Agile Methodologies

SWEN 256 – Software Process & Project Management

Process Methodology Myths

- Agile Methods
  - cowboys and hackers
  - undisciplined
  - low quality
- Plan Driven Methods
  - process worship
  - document laden
  - excessive discipline
- It's not that black and white. The process spectrum spans a range of gray.

Important Concepts

Plan-Driven

- Process Improvement
- Process Capability
- Organizational Maturity
- Process Group
- Risk Management
- Verification (building the product right)
- Validation (building the right product)
- System Architecture

Agile

- Embrace Change
- Frequent Delivery
- Simple Design
- Refactoring
- Pair Programming
- Retrospective
- Tacit Knowledge
- Test-Driven Development (TDD)

Both try to minimize risk, but in drastically different ways

Plan-Driven Approach

- Characteristics
  - Systematic engineering approach
  - Completeness of documentation
  - Thorough verification - traceability
  - Traditionally waterfall, but more incremental and evolutionary processes are the norm.

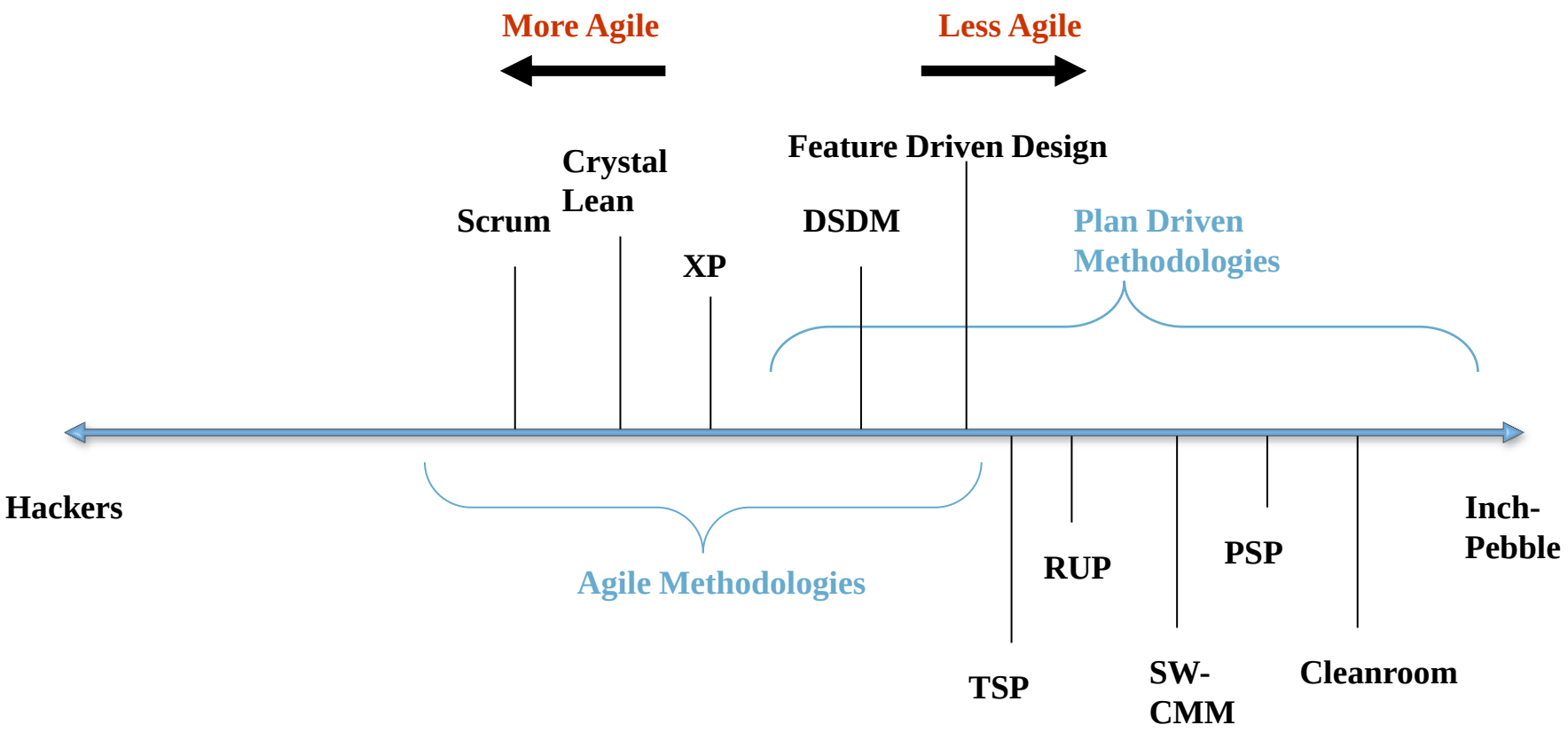
- Examples
  - Cleanroom (mathematically driven)
  - PSP/TSP (Humphrey, SEI)

