

# ***Software Project Planning: Iterative Planning and Tracking***

## ***Software Project Planning: Iterative Planning and Tracking***

Johanna Rothman  
Rothman Consulting Group, Inc.  
781-641-4046  
jr@jrothman.com  
www.jrothman.com

## ***Software Project Management***

2

- 
- Software projects are hard to plan and manage
    - ♦ May not know exactly what we are planning
    - ♦ May not know how to estimate
  - Typical choices for a project in trouble
    - ♦ Slip the schedule
    - ♦ Reduce feature set

## ***Iterative Project Planning and Tracking***

---

3

- Separate features into chunks of work
- Review all critical paths
- Plan to replan
  - ◆ Use inch-pebbles where useful

## ***Initial Project Planning***

---

4

- Senior management fixes the ship dates
- We plan it
  - ◆ We try to do it anyway
- Plan to iterate
  - ◆ Start early on what you know
  - ◆ Don't get concerned about you don't know at the beginning

## ***Critical Paths***

---

- Task
- People
- Resources
- Project's critical path flows through all three areas

## ***Traditionally-planned Projects***

---

- Assume
  - ♦ All features (requirements) are known
  - ♦ Risks and implications are known
  - ♦ All critical paths are known
- What happens when Murphy shows up?
  - ♦ New tasks
  - ♦ New expertise
  - ♦ New resources

## ***Iterative Planning***

---

- Plan major milestones
  - ◆ Freezes
    - Feature freeze
    - Code freeze
    - System test freeze
  - ◆ Beta ship
  - ◆ Product ship
- Agree on criteria for when these milestones occur
- Plan each project segment as know more
- Iterate on the future plan between milestones

## ***Benefits of Iterative Planning***

---

- Reduces uncertainty
  - ◆ Does not make uncertainty appear certain
- Plan in advance where possible
  - ◆ Pre-beta tasks
  - ◆ Pre-ship tasks
  - ◆ The “whats”, even if you don’t know the “how-manys”
- May reduce task slips
- Task slips
  - ◆ Increase risk
  - ◆ Reduce mitigation capabilities
  - ◆ Reduce ability to replan

## ***Planning and Replanning a Project Phase***

---

9

- Create an overall plan
- Plan current stage in detail
- Plan next stage in some detail
  - ♦ Iterate on the planning for the next stage
  - ♦ By the end of current stage, next stage is completely planned
- Take advantage of project advances
  - ♦ Smaller task lists help people make progress
  - ♦ Task completion leads to high productivity

## ***Traditional Schedules Waste Project Advances***

---

10

- “Student Syndrome”
  - ♦ Wait until just before the task is due to start it
- Context switching wastes people’s time
  - ♦ Get less than 50% of a person when you work on two projects simultaneously
  - ♦ The more projects, the lower the productivity
- Silent dependencies waste time, people, and resources
  - ♦ Waiting for a key resource: person, hardware may be a silent dependency

## ***Replanning***

---

- For delays and advances
- Replan the critical paths
  - ♦ Take tasks, people, resources into consideration
- Replan in detail where possible
  - ♦ Use inch-pebbles

## ***Inch-pebbles***

---

- Break each task down into very small components
  - ♦ Some people create inch-pebbles of 3-4 days in duration
  - ♦ I plan tasks of no more than 1-2 days duration
- Each task is either done or not done
  - ♦ Binary decision
  - ♦ No percentage complete
- If you have to show percentages
  - ♦ Gather a number of tasks into the main task
  - ♦ Report on the percentage of main task, by task completion, not by time used

### ***Use Inch-Pebbles When***

---

- You know what you have to do to
- Risks are in getting the work done, not in knowing what you have to do

### ***Benefits of Using Inch-Pebbles***

---

- Inch-pebbles reduce micro-management
  - ♦ The work is either done or not, no one has to keep asking
- Planning
  - ♦ Most of us don't plan enough
  - ♦ More planning helps
  - ♦ Reduces optimism (50% underestimate) problem
- May help reduce overall critical path duration
- Handoffs are clearly defined

