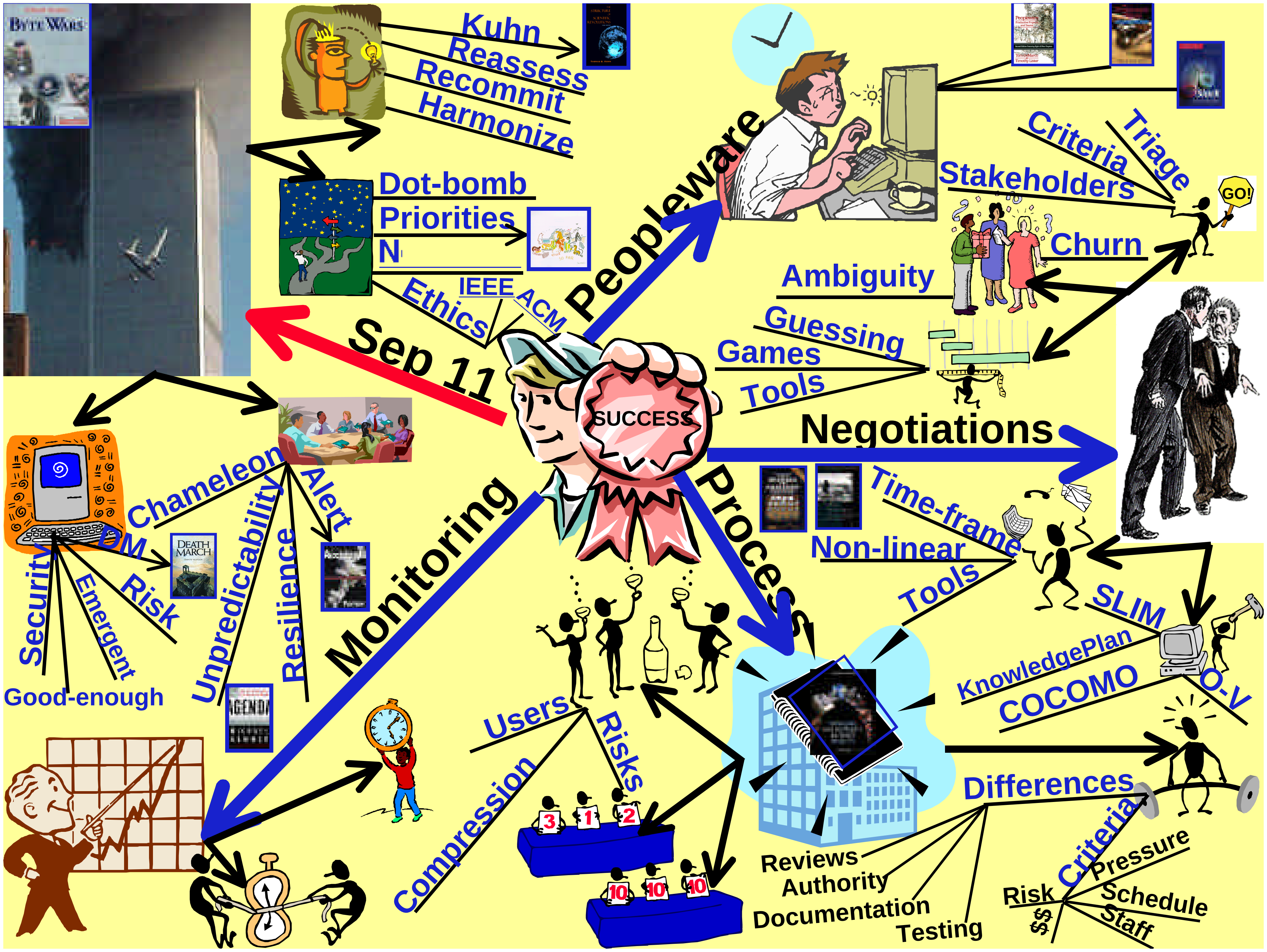
Preparing Software Engineers for the "real world"

