CS360
Business Information Systems
Analysis and Modeling

Project Management, JAD, Agile
Risk = Stress

* Projects are Complex in themselves
  * requirement discovery
  * client personalities
  * technology diversity and complexity
  * time and resource constraints

* Work + Stress Reduces Efficiency
  * Decisions require clear information
  * Information comes from careful investigation
  * Surprises limit investigation before a decision
  * Decisions based on surprises are risky
  * Projects full of surprises are very stressful

* ∆ Controlled Risk = Improved Efficiency and Effectiveness
  * Manage Risk by Avoiding Surprises
Managing Cost

Projects Consume Resources
- Time
- Personnel
- Money

Projects are Organized Around Tasks
- Tasks are clearly identified Units of Work
- \( \text{Person} + \text{“Money”} + \text{Time} = \text{Task Accomplished} \)
- Tasks prepare the groundwork for other tasks
- Tasks cannot begin until the groundwork is laid

Project Management Tracks Resources
- Time and Money are expended only once
- Persons can only do one task at a time
- Tasks may be accomplished at the same time
- Arranging the combination of Persons to Tasks at the right time and with the right “money”
Project Money Estimation

* Budgeting is Estimating + Measuring
* Costs
  * Hardware: new, used, refurbished
  * Software: new, upgraded, tailor made
  * Labor: professional, amateur, “farmed out”
* Estimation Resources
  * Hardware: vendors, advertising, agents
  * Software: vendors, dealers, bundles, upgrades
  * Labor
    * “Open Ended” - by the hour
    * “Fixed” - by the task or job
* Estimation Approaches
  * “Time and Materials”
  * “Two times materials”
  * “Top Down”
  * “Sum of the tasks”
## Silhouette Project Budget

<table>
<thead>
<tr>
<th></th>
<th>Month 1</th>
<th>Month 2</th>
<th>Month 3</th>
<th>Month 4</th>
<th>Totals</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Estimates</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>0</td>
<td>2000</td>
<td>2000</td>
<td>0</td>
<td>4000</td>
</tr>
<tr>
<td>Software</td>
<td>500</td>
<td>500</td>
<td>500</td>
<td>0</td>
<td>1500</td>
</tr>
<tr>
<td>Labor</td>
<td>1000</td>
<td>1500</td>
<td>1000</td>
<td>1000</td>
<td>4500</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>1500</td>
<td>4000</td>
<td>3500</td>
<td>1000</td>
<td>10000</td>
</tr>
<tr>
<td><strong>Actual</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>0</td>
<td>3000</td>
<td></td>
<td></td>
<td>3000</td>
</tr>
<tr>
<td>Software</td>
<td>400</td>
<td>500</td>
<td></td>
<td></td>
<td>900</td>
</tr>
<tr>
<td>Labor</td>
<td>1200</td>
<td>1200</td>
<td></td>
<td></td>
<td>2400</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>1600</td>
<td>4700</td>
<td></td>
<td></td>
<td>6300</td>
</tr>
<tr>
<td><strong>Monthly +/-</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>0</td>
<td>-1000</td>
<td></td>
<td></td>
<td>-1000</td>
</tr>
<tr>
<td>Software</td>
<td>100</td>
<td>0</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Labor</td>
<td>-200</td>
<td>300</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>-100</td>
<td>-700</td>
<td></td>
<td></td>
<td>-800</td>
</tr>
<tr>
<td><strong>Cum. +/-</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>0</td>
<td>-1000</td>
<td></td>
<td></td>
<td>-1000</td>
</tr>
<tr>
<td>Software</td>
<td>100</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Labor</td>
<td>-200</td>
<td>100</td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>-100</td>
<td>-800</td>
<td></td>
<td></td>
<td>-800</td>
</tr>
</tbody>
</table>
### Managing Time and Tasks

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Planned Start</th>
<th>Planned Finish</th>
<th>% Done</th>
<th>February, 1998</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>2/1/98</td>
<td>2/4/98</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>2/4/98</td>
<td>2/8/98</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>Development</td>
<td>2/8/98</td>
<td>2/12/98</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Implement.</td>
<td>2/12/98</td>
<td>2/16/98</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Contract</td>
<td>2/1/98</td>
<td>2/1/98</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Preliminary Presentation</td>
<td>2/3/98</td>
<td>2/3/98</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Design Review</td>
<td>2/8/98</td>
<td>2/8/98</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>Prototype Review</td>
<td>2/9/98</td>
<td>2/9/98</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td>2/12/98</td>
<td>2/13/98</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

#### Gantt Chart

- **Planned**: Yellow
- **Actual**: Red
- **% Done**: Green
- **Slack Available**: Light Blue

- **Task Begin or Task End**: Yellow Diamond
Budget and Status Reports

- Develop gross estimates
  - hardware and software based on constraints and similar projects
- Use top-down or bottom up estimation
- Estimate labor costs
  - with bottom up it’s the sum of the tasks defined
  - with top down some breakdown is needed for clarity
- Spread the cost along the timeline
- Build a budget
- Build a status sheet
- Collect billable hours regularly
- Post the actual against the projected
Managing Task Flow

**Program Evaluation and Review Technique, “PERT”**
- identifies project work in tasks
- defines the precedence and dependency of tasks
- determines the “time path” for task completion
- identifies “critical path tasks”
- is readily supported by automated tools !!!

