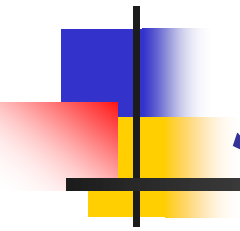
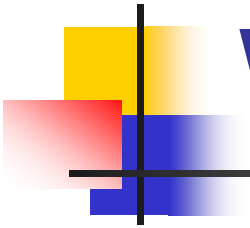


COMP 354: Introduction to Software Engineering

Agility and Process

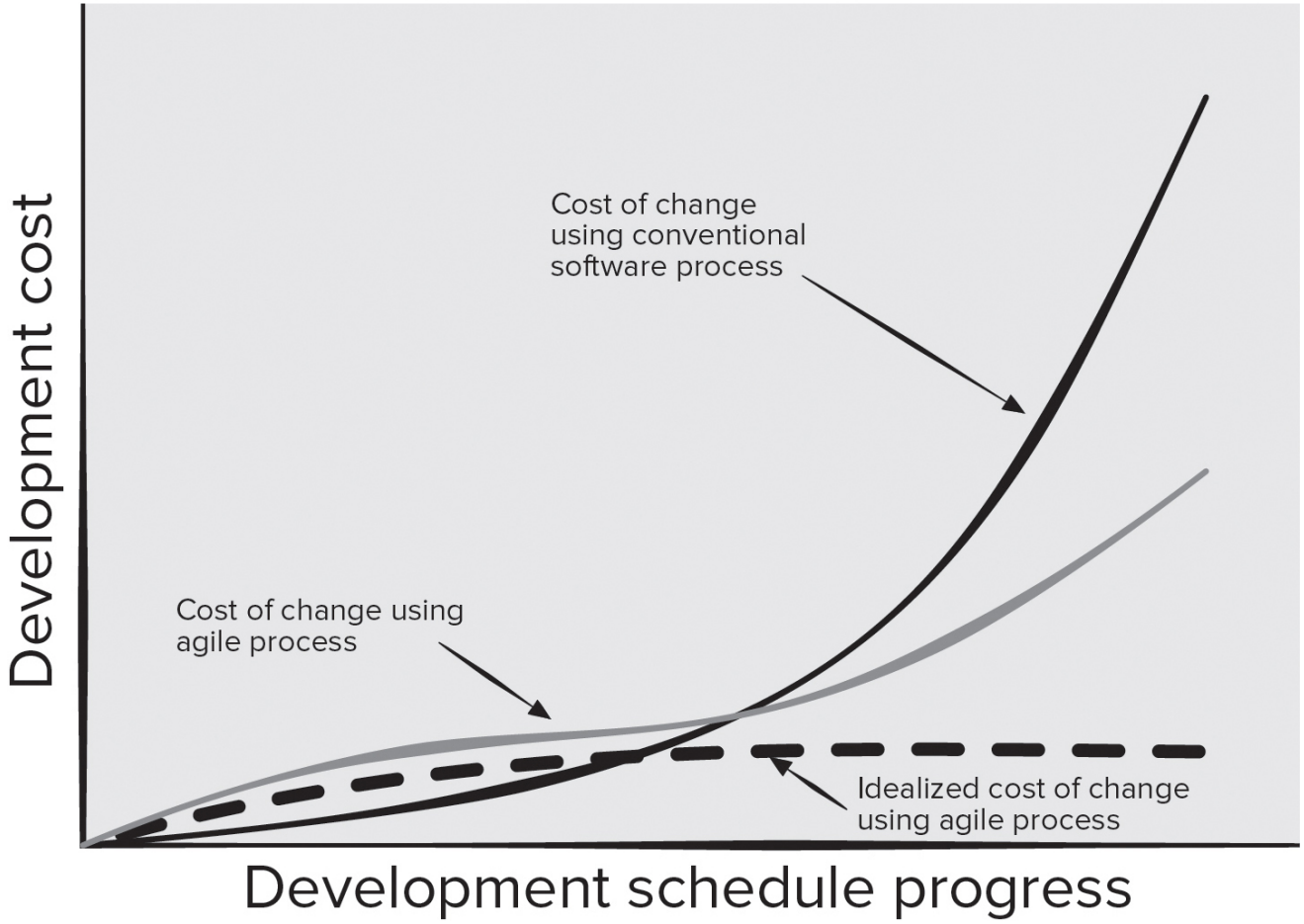
Based on Chapter 3 of the textbook

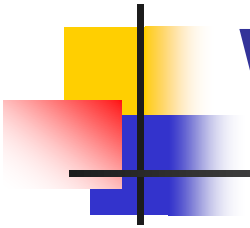


What is Agility?