Ed Yourdon

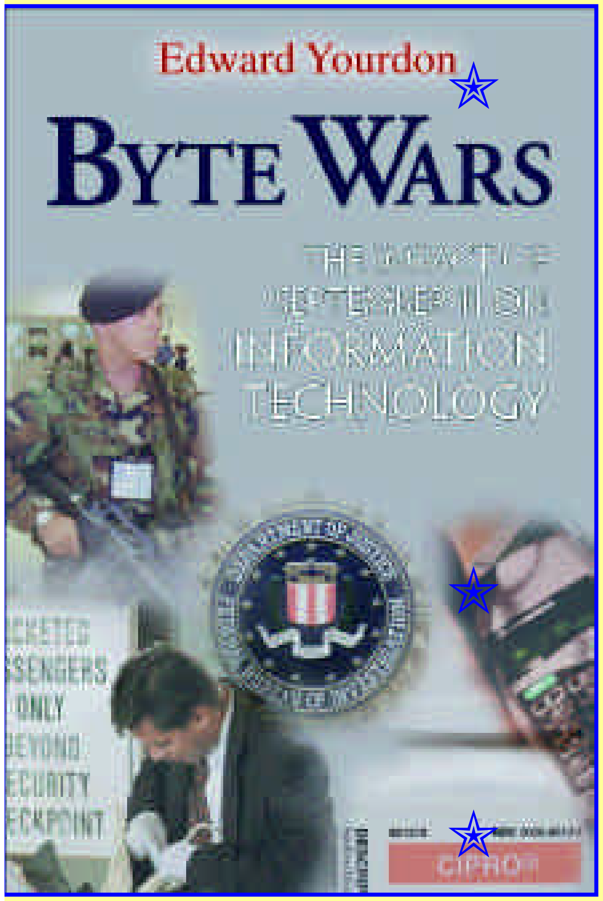
ed@yourdon.com   http://www.yourdon.com

ACM CSEE conference

Cincinnati, Feb 26, 2002


Byte Wars
Kuhn
Reassess
Recommit
Harmonize
Dot-bomb
Priorities
N
IEEE
Ethics
ACM
Sep 11
Peopleware
Criteria
Triage
Stakeholders
GO!
Churn
Ambiguity
Guessing
Games
Tools
Negotiations
SUCCESS
Chameleon
Alert
DM
Death March
Risk
Security
Emergent
Good-enough
Unpredictability
Resilience
Agenda
Monitoring
Process
Time-frame
Non-linear
Tools
SLIM
KnowledgePlan
COCOMO
O-V
Users
Risks
Compression
3
1
2
10
10
10
Differences
Reviews
Authority
Documentation
Testing
Risk
$$
Criteria
Pressure
Schedule
Staff

1. Paradigm shift



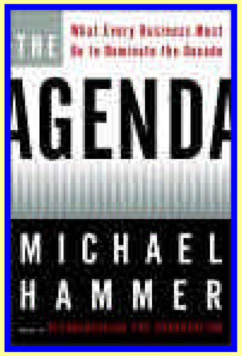
- ☆ *September 11th was a paradigm shift.* See *Byte Wars* for a more complete discussion of why I think this is the case
- ☆ See Thomas Kuhn's *The Structure of Scientific Revolutions* to understand what "paradigm shift" really means.
  - ✓ not a repeal of the law of gravity, or other scientific laws
  - ✓ But what about the lean inventory approach?
  - ✓ What about globalization?
  - ✓ What about replacing server-based systems with P2P systems like Groove? See "Uncle Sam Wants Napster!", in Nov 8, 2001 issue of *The Washington Post*
- ☆ *Reexamine your assumptions, values, priorities* — some assumptions need to be thrown out, some need to be re-assessed in the light of September 11th.
- ☆ *Re-commit to the things that really matter* — sometimes we need a wake-up call.
- ☆ *Look at personal, professional, corporate consequences of September 11* . They should be compatible; if not, do something about it.

