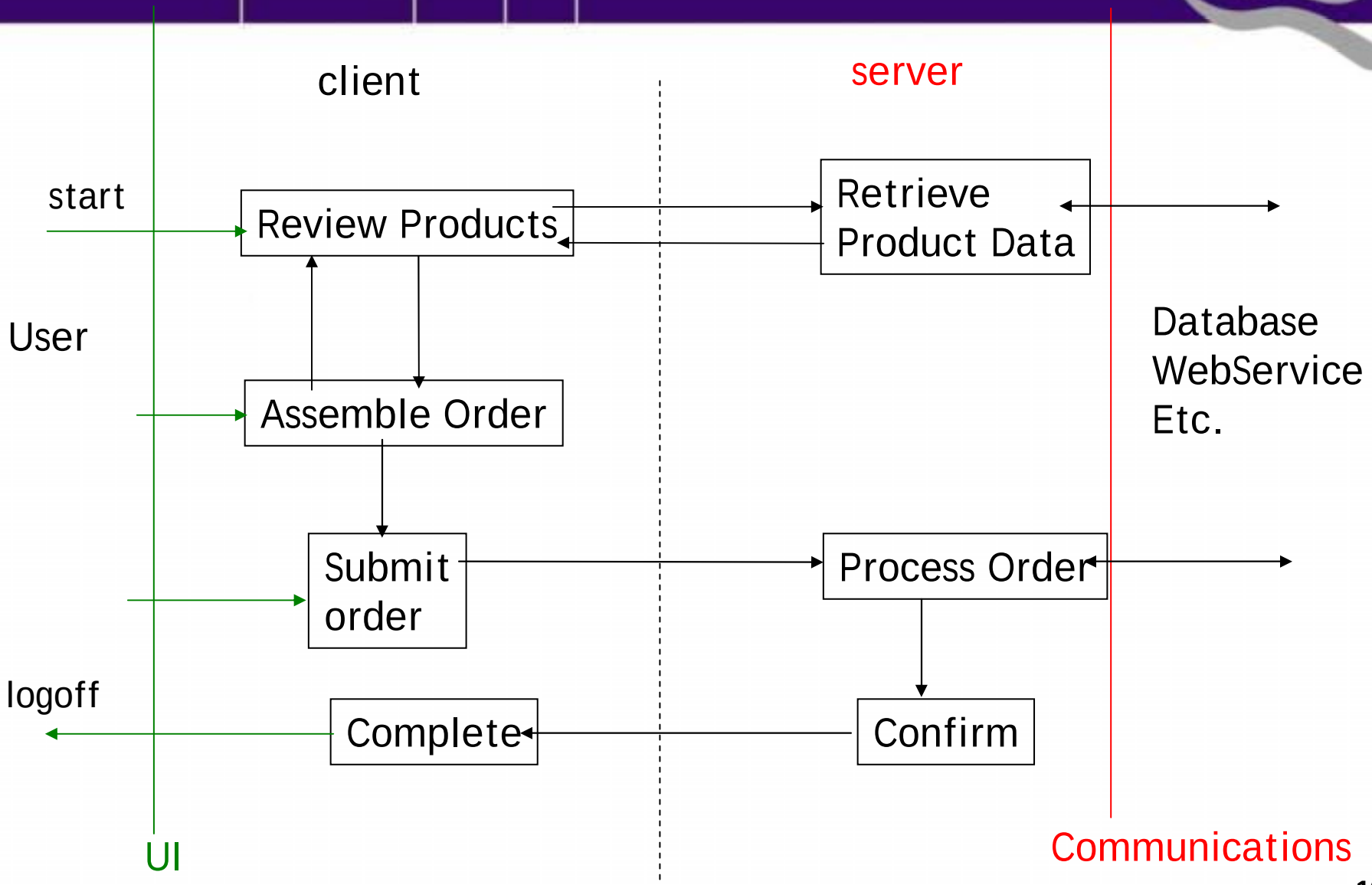
Nested layers
of activity diagrams

Modeling Organization

