# 10. Business Process Management

## CSEP 545 Transaction Processing

## Philip A. Bernstein

# Outline

1. Introduction
2. Managing Process State
3. Making a Workflow ACID
4. Other Workflow Models

# 10.1 Introduction

- Business process - a partially ordered set of a steps, where each step performs an administrative function usually by accessing a shared database.

- Examples – place an order, reserve a trip, buy a house, adjust an insurance claim

- Each step may be a transaction, an execution of a program that is not a transaction, or a manual activity performed by a person.

- Examples – debit an account, approve a large debit

# Business Process Management

- Business process management is the activity of creating, managing, adapting, and monitoring business processes.

- Most of this is business management, not necessarily a technical activity
  - Analyzing business processes
  - Defining improved processes
  - Which eventually affects requirements for transactions and other TP mechanisms.

# Business Process Specification

- Flowgraph language for describing processes consisting of steps, with preconditions for moving between steps
  - Some people recommend state machines, but imperative languages are more popular.

- Representation of organizational structure and roles
  - a step can be performed by a person in a role, with a (possibly complex) role resolution procedure

- Choreography - a message protocol between independent business processes

# Business Process → Many ACID Txns

- Some requests cannot execute as one transaction because
  - It executes too long (causing lock contention) or
  - Resources don't support a compatible 2-phase commit protocol.

- A transaction may run too long because
  - it requires display I/O with user
  - people or machines are unavailable (a step that includes manager approval, or a billing step that runs in batch)
  - it requires long-running real-world actions (get two estimates before settling an insurance claim)

- Steps may require independent ACID transactions in different subsystems (capture an order, schedule a shipment, report commission, send an invoice)
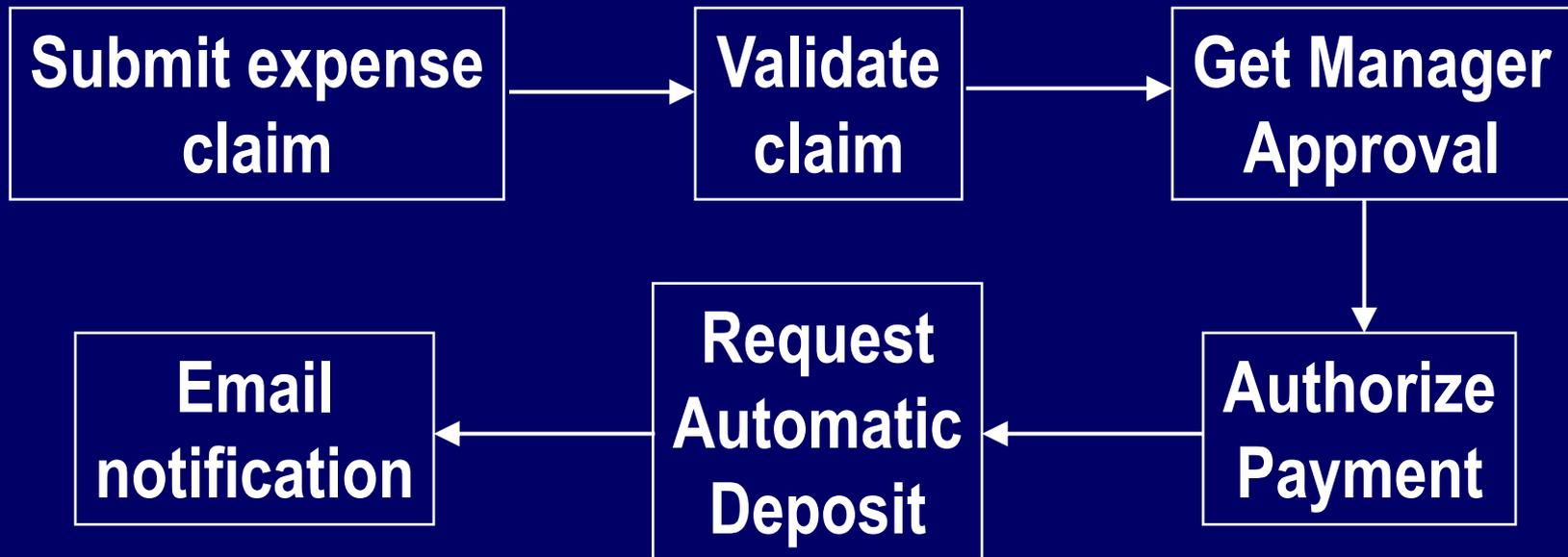
# Workflow