2. PERSONAL CONSEQUENCES

- **For most of us, the go-go, get-rich-quick, dot-com days of the late 90s are not only gone, but *permanently* gone.**
- **We need to ask ourselves: *what really matters*?**
  - ✓ **Much of what goes on in corporate IT departments seems utterly irrelevant and petty in the post-9-11 world.**
  - ✓ **Ask your children what they think (for inspiration listen to *Teach Your Children*, from Crosby, Stills, and Nash)**
- **Review the ethics statements of ACM and IEEE**
- **Notice how we all used our own "networks" to communicate in the aftermath of Sep 11th**
  - ✓ **Compare this to the communication that took place after JFK assassination, or after Pearl Harbor attack, or Gettysburg battle**
  - ✓ **Recommendation: focus on bottom-up, grass-roots, *emergent* networks**
  - ✓ **Beware efforts to "control" future crises through top-down, hierarchical, communication mechanisms.**

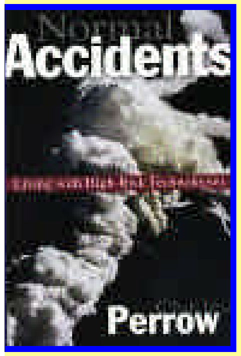
3. CORPORATE CONSEQUENCES

- **See Chapter 12 of Michael Hammer’s new book, *The Agenda: what every business must do to dominate the* *The Agenda*, for a good discussion of this.**
- **Prepare for a world you cannot predict:**
  - ✓ **In 5-year strategic plans developed ~1990-1995, how many would have predicted the Asian financial crisis, Internet/Web, ERP, Euro, supply-chain integration, consequences of deregulation (CA energy crisis)**
  - ✓ **How many would have predicted Sep 11th, and its consequences?**
  - ✓ **Bottom line: change is now too fast, too chaotic, too disruptive, and sometimes too malevolent for us to be able to “plan” for**
- **What this suggests**
  - ✓ ***Change-spotting:* creating an “early warning system”**
  - ✓ **Become adept at *rapid* organizational change**
  - ✓ **Create an organizational infrastructure that supports early-warning and rapid change**
    - ∗ **some of this involves technology**
    - ∗ **but much of it involves organizational culture**



3.1 Change-spotting

- ☆ There *are* "early warning" indicators of disruptive change



  - See *Normal Accidents: Living with High-Risk Technologies*, by Charles Perrow
  - Watch for "near-misses" and avoid common temptation to say, "Whew!"
  - Use metaphors to help categorize "categories" of change — e.g., the "weather" metaphor used by the *Naval War College* during its planning for Y2K.
- ☆ Recognize that lower-level, front-line employees are usually the first to see hints and clues of critical change
  - ✓ Michael Hammer: "The powerless know more than the powerful in virtually all organizations. During periods of intense change, this paradox can be fatal."
  - ✓ Michael Hammer: "…anyone looking for signs of change is almost certainly guilty of not keeping his/her mind clamped on the formal job"
- ☆ One solution: develop a formal *business process* for detecting and reporting change, which incorporates:
  - ✓ deep insight into customers
  - ✓ analyzing *potential* as well as existing competitors
  - ✓ looking for the seeds of the future, by extrapolating the present

4. IT CONSEQUENCES

- ☆ **Risk management has a new level of respectability**
- ☆ **Security now has a greater degree of urgency**
  - ✓ **Prepare for cyber-warfare**
    - ∗ **50% of corporate web servers have been attacked this year, and 90% of companies have experienced worms/viruses; see "Web Attacks Have Doubled, Survey Says" (*PC World,* Oct 10, 2001)**
    - ∗ **longer range: massive DOS zombie-army attacks, facilitated by IP-spoofing capabilities of new Microsoft XP — see description of May 2001 DOS attack on Gibson Research web site**
  - ✓ **See adminspotting for a reminder that cyber-attacks can be caused by disgruntled insiders, as well as outside hackers and terrorists.**
  - ✓ **Develop contingency plans for extended outages of the Internet**
- ☆ **Death-march projects will continue, for obvious reasons…**
- ☆ **Because the dot-com bubble has burst, the era of "glorious anarchy" has been replaced with "extreme programming" and "agile" methods**
- ☆ **And quality may be defined more in terms of "triage," "survival," and "good enough" than "perfection" or "exceeding customer expectations"**