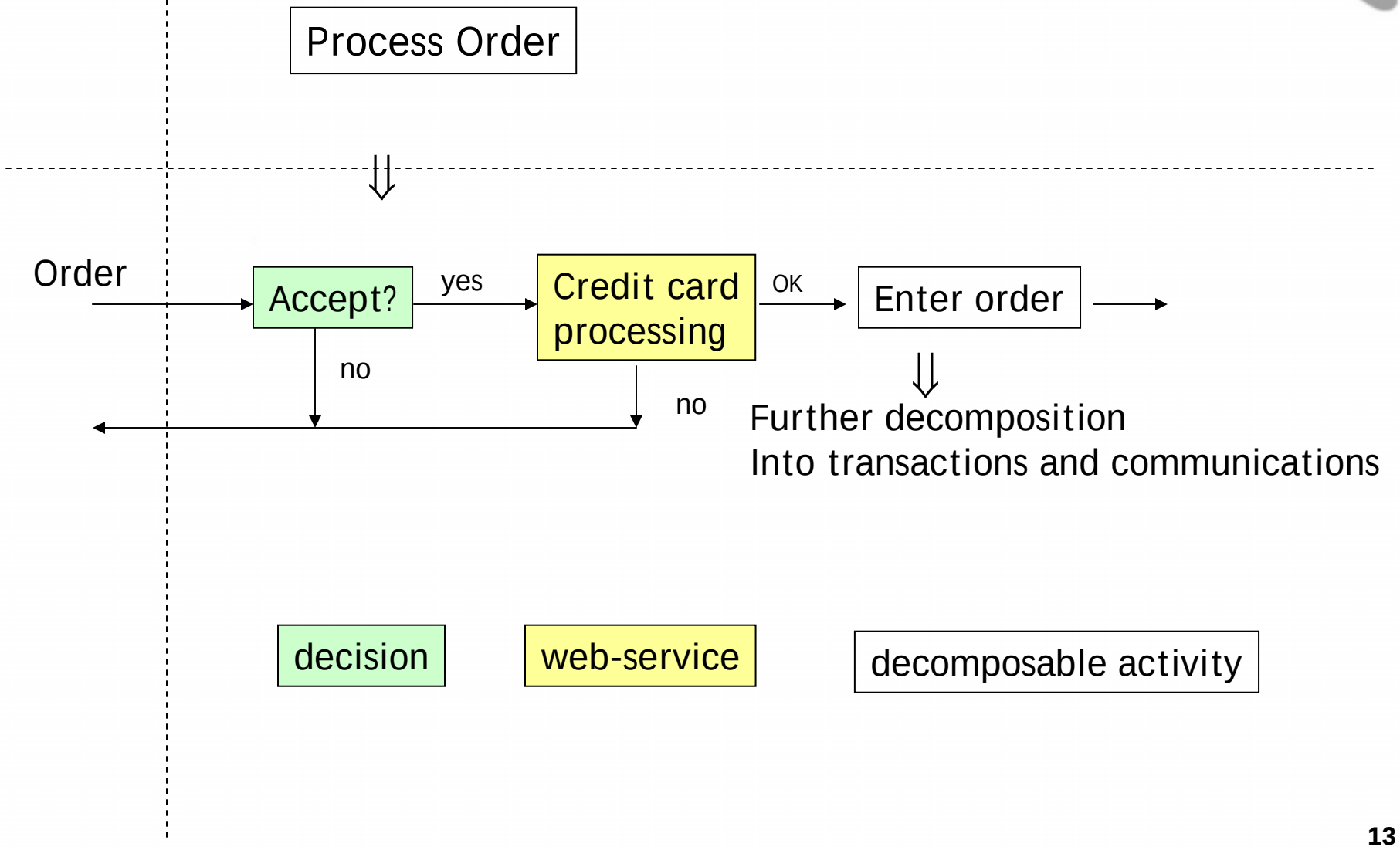


Current Versata product suite provides all three (to some degree) for both design and execution.