Agile Approach

- Characteristics
  - Short, iterative cycles
  - Incremental delivery
  - Evolutionary work artifacts (test,design,code)
  - Active customer involvement
  - Dynamic application domains (requirements)

- Examples
  - eXtreme Programming (XP) – (Beck)
  - Crystal family (Cockburn)
  - Scrum (Schwaber)
  - Feature-Driven Development (Coad)

The Process Methodology Spectrum



from “Balancing Agility & Discipline” (Boehm & Turner)

What Is Agile Software Development?

- In the late 1990's several methodologies began to get increasing public attention. All emphasized:
  - close collaboration between the programmer team and business experts
  - face-to-face communication (as more efficient than written documentation)
  - frequent delivery of new deployable business value
  - tight, self-organizing teams
  - ways to craft the code and the team such that the inevitable requirements churn was not a crisis.

- 2001 : Workshop in Snowbird, Utah, Practitioners of these methodologies met to figure out just what it was they had in common. They picked the word "agile" for an umbrella term and crafted the
  - Manifesto for Agile Software Development,

Manifesto for Agile Software Development

Statement of shared development values:

- Individuals and Interactions – over process 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. "

Principles behind the Agile Manifesto

We follow these principles:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Principles behind the Agile Manifesto

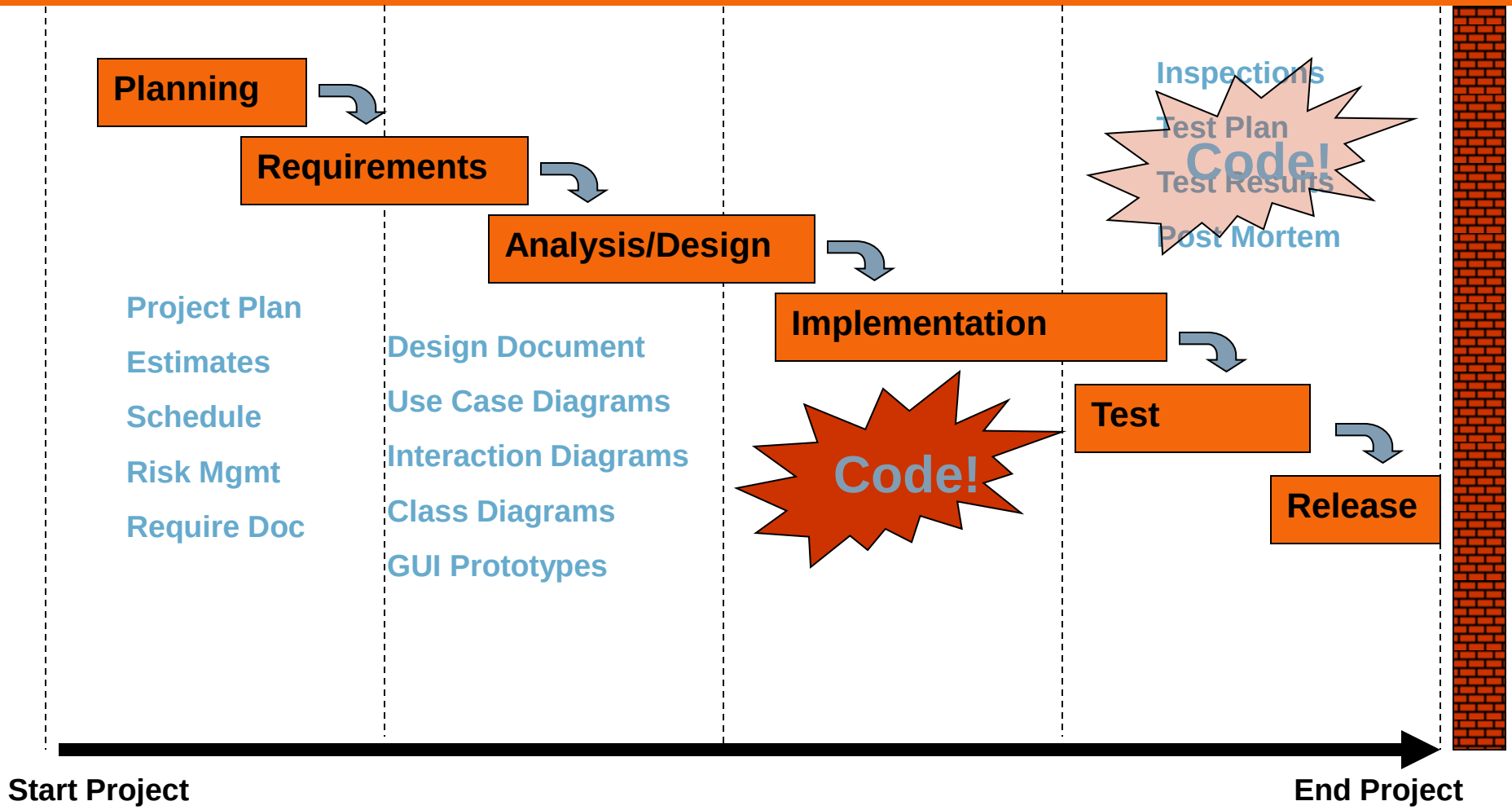
We follow these principles (continued):

- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity--the art of maximizing the amount of work not done--is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Traditional Approach

- Project follows a waterfall process (plan driven)
- Teams produce artifacts at each phase of the life-cycle in a sequential manner.
- Significant upfront design effort
- Implementation delayed until later stages of the project
- Testing deferred until coding complete
- Teams make final presentation to the customer
- Teams participate in postmortem session