5. PROJECT NEGOTIATIONS

- ☆ Managing project definition at the beginning of the project
- ☆ Using project definition to manage requirements creep
- ☆ Estimating techniques
- ☆ Tools for assisting estimation process
- ☆ Tradeoffs between schedule, budget, staff, quality
- ☆ Tools for rational negotiation
- ☆ What to do when rational communications are impossible

5.1 Managing Project Definition: What does “success” mean?

- ☆ Many projects succeed or fail at the very beginning, before any technical work is done.
- ☆ Fundamental requirement: identifying who has the right to declare “success” — owner, shareholder, etc, etc.
- ☆ Fundamental elements of “success”
  - ✓ finishing on time
  - ✓ staying within budget
  - ✓ delivering the required functionality
  - ✓ providing “good enough” level of quality
  - ✓ getting the next round of VC funding, or launching the IPO
- ☆ The combination of these constraints may prove impossible to achieve — so the *pragmatic* aspect of success often depends on agreement as to which areas can be compromised or satisfied.
- ☆ Biggest risk: lack of realistic triage at *beginning* of project

5.2 Using Project Definition to Manage Requirements Creep

- ☆ **Typical behavior in projects: new requirements are added at the rate of 1% per month**
- ☆ **Requirements "creep" and requirements "churn" are a major element of project management risk.**
- ☆ **But if you don't have a formal document describing the requirements, it's hard to identify creep or churn.**
- ☆ **Assuming that you *do* have such a document, you need to use it to negotiate schedule/budget/staff modifications if the requirements change or increase.**
- ☆ ***Biggest risk of all: an ambiguous spec is usually a sign of unresolved conflict between diverse political camps in the user community. Related risk: techies assume that it's their fault they can't understand ambiguous spec***

5.3 Estimating Techniques

- ☆ **Fundamental truth: it's almost impossible to estimate a project if you don't have metrics from previous projects.**
- ☆ **Consequence: most of what's described as "estimating" is either "guessing" or "negotiating"**
- ☆ **Political reality: estimates are produced by people who have little prior estimating experience, and who have a vested interest in believing their optimistic predictions**
- ☆ **A radical suggestion: create a separate estimating group whose work is judged and rewarded by the *accuracy* of its estimates, not the political acceptability of estimates**
- ☆ **Main technical suggestion: break the project down into small, independent "inch-pebbles" and get *several* estimates**
- ☆ **For complex projects, get a commercial estimating tool**