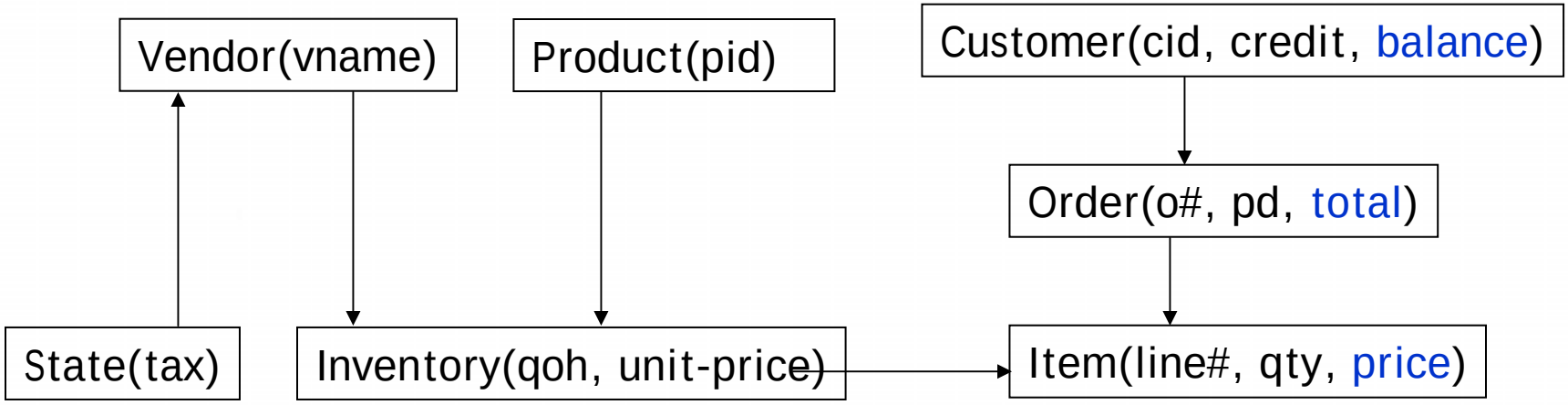
Activity Flow Example: Online Purchase



Decomposition

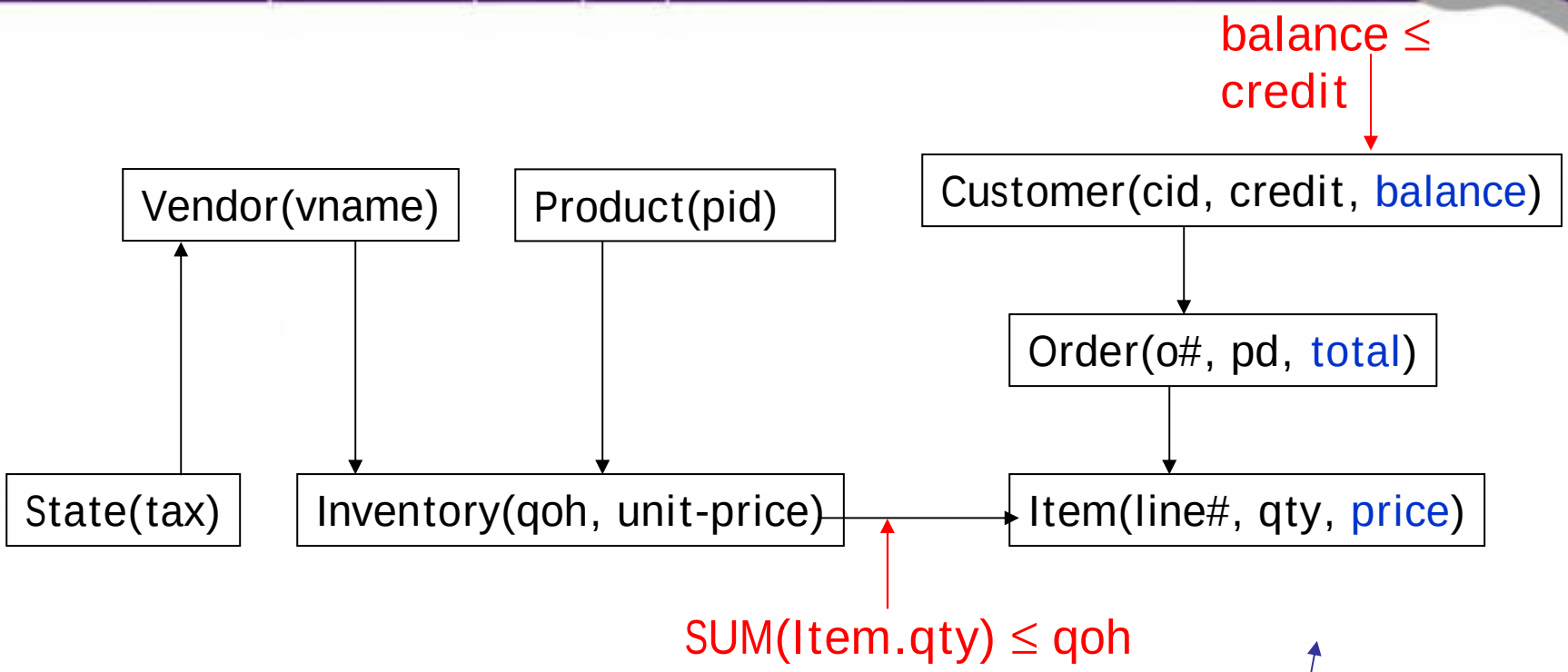


Logic Modeling - Schema



All arcs are associations (1-to-n relationships)
All attributes are automatically inherited

