# Agile Development

- Flexibility, adaptability
  - Change is guaranteed, expected, inevitable
    - Requirements, technology, people, market demands, competition, risks, schedules

# 12 Principles

1. Highest priority is to satisfy the customer
2. Welcome changing requirements, even late
3. Deliver working software frequently
4. Foster relationships with business people, work with them everyday
5. Build projects around motivated people, trust them to get job done
6. Use face-to-face communication, shun email, phone, chat

# 12 Principles

1. Working software is primary measure of progress
2. Agile processes promote sustainable development (maintain a constant pace, indefinitely)
3. Continuous attention to technical excellence and good design enhances agility
4. Simplicity is essential
5. The best architectures, designs, requirements come from self-organizing teams
6. At regular intervals, the team reflects on how to become more efficient , then makes appropriate changes

# 3 Key Assumptions

- “Impossible” to predict which requirements will be fixed and which will change.
2. For many projects, design and construction are interweaved. It is difficult to know how much construction is needed to “prove” a design.
3. Analysis, design, coding, and testing are not easy to plan (they are not as predictable as we would like)

# Human Characteristics

- Competence – innate talent, specific skills, knowledge of process
- Common focus – create working software in a timely fashion
- Collaboration – assess, analyze, use, communicate information
- Decision making ability – authority to make decisions

# Human Characteristics (2)

- Fuzzy problem solving – change = unknown, be prepared
- Mutual trust and respect – “jelled” : whole team is greater than the sum of its parts
- Self-organization – 1) organizes itself for work 2) organizes itself for the environment 3) organizes its work schedule for on-time delivery

# Extreme Programming (XP)

- Planning – user created stories, ranked by the customer
- Team assigns cost (in weeks)
- Decide on which stories to complete first
- Project velocity - # of stories completed in first deliverable

# XP

- Design – ethereal.
  - Develop CRC cards (more later)
- Coding – before you start coding, create tests
  - Pair programming: two people working together
- Testing – mostly automated because we already have the tests, must test after each modification (unit tests)
  - Integration testing – put multiple pieces together
  - Acceptance testing – user says “yes” or “no”

# ASD

- Adaptive software development
- Speculation – planning
- Collaboration – most work is done, mostly trust
  - Criticize without animosity
  - Assist without resentment
  - Work as hard or harder than teammates
  - Have skills (can be learned)
  - Communicate

# ASD

- Learning – reflection on recent projects, try to make things better
  - Focus groups – take time to set down and talk with customer about process
  - Formal technical reviews – assess technology, try to find better technologies
  - Post-mortem – try to correct flaws, be better next time
- ASD is generic and could be applied anytime

# DSDM

- Dynamic system development model
- Follows the Pareto principle:
  - 80% of the product can be completed in 20% of the time required to complete 100% of the product
- Iterative
- Feasibility – decide (with business people) whether or not its possible
- Business study – detailed analysis of costs
- Functional model iteration – iterate through prototypes to illustrate functionality
- Design and build iteration – revisit prototypes, analyze engineering practices, technologies, etc
- Implementation – may not be complete (80%), changes can occur

# SCRUM

- Very small teams
- Highly adaptable process
- Frequent increments
- Work is partitioned
- Testing and documentation are constant
- “done” whenever you want

# SCRUM Process

- Backlog – lists of requirements, prioritized
- Sprints – rapid work sessions (30 days)
- Scrum Meetings – short (15 minutes) meetings (often daily) to assess progress
- Demos – deliver the software increment

# FDD

- Feature driven design
- Feature – a client-valued function, implemented in 2 weeks or less
- Feature hierarchy:
  - <action> the <result> <by\for\to\of> a(n) <object>
    - Add the product to the shopping cart
    - Store the shipping information for a customer
  - Groups: <action>-ing an <object>
    - Making a product sale

# Agile Modeling (AM)

- Model with a purpose (a specific goal)
- Use multiple models – keep options on table, gives input and ideas to design
- Travel light – as progress occurs, only keep the useful models
- Content is more important than representation
- Know your models and the tools required
- Adapt locallly