### ***Replanning May Help You Shift the Critical Path***

---

15

- Shifting is an option
  - ◆ Reorganize the tasks
  - ◆ Implement in a different order
- If shifting is not an option
  - ◆ Reduce features
  - ◆ Slip schedule

### ***Project 1: Iterative Planning Without Shifting Tasks***

---

16

- Senior management fixed the ship date
- Management did not understand the feature set
- Original schedule
  - 1/15      Complete requirements (feature freeze)
  - 3/31      Code freeze
  - 4/15      Beta ship
  - 5/15      Product ship

## ***Inadequate Initial Project Management***

---

17

- Insufficiently defined task list
- No realization team slipped a week every two weeks
- No project plan
  - ♦ No agreement on interim deliverables and criteria
- No critical path tracking
- New project manager 3/15

## ***New Project Manager's First Schedule***

---

18

Original Schedule		First Replanned Schedule	
1/15	Feature freeze	4/5	Feature Freeze
3/31	Code freeze	4/12	Code Freeze
4/15	Beta ship	4/18	System test freeze
5/15	Product ship	4/22	Start System test. Start beta test.
		6/14	Product Ship

# Software Project Planning: Iterative Planning and Tracking

19

## Final Schedule

---

Original Schedule	First Replanned Schedule	Last Replanned Schedule
1/15 Feature freeze	4/5 Feature Freeze	4/30 Feature Freeze
3/31 Code freeze	4/12 Code Freeze	5/1 Code Freeze, start system test, Start initial beta test. Inch-pebbles used to reduce wait states
4/15 Beta ship	4/18 System test freeze	7/11 Final code and doc freeze
5/15 Product ship	4/22 Start System test. Start beta test.	7/12 Product Ship
	6/14 Product Ship	

20

## Project 1 Results

---

- Did not make hoped-for date
  - ♦ Started iteration too late
  - ♦ Started inch-pebbles too late
- Did make better results than any of the team's previous release

## **Project 2 Iterative Planning with Shifting Tasks: Initial Schedule**

---

21

09/03 Feature Freeze  
10/07 Code Freeze, Start system test  
11/14 Beta freeze, Start beta test  
12/14 Final code (and doc) freeze  
12/31 Product Ship

## **Project State at 10/7**

---

22

- Incomplete feature freeze
  - ♦ Missing one feature
- Knew about the incompleteness at 9/15
  - ♦ Planned for iterative system test
  - ♦ Started partial system test earlier

# Software Project Planning: Iterative Planning and Tracking

23

## Project 2

---

Original Schedule	Next Schedule
09/03 Feature Freeze	09/03 Feature Freeze
10/07 Code Freeze, Start system test	10/07 Partial code Freeze Start partial system test
11/14 Beta freeze, Start beta test	11/01 Final code freeze
12/14 Final code and doc freeze	Start full system test
12/31 Product Ship	11/14 Beta freeze, Start beta test
	12/14 Final code (and doc) freeze
	12/31 Product Ship

24

## Project 2 Results

---

- Made the final date
- Used all the people at all times
- Stayed focused on the endpoint
  - ♦ Even though people were concerned they might not make it

### **Exercise**

---

- Break into two primary groups, include observers
  - ♦ Plan-all-the-work-upfront group
  - ♦ Plan-to-replan group
  - ♦ Observers
- You have a project to create, I will pass out instructions
- Do the project
- Debrief

### **Summary: Iterate When**

---

- You don't completely know what you have to do
  - ♦ Product uncertainty
- You don't know if you have enough time to do it
  - ♦ High schedule risk

## ***Iterative Planning and Tracking***

---

- Project team urgency
- Plan and adapt critical path
  - ♦ As obtain knowledge
  - ♦ Studiously and continuously
- Planned replanning effort
- Use of the project critical path
- Brings rationality into a non-rational project