Logic Modeling - Rules

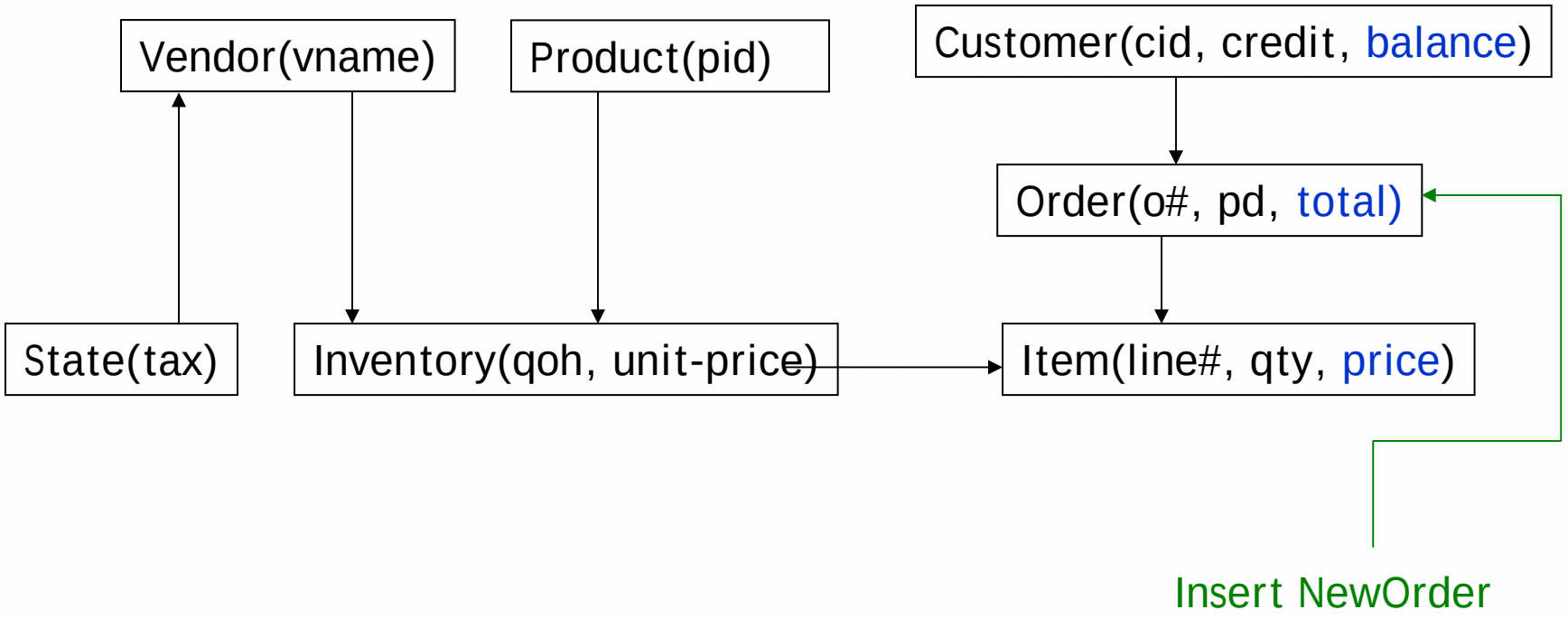


Derivation rules

Constraint rules

price = (1+tax)*qty*unit-price
total = SUM(Item.price)
balance = SUM(Order.total
where pd="no")