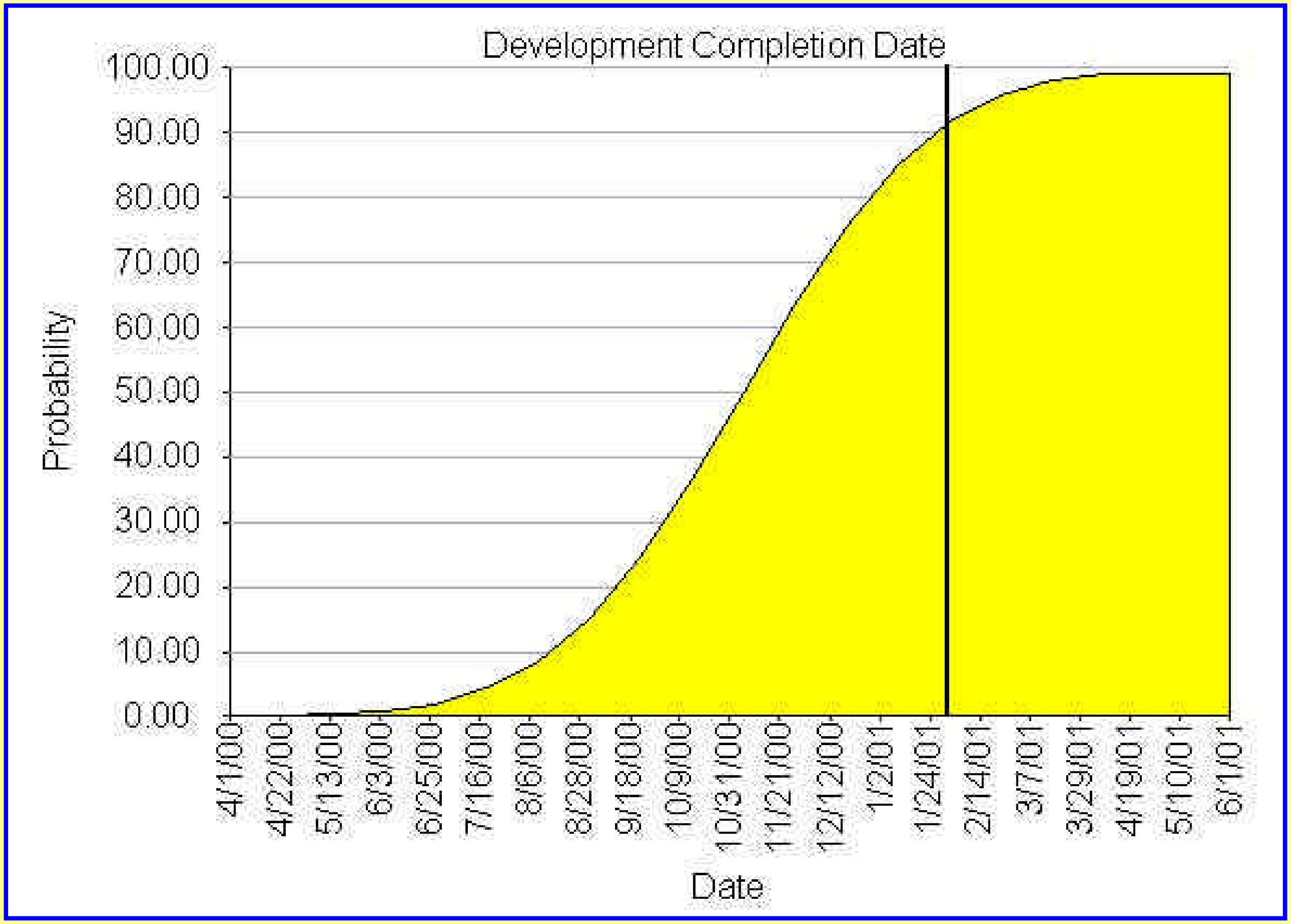
5.4 Tools for Estimating

- ☆ **KnowledgePlan, from Software Productivity Research**
- ☆ **SLIM, from Quantitative Software Management**
- ☆ **ESTIMACS, from Computer Associates**
- ☆ **COCOMO-2, available from several commercial vendors (See CoStar from SoftStar Systems)**
- ☆ **OnYourMarkPro, from Omni-Vista (*caveat emptor*: I'm on the Board of Technical Advisors at this company)**

5.5 Tradeoffs between schedule, budget, functionality, staff, quality

- ☆ Key point: it's not a linear tradeoff — see Fred Brooks, *The Mythical Man-Month* (Addison-Wesley, 1995)
- ☆ Relationship is a non-linear, third-order polynomial relationship — see Larry Putnam and Ware Myers, *Measures for Excellence: Reliable Software on Time, Within Budget* (Prentice-Hall, 1992)
- ☆ Biggest risk: tradeoffs are usually negotiated, under pressure, late in the project schedule — without accepting the non-linear tradeoffs...
- ☆ ...and without accepting the reality that much of the partially-finished work will be lost forever
- ☆ To negotiate tradeoffs rationally, you need to have one of the estimating packages mentioned earlier

Typical trade-off chart from estimating tools



5.6 Project Negotiations

- Beware the temptation to give up... e.g.,
- “We have no idea how long this project will really take, and it doesn’t matter, since they’ve already told us the deadline...
- ...so we’ll just work 7 days a week, 24 hours a day, until we drop from exhaustion. They can whip us and beat us, but we can’t do any more than that...”