- Effective (rapid and adaptive) response to change.
- Effective communication among all stakeholders.
- Drawing the customer onto the team.
- Organizing a team so that it is in control of the work performed.
- Rapid, incremental delivery of software.

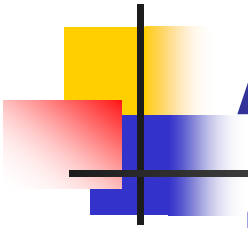
Agility and Cost of Change





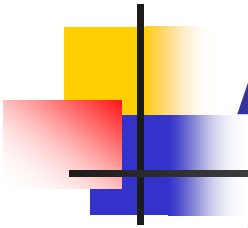
What is an Agile Process?

- Driven by customer descriptions of what is required (scenarios).
- Customer feedback is frequent and acted on.
- Recognizes that plans are short-lived.
- Develops software iteratively with a heavy emphasis on construction activities.
- Delivers multiple 'software increments' as executable prototypes.
- Adapts as project or technical changes occur.



Agility Principles

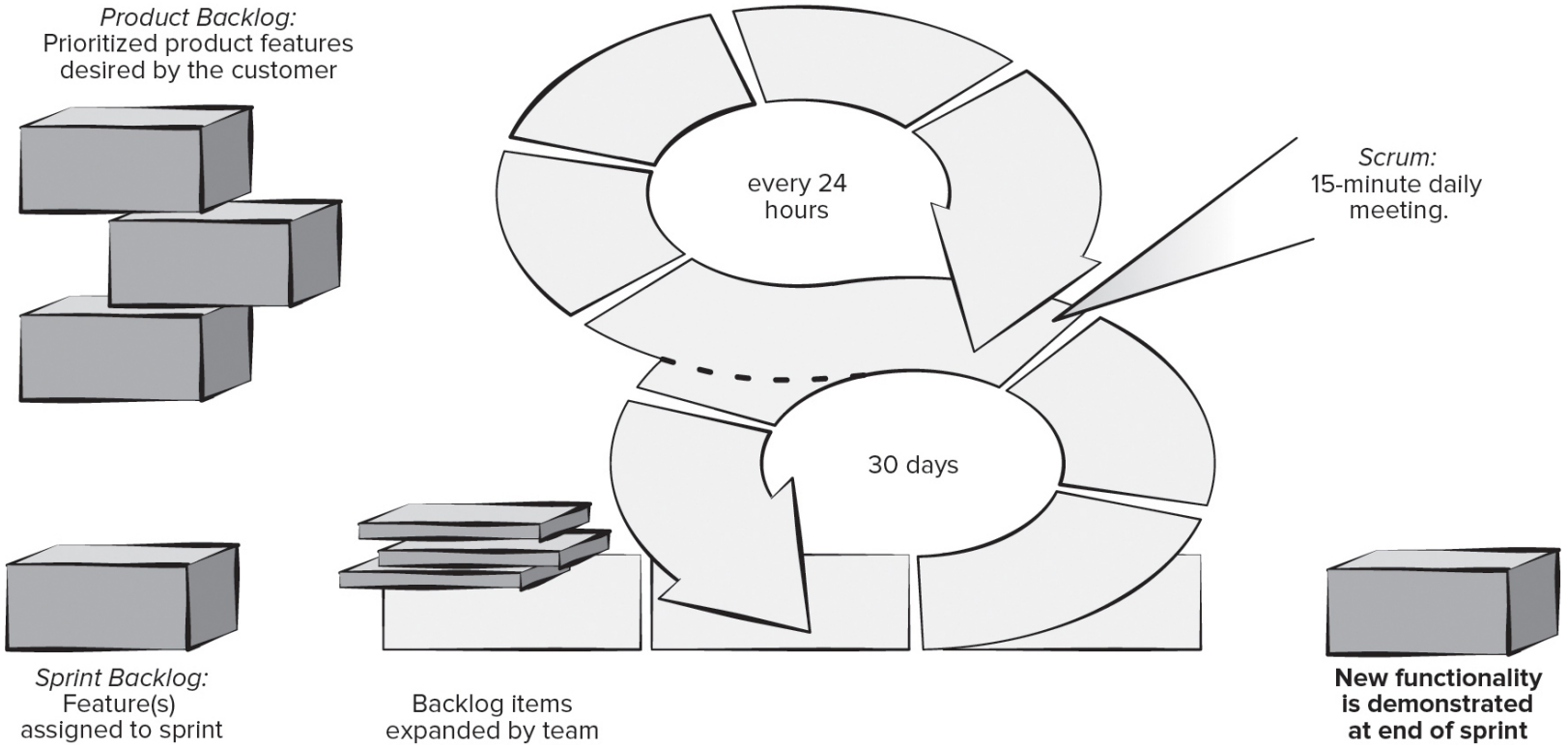
- Customer satisfaction is achieved by providing