Transaction Primitive - Example



Rules Automated Transactions

Transaction modeled at two levels of abstraction

Rules free layer

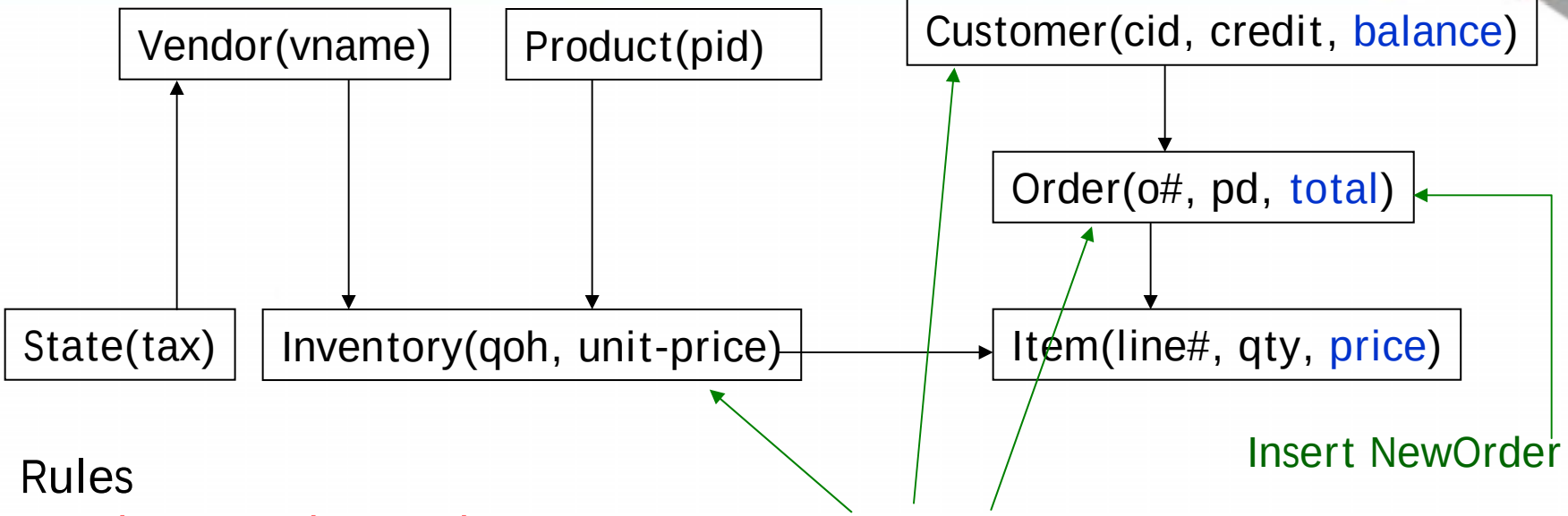
Insert NewOrder

⇓ Decomposition
Automated by rules

Rules automated layer

Verify qoh >= SUM(Item.qty)
Derive New-balance
Verify credit >= New-balance
Replace Customer.New-balance
Insert New-Order

Transaction - Insert Order



Rules

SUM(Item.qty) <= qoh

balance <=
credit

price = (1+tax)*qty*unit-price

total = SUM(Item.price)

balance = SUM(Order.total
where pd="no")

Rules Automated Transaction

Verify qoh >= SUM(Item.qty)

Derive New-balance

Verify credit >= New-balance

Replace Customer.New-balance

Insert New-Order

UI Automation

Principle: Design by type, not instance

Strategy:

Consider UI as transformation of Client-side activity-flow into displayable pages

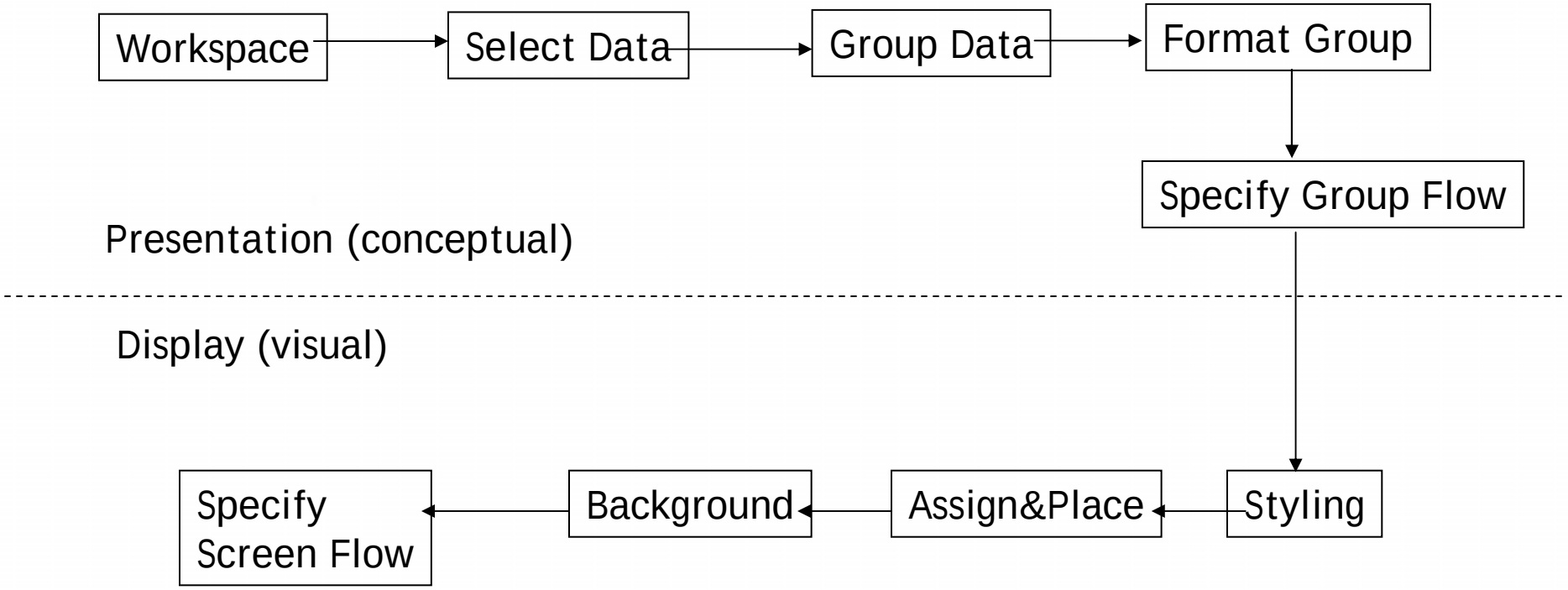
Model:

