# REVISION SDP5

Zhanerke Gabidenkyzy

# THREE TYPES OF DESIGN?

# THREE TYPES OF DESIGN?

- Evolutionary
- Planned
- Minimal planned

XP?

# SOFTWARE DEVELOPMENT PROCESSES

- Examples:
- 1.
- 2.
- 3.
- 4.
- 5.

# SOFTWARE DEVELOPMENT PROCESSES

| Process | Up-front design | Nature of design | Prioritization of work | Iteration length |
|---|---|---|---|---|
| Waterfall | In analysis & design phases | Planned design; no redesign | Open | Open |
| Iterative | Optional | Planned or evolutionary; redesign allowed | Open, often feature-centric | Open, usually 1-8 weeks |
| Spiral | None | Planned or evolutionary | Riskiest work first | Open |
| UP / RUP | Optional; design activities front-loaded | Planned or evolutionary | Riskiest work first, then highest value | Usually 2-6 weeks |
| XP | None, but some do in iteration zero | Evolutionary design | Highest customer value first | Usually 2-6 weeks |

Figure 3.4: Examples of software development processes and how they treat design issues. For comparison purposes, a waterfall process is treated as having a single long iteration.

# RISK-DRIVEN MODEL

Steps:

1.
2.
3.

# RISK-DRIVEN MODEL

- Steps:
- 1. Identify and Prioritize Risks
- 2. Select and Apply a Set of Techniques
- 3. Evaluate Risk Reduction

NAME 3 OF THE ARCHITECTURAL STYLES DESCRIBED IN FAIRBANKS AND BRIEFLY DESCRIBE EACH?

# NAME 3 OF THE ARCHITECTURAL STYLES DESCRIBED IN FAIRBANKS AND BRIEFLY DESCRIBE EACH?

| | Viewtype | Elements & Relations | Constraints / guide rails | Qualities Promoted |
|---|---|---|---|---|
| **Layered** | Module | Layers, uses relationship, callback channels | Can only use adjacent lower layers | Modifiabilty, portability, reusability |
| **Big Ball of Mud** | Module | None | None | None, but many inhibited |
| **Pipe-and-Filter** | Runtime | Pipe connector, filter component, read & write ports | Independent filters, incremental processing | Reconfigurability (modifiability), reusability |
| **Batch-Sequential** | Runtime | Stages (steps), jobs (batches) | Independent stages, non-incremental processing | Reusability, modifiability |
| **Model-Centered (Shared Data)** | Runtime | Model, view, and controller components; update and notify ports | Views and controllers interact only via the model | Modifiability, extensibility, concurrency |
| **Publish-Subscribe** | Runtime | Publish and subscribe ports, event bus connector | Event producers and consumers are oblivious | Maintainability, evolvability |
| **Client-Server & N-Tier** | Runtime | Client and server components, request-reply connectors | Asymmetrical relationship, server independence | Maintainability, evolvability, legacy integration |
| **Peer-to-Peer** | Runtime | Peer components, request-reply connectors | Egalitarian peer relationship, all nodes clients and servers | Availability, resiliency, scalability, extensibility |
| **Map-Reduce** | Runtime & allocation | Master, map, and reduce workers; local and global filesystem connectors | Divisible dataset amenable to map & reduce functions, allocation topology | Scalability, performance, availability |
| **Mirrored, Farm, & Rack** | Allocation | Varies | Varies | Varies: Performance, availability |

Figure 14.1: A condensed summary of the architecture styles in this chapter. Refer to text for a full listing of elements, relationships, constraints, and quality attributes.

# CANONICAL MODEL STRUCTURE

- Models:
- 1.
- 2.
- 3.

# CANONICAL MODEL STRUCTURE

- Models:
- 1. Domain
- 2. Design
- 3. Code

- Architecture
- Architecting
- Architect

- Designation
- Refinement