5.6, cont'd Negotiating games

- ☆ Doubling and add some...
- ☆ Reverse doubling
- ☆ Guess the Number I'm Thinking of...
- ☆ Double Dummy Spit
- ☆ The X-Plus Game
- ☆ Spanish Inquisition
- ☆ Low Bid
- ☆ Gotcha — throwing good money after bad
- ☆ Chinese Water Torture
- ☆ Smoke and Mirrors/Blinding with Science
  - ✓ thanks to Rob Thomsett, "Double Dummy Spit, and Other Estimating Games," *American Programmer* (now *Cutter IT Journal*), June 1996

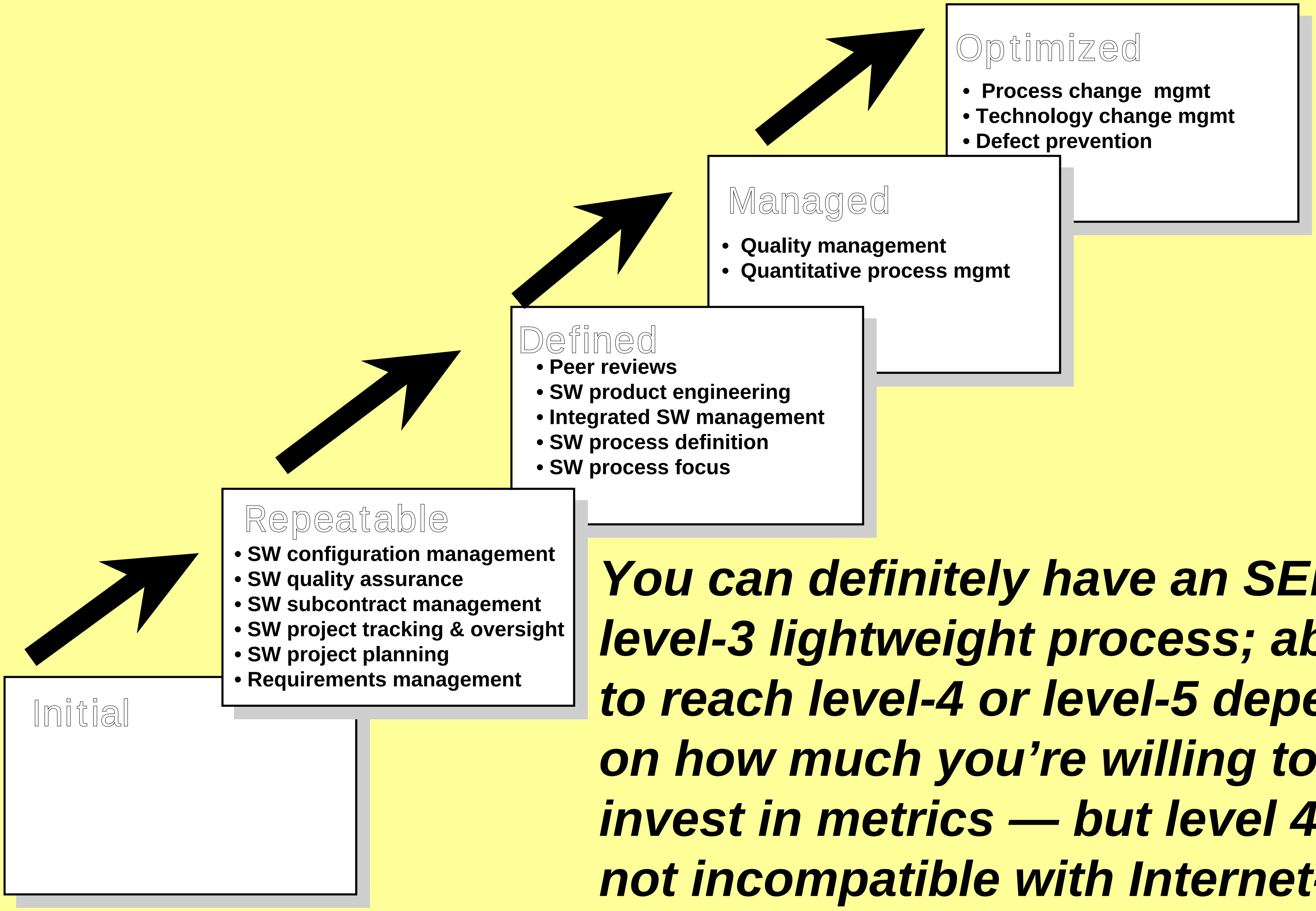
5.6 Negotiating strategies

- ✩ **Don’t get tricked into making an “instant estimate” — ask for time to think about (a week, a day, even an hour)**
- ✩ **State the estimate in terms of confidence levels, or ± ranges, etc.**
- ✩ **Jim McCarthy (formerly of Microsoft, author of *Dynamics of Software Development*): make the customer, or other members of the organization, *share* some of the uncertainty.**
- ✩ **Project manager: “I *don’t know* precisely when we’ll finish — but I’m more likely to be able to figure it out than anyone else in the organization. I promise that as soon as I have a more precise estimate, I’ll tell you right away.”**
- ✩ **Do some reading and research to become better at this area, e.g.:**
  - ✓ ***Bargaining for Advantage: Negotiating Strategies for Reasonable People*, by G. Richard Shell (reissue edition, Penguin Books, June 2000)**
  - ✓ ***Getting Past No: Negotiating Your Way from Confrontation to Cooperation*, by William Ury (Bantam Doubleday Dell, 1993)**

5.7 What to do when rational negotiation breaks down

- ☆ **Quit (the project or the company)**
- ☆ **Appeal to a higher authority**
- ☆ **Go see the movie *Gladiator*, and learn to say, like Russell Crowe, "We who are about to die salute you!"**
- ☆ ***Decide which "rules" you're going to break in order to achieve an "irrational" set of schedule/resource demands that have been imposed upon you.***
- ☆ **Redefine the project as a kamikaze, suicide, etc., and make sure entire project team knows it.**
- ☆ ***Key point:* project leader has to believe in the possibility of achieving project goals**
- ☆ **...and must be able to convince team members without "conning" them**