Explicit representation of the transformation

High Level Primitives:

Pre-built “templates” for each type of data

Presentation and Display Operations



Can loop from any point

Presentation Types

Basic building block = list

List = Ordered collection of items = {items}
(items may repeat)

Record = {(field name, field value)}

Simple list = {records}

Master-Detail = {(record, simple list)}

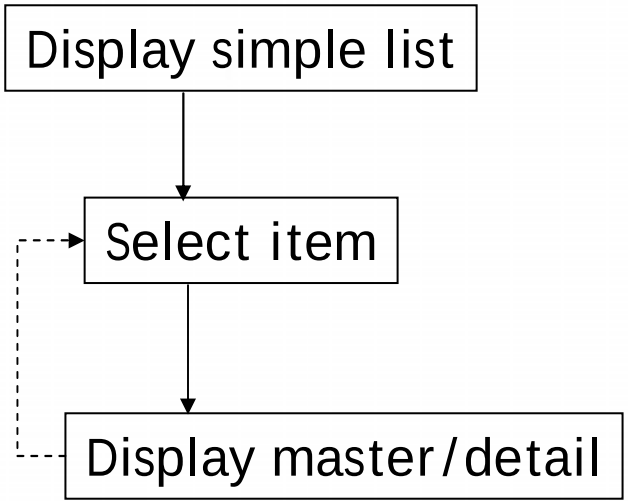
Tree: simple list is a tree
{(root, tree)} is a tree if root = record

(note: tree is the basic data model for XML)