**Events, Tasks, Dependencies**
- Events represent identifiable milestones
  - completion of a report, preparation, or client sign-off
- Tasks represent work
  - defines nature and duration of the work (and resources)
- Dependencies
  - identifies the allowable sequence of tasks
  - exposes tasks which may be accomplished in parallel !
CIS Lab PERT Chart

- Events are connected by tasks with defined durations
- Multiple dependencies are noted as dots
Critical Path

The sequence of tasks that any of which are delayed will cause the project completion to be delayed as well!
Projects accumulate artifacts with time
- documentation
- diagrams
- interview / survey notes
- reports: analysis, design, proposals, changes

Documents lost cost “Twice the Price!!”

Project Dictionary / Directory catalogs the project artifacts for later use
- during later phases of the same project
- during maintenance of the system after install
- in subsequent projects to compare issues / costs
- to acclimate new project team members along the way

The project dictionary protects the documentation investment for analyst and client!
Project Architecture

* IS projects are about "building things"
  * requirement specification
  * system designs
  * computer systems software
  * testing plans

* Project architecture is about the building PROCESS
  * designation of team responsibilities
  * identification of task distribution
  * establishing team-wise and step-wise means of progress
  * providing means of determining progress
  * risk management
  * resource control / accountability
JAD, Joint Application Design

* JAD attempts to produce the analysis & design products in a concentrated and abbreviated process involving analysts and users

* Revolving around the User System Diagram (as the shared vision of the system), the group evolves a system skeleton from the individual model products of Use Case, Class Diagram, Sequence Diagrams, ERD and USD.

* Formal JAD accomplishes design as a “Blitzkrieg”

* Informal JAD accelerates the cycle of analysis / user review but, is more “casual” and less intense for the user
Formal JAD

* Formal JAD is intended to substitute for the classic development cycle
  * Participants: Users, Analysts, Observers, Scribe and Facilitator
  * Setting: outside the normal work environment, group decision making work setting, A/V aids to brainstorm and idea capture as a group
  * Structured agenda
    * evaluate the existing system models and documentation
    * agree on the new system goals
    * develop alternative designs
    * select the “best”
  * JAD products:
    * background summary
    * I/O interface designs and definition
    * menu tree and GUI design
    * processing rules
    * system operating procedures
JAD Products

* JAD produces . . .
  * All the user recognizable model products
  * Use Case, Class, Sequence, Activity diagrams
  * ERD for persistent data objects
  * Menu Tree
  * GUID
  * USD
  * JAD does not produce
    * the detailed data model
    * the detailed processing model
    * the detailed system model
  * JAD does define the interfaces that the user will expect to see in prototypes and user documentation
  * Eventual detailed design decisions that impact any of these JAD products will cause the JAD products to be revisited with the user / client
“Coarse” JAD Steps

“Season to taste...”

**Step 1: Sketch the Class Diagram**
- using the system narrative draft Use Cases as a baseline, then sketch out the primary classes that support the business rules
- validate the consistency of the use cases and business classes with the business rules

**Step 2: Sketch the ERD’s**
- Persistent data classes evolve as Entities
- Persistent objects, PO’s, require “keys” when databases are involved
- Normalization completes the PO design process

**Step 3: Sketch the GUID’s**
- Control flow requires menu / command GUI’s
- Data entry / retrieval requires data store access
- Access depends on ENTITY queries (“joins”)
- All data entering/emerging from GUI’s must exist in persistent objects
- All scenarios in the system narrative are covered
AGILE DEVELOPMENT

VALUES
- adaptability
- transparency
- simplicity
- unity

Strategic Vision
- charter
- funding
- estimation

Objectives
- goals
- release plan
- review

Agility is...
- iteration
- standup
- integration
- refactoring

Visibility
- velocity
- burndown

Continuous Improvement
- TDD
- build
- collaboration

Daily
- acceptance
- retrospective

Release
- acceptance
- integration

Working Software
- iteration plan
- backlog

Accelerate Delivery
- tests
- tests

Requirements Engineering Slides Four: 17
Agile Manifesto

1. Customer satisfaction by rapid delivery of useful software
2. Welcome changing requirements, even late in development
3. Working software is delivered frequently (weeks rather than months)
4. Working software is the principal measure of progress
5. Sustainable development, able to maintain a constant pace
6. Close, daily co-operation between business people and developers
7. Face-to-face conversation is the best form of communication (co-location)
8. Projects are built around motivated individuals, who should be trusted
9. Continuous attention to technical excellence and good design
10. Simplicity
11. Self-organizing teams
12. Regular adaptation to changing circumstances

SCRUM Architecture

- **Product Owner** role
- **Sprint Review** ceremony
- **Sprint Planning** ceremony
- **Daily SCRUM Meeting** ceremony
- **Burn Down Chart** artifact
- **Sprint Deliverable** artifact
- **Product Backlog** artifact
- **Sprint Backlog** artifact
- Collaboration
- Artifact Flow