6. SOFTWARE PROCESSES



You can definitely have an SEI level-3 lightweight process; ability to reach level-4 or level-5 depends on how much you're willing to invest in metrics — but level 4/5 is not incompatible with Internet-time!

6.1. "Lite" vs. "Heavy" Processes

- Formal (heavy) processes are great if you know what you're doing, and if you've done the same thing several times before
- SEI-CMM guru Watts Humphrey: "if a process can't be used in a crisis, it shouldn't be used at all."
- But many high-pressure projects involve doing things that have never been done before — with teams that have never worked together before.
- Conversely, if a team has worked together before, and really "jells", then it doesn't need a formal, heavy process
- Nevertheless, team needs to agree on what processes will be formalized (e.g., change management, source code control, testing(a la XP)), and what processes will be done on a completely ad hoc basis.
- For more details, see
  - ✓ "Extreme Programming," by Jim Highsmith, *e-Business Application Delivery*, Feb 2000.
  - ✓ November 2000 issue of *Cutter IT Journal* on "Light Methodologies"
  - ✓ "Put Your Process on a Diet," by Martin Fowler, *Software Development*, Dec 2000
  - ✓ "Retiring Lifecycle Dinosaurs," by Jim Highsmith, *Software Testing & Quality Engineering*, Jul/Aug 2000
  - ✓ "*The Light Touch*," by Ed Yourdon, Computerworld, Sep 18, 2000

6.2 More on “lite” vs “heavy”

- ☆ **Areas where there are differences**
  - ✓ Degree/volume of documentation
  - ✓ Frequency of reviews and approvals
  - ✓ Degree of decision-making authority — borrowed from “lean manufacturing” approach
- ☆ **Examples of documentation differences: the requirements analysis phase**
  - ✓ Lite approach: one sentence per requirement
  - ✓ Medium approach: one paragraph per requirement
  - ✓ Heavy approach: detailed UML models, data dictionary,etc.
  - ✓ What happens to requirements when development is done?
- ☆ **Criteria for choosing lite vs heavy:**
  - ✓ Degree of pressure for fast delivery
  - ✓ Project cost
  - ✓ Project duration
  - ✓ Staff size
  - ✓ Risk assessment — consequences of failure (safety-critical?)