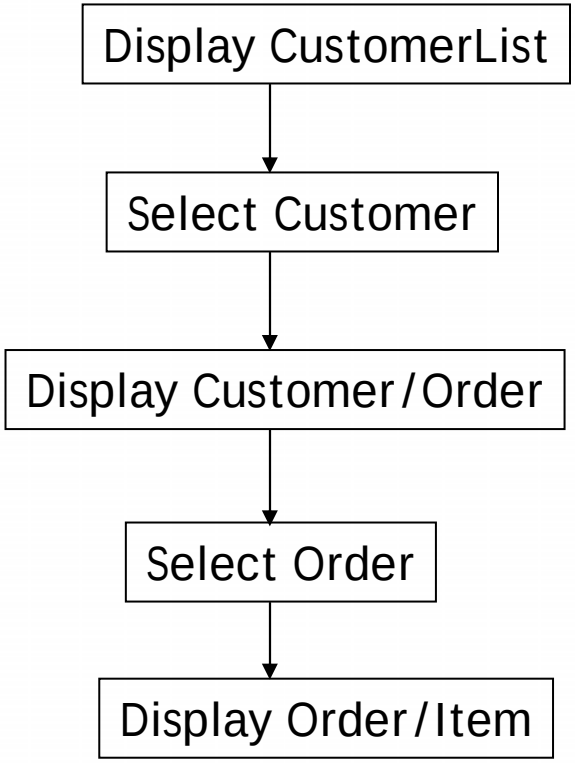
Display

Example

Basic building block

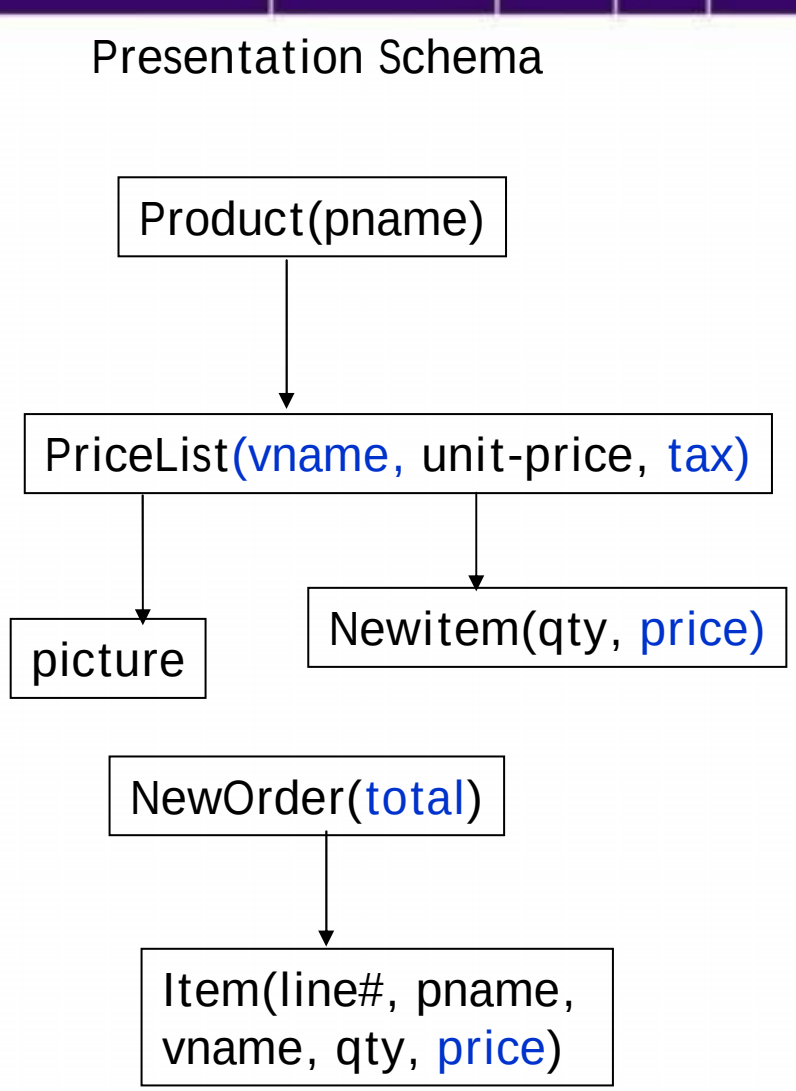


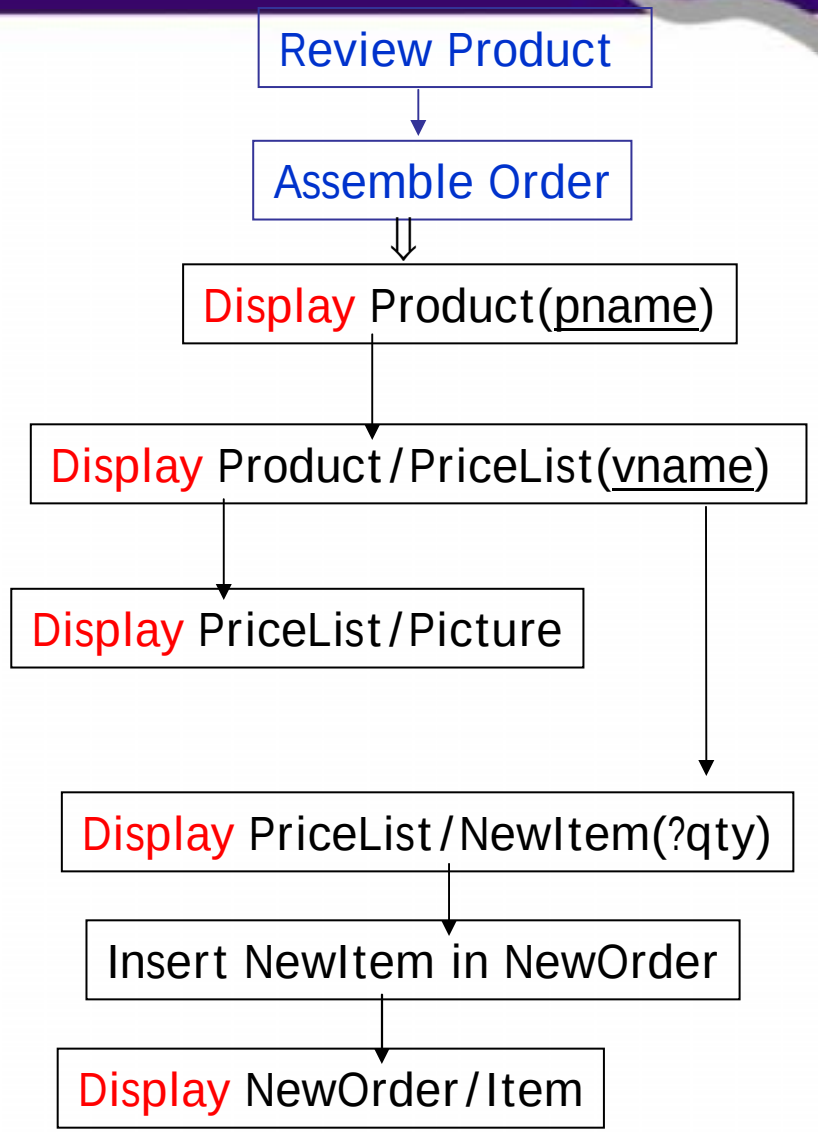
List-to-list mapping
Can be concatenated