- Workflow - A technology to enable the execution of long running, multi-transaction requests.
  - Long running → manage process state recoverably
  - Multi-txn → mechanisms for isolation and atomicity

- Textbook says BPM and workflow are synonyms

- But often, BPM refers to the business activity and workflow to the technical implementation
  - This terminology distinction isn't universally used

# 10.2 Managing Process State

- Since processes can execute for a long time (weeks), you need state management
  - Save state persistently (when process is idle) and restore it later (when it becomes active again)
  - Find the state of process (which might be inactive)
- Process state – data and control state
- User wants to know which steps ran (with what inputs and outputs) and which are next to run
  - Log all interesting events and make them queryable
- Usually requires a workflow-specific run time

# Managing Workflow with Queues

- Each workflow step is a request. Send the request to the queue of the server that can process the request
- Server outputs request(s) for the next step(s) of the workflow
- May be hard to answer a query about workflow state

```
┌──────────────┐      ┌──────────┐      ┌──────────────┐
│ Submit expense│ ──→  │ Validate │ ──→  │ Get Manager  │
│    claim     │      │  claim   │      │  Approval    │
└──────────────┘      └──────────┘      └──────────────┘
                                                │
                                                ▼
┌──────────────┐      ┌──────────┐      ┌──────────────┐
│    Email     │ ←──  │ Request  │ ←──  │  Authorize   │
│ notification │      │Automatic │      │   Payment    │
│              │      │ Deposit  │      │              │
└──────────────┘      └──────────┘      └──────────────┘
```

# Pseudo-conversations

- Simple solution to manage state in early TP systems
- A <u>conversational</u> transaction interacts with its user during its execution
- This is a sequential workflow between user & server.
- Since this is long-running, run it as multiple requests
- Since there are exactly two participants, just pass the request back and forth
  - request carries all workflow context
  - request is recoverable, e.g. send/receive is logged or request is stored in stable storage
- This simple mechanism has been superseded by queues and general-purpose workflow systems.

# Other Approaches to State Mgmt

- Queue elements and pseudo-conversation requests are places for persistent workflow state. Other examples:
  - Browser cookies (files that are read/written by http requests), containing user profile information
  - Shopping cart (in web server cache or database)

- Such state management arises <u>within</u> a transaction too
  - Server scans a file. Each time it hits a relevant record, return it.
  - Issue: later calls must go to the same server, since it knows where the transaction's last call left off.
  - Sol'n 1: keep state in the message (like pseudo-conversation)
  - Sol'n 2: first call gets a binding handle to the server, so later calls go to it. Server needs to release state when client disappears

# 10.3 Making a Workflow ACID

- If a workflow runs as many transactions,
  - it may not be serializable relative to other workflows (i.e., not isolated)
  - it may not be all-or-nothing (i.e., not atomic)
- Suppose a workflow auto-pays a credit card
  - $T_1$ debits checking and $T_2$ credits the card
  - Not Isolated - A query could run in between, looking for accounts where card debit exceeds checking balance.
  - Not atomic - A failure after $T_1$ might prevent $T_2$ from running.

# Making a Workflow ACID (cont'd)

- These problems require app-specific logic.

- Isolation – App must understand that some money could be in flight.

- Atomicity - $T_2$ sends ack to $T_1$'s node. If $T_1$'s node times out waiting for the ack, it takes action, possibly compensating for $T_1$

# Automated Compensation

- Each step in a workflow program identifies a compensation. This his called a <u>saga</u>.

- If a workflow stops making progress, the workflow system runs compensations for all committed steps, in reverse order (like transaction abort).

- Need to ensure that each compensation's input is available (e.g. log it) and that it definitely can run (enforce constraints until workflow completes).

# 10.4 Other Workflow Models

- Scientific workflow
  - Use a workflow definition to drive an experiment
  - Review history of executions (provenance)
  - Capture sequence of steps for replay
- Configuration management
  - Check-out and Check-in of engineering docs or code
  - Can include customizable engineering process
  - Similar functions for managing system configuration tasks (e.g., how to provision a server)

# Products

- IBM MQSeries Workflow
- MS BizTalk Orchestration
- MS SQL Server Service Broker

- JetForm
- TIBCO
- BEA WebLogic Process Integrator

- See also www.workflowsoftware.com, www.wfmc.org