6.3 The Airlie Council "Principal Best Practices"

- ☆ Formal Risk management
- ☆ Agreement on Interfaces
- ☆ Peer Reviews
- ☆ Metric-Based Scheduling and management
- ☆ Binary Quality Gates at the "Inch-Pebble" Level
- ☆ Program-Wide Visibility of Project Plan and Progress Vs. Plan
- ☆ Defect Tracking Against Quality Targets
- ☆ Configuration management
- ☆ People-aware management Accountability

6.4 Worst Practices

- ☆ **Don't expect schedule compression of ≥10% compared to statistical norm for similar projects**
- ☆ **Don't justify new technology by the need for schedule compression**
- ☆ **Don't force customer-specific implementation solutions on the project**
- ☆ **Don't advocate the use of silver bullet approaches**
- ☆ **Don't miss an opportunity to move items that are under external control off the critical path**
- ☆ **Don't bury all project complexity in software as opposed to hardware**
- ☆ **Don't conduct critical system engineering tasks without sufficient software engineering expertise**
- ☆ **Don't expect to achieve an accurate view of project health from a formal review attended by a large number of unprepared, active reviewers**
- ☆ **Don't expect to recover from a schedule slip of ≥10% without acknowledging a disproportionately *greater* reduction in software functionality to be delivered.**

For more discussion along the same lines, involving the concept of "anti-processes," see *Anti-Patterns and Patterns in Software Configuration Management*, by William J. Brown, Hays W., Iii McCormick, Scott W. Thomas (Wiley, 1999).

6.5 Breathalyzer Test

- ☆ Do you have a current, credible activity network supported by a work breakdown structure (WBS)?
- ☆ Do you have a current, credible schedule and budget?
- ☆ Do you know what software you are responsible for delivering?
- ☆ Can you list the top ten project risks?
- ☆ Do you know your schedule compression percentage?
- ☆ What is the estimated size of your software deliverable? How was it derived?
- ☆ Do you know the percentage of external interfaces that are not under your control?
- ☆ Does your staff have sufficient expertise in the project domain?
- ☆ Have you identified adequate staff to allocate to the scheduled tasks at the scheduled time?

7. MEASURING, MANAGING, AND CONTROLLING PROGRESS

- ☆ General comments and suggestions
- ☆ The importance of the “daily build” approach

7.1 General comments

- **Management approaches based on classical waterfall approach are almost certain to fail in large, complex projects**
- **Need some kind of "time-box" approach based on versions, features, deliverables, etc.**
- **Jim McCarthy: "*Never let a programmer disappear into a dark room*"**
- **If team understands what features/dependencies are required for the next milestone, they will exert their own pressure upon themselves, rather than depending on the manager to beat them up.**
- **If you miss one milestone deadline, it's *crucial* to succeed on the next one.**
- ***Milestone* post-mortems can be incredibly valuable.**

7.2 The “daily build”

- Popularized by Dave Cutler at Microsoft
- Jim McCarthy (former head of Microsoft’s Visual C++ project): “The daily build is the heartbeat of the project — it’s how you know you’re alive”
- Should be automated, and performed overnight — or even more often.
- Various “tricks” can be used to increase its effectiveness
  - ✓ Punishing people who “break” the daily build
  - ✓ Using red-flag/green-flag at office entrance

5. CONCLUSIONS

- September 11th has profound consequences that we don't even fully grasp yet
- We need to help our organizations implement "change-spotting"
- Professional/IT consequences
  - ✓ Risk management has a new level of respectability
  - ✓ Security now has a greater degree of urgency
  - ✓ Death-march projects will continue, for obvious reasons…
  - ✓ Quality may be defined more in terms of "triage," "survival," and "good enough" than "perfection" or "exceeding customer expectations"
  - ✓ The era of "glorious anarchy" has been replaced with "extreme programming" and "agile" methods

Words to live by in the software field

“I wake up each morning determined to change the World ...

...and also to have one hell of a good time.

Sometimes that makes planning the day a little difficult.”

E.B. White

found in the opening of the preface of *Succeeding with Objects*, by Adele Goldberg and Kenneth S. Rubin (Addison-Wesley, 1995)

Preparing Software Engineers for the “real world”

Ed Yourdon

ed@yourdon.com

http://www.yourdon.com