Example

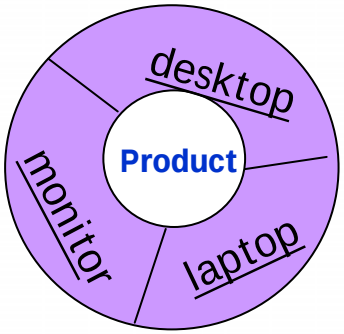
Presentation Schema





Modeling Display

Simple List →

- Column → **Product**: desktop, laptop, monitor
- Row
- Table
- Wheel → 

- Etc.

Master/Detail →

- Master = caption
- Master = highlighted item
- Master = icon
- Etc.

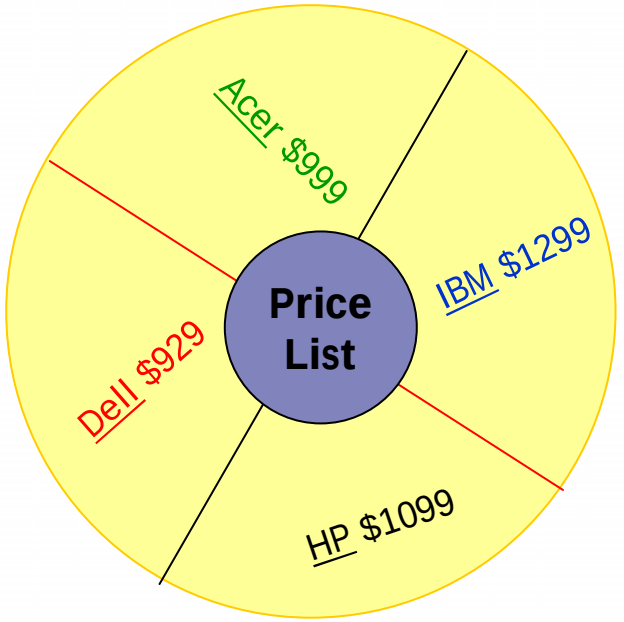
Detail is always a simple list

Screen Flow

Product
Desktop
Laptop
Monitor

→

Product
Desktop
Laptop
Monitor



Screen Flow Continued

Price List

Company	Price
Acer	$999
Dell	$929
HP	$1099
IBM	$1299

Desktop from Dell

Add this to my shopping cart

Screen Flow Continued

New Shopping Cart Item

Please specify quantity

Product	laptop
Vendor:	Dell
Unit-price	$929.00
Quantity	3

Add to my order

My Order

Vendor	Product	Qty	Price
HP	monitor	5	$3250
IBM	desktop	2	$1000
Dell	laptop	3	$2850
		Total:	$7100

Review more products

Submit order

Summary

- Model = Data Model + Activity Flow + UI
- Coarse-Grain Execution:
  - High Level Primitives: transactions + presentation and display
- Transactions: automated by rules
- Presentation and Display:
  - Design by type
    - list and master/detail
- Communications: yet to be modeled
  - XML and tree are key