software development processes

Daniel Jackson
the waterfall model, 1970

I believe in this concept, but the implementation described above is risky and invites failure.
Figure 4. Unfortunately, for the process illustrated, the design iterations are never confined to the successive steps.
Royce’s fixes

program design comes first
› do some design between requirements and analysis phases

document the design
› how much? “my own view is quite a lot”

do it twice
› “If the computer program in question is being developed for the first time, arrange matters so that the version finally delivered to the customer for operational deployment is actually the second version insofar as critical design/operations areas are concerned”

plan, control and monitor testing
› with a separate testing team

involve the customer
› “in a formal way, committed... at earlier points before final delivery”
spiral model, 1986

origins of iterative approaches

- **Act**
  - What changes are to be made?
  - Next cycle?

- **Plan**
  - Objective
  - Predictions
  - Plan to carry out the cycle (who, what, where, when)
  - Plan for data collection

- **Study**
  - Analyse data
  - Compare results to predictions
  - Summarise what was learned

- **Do**
  - Carry out the plan
  - Document observations
  - Record data

V model

tests developed in early phases, applied in later phases
extreme programming

Kent Beck, 1999
› take best practices to “extreme” levels
› developed during C3 project with Ron Jeffries

a sample of XP practices
› test first: acceptance and unit tests
› continuous integration
› pair programming
› repeated refactoring

Chrysler’s C3 payroll system
› started in 1996, cancelled in 2000
› implemented in Smalltalk
› running payroll took 1000 hours initially
› Chrysler said they abandoned XP after this
We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.
agile approaches

agile manifesto (2001)
› an articulation of common practices
› a reaction to traditional notions

rejected notions
› upfront design (“BDUF”)
› written documentation (“ceremonial”)
› planning for future modifications

key practices like XP
› continuous integration, test first, refactoring
› features added incrementally (“sprints” and “scrums”)