Traditional Project Approach



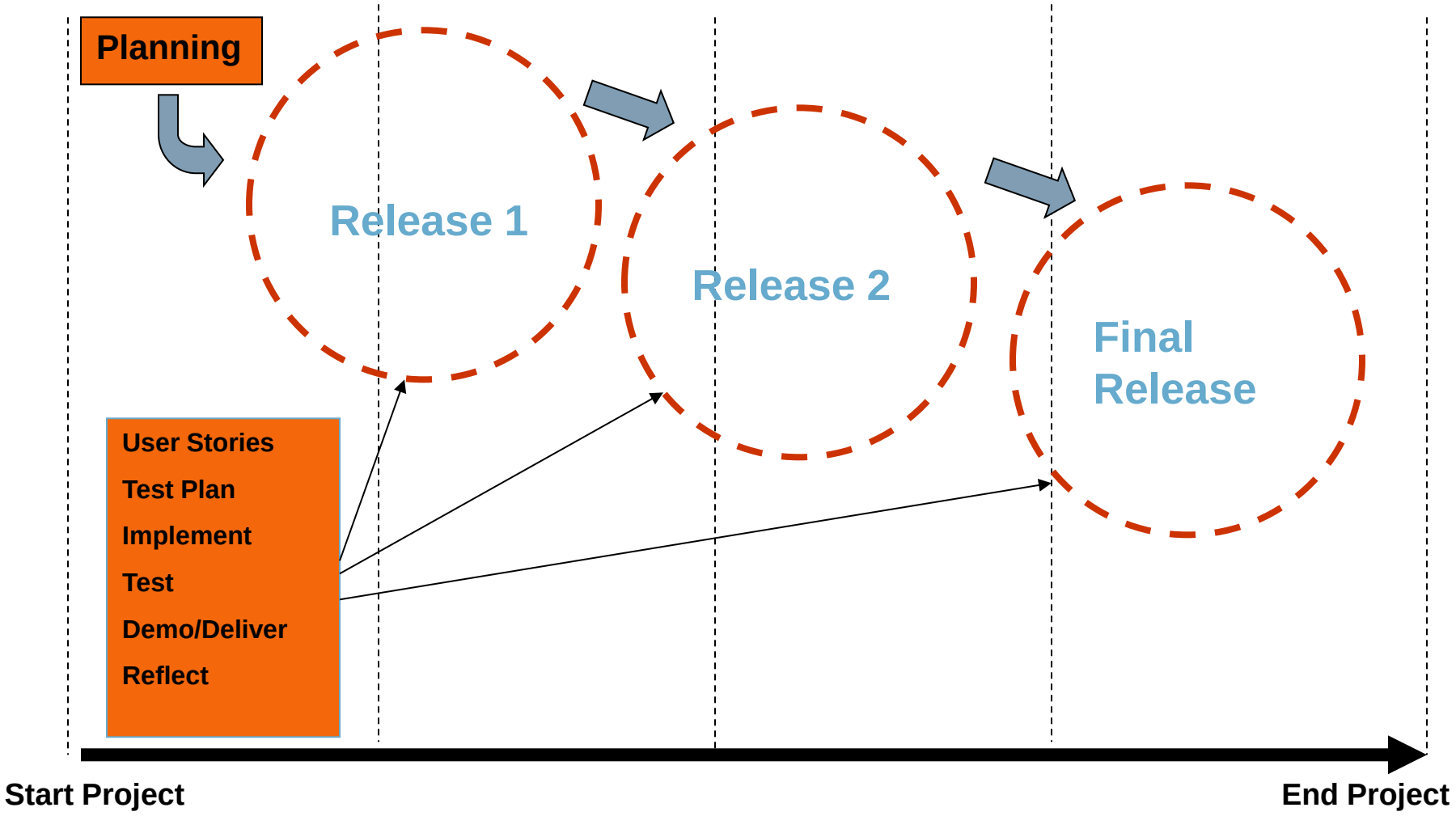
Traditional Challenges

- Lightweight applications/heavyweight process
- Document intensive (perceived)
- Less flexible design
- Big bang approach to coding/integration
- Testing short-shifted
- One-shot delivery opportunity
- Lack of opportunity for process improvement

Agile Characteristics

- Incremental development – several releases
- Planning based on user stories
- Each iteration touches all life-cycle activities
- Testing – unit testing for deliverables
- Testing – acceptance tests for each release
- Flexible Design – evolution vs. big upfront effort
- Reflection after each release cycle
- Several technical and customer focused presentation opportunities

Applying Agility



Key Agile Components

- Team Skills
  - Collaborative Development
  - Reflections (process improvement)
- User Stories
  - Requirements elicitation
  - Planning – scope & composition
- Evolutionary Design
  - Opportunity to make mistakes
- Continuous Integration
  - Code (small booms vs big bang)
- Testing
  - Dispels notion of testing as an end of cycle activity
- Communication
  - Interacting with customer / team members

Agile Themes

- Agile Themes:
  - Lightweight disciplined processes
  - Feature / Customer Focused
  - Small teams
  - Short delivery cycles

Agile Benefits

- User stories drive planning and requirements in a manageable work units
  - Customer perspective
  - Risk management
- Frequent delivery of working software
  - Process reflection opportunities
  - Implementation refactoring
  - Positive feedback to team
- Testing Focus
  - Test early and often
  - Change in attitude towards testing

Resources

- Agile Software Development Portal: agile.csc.ncsu.edu/
- Agile Alliance – www.agilealliance.com
- www.extremeprogramming.org/
- Laurie Williams – North Carolina State: collaboration.csc.ncsu.edu/laurie/index.html

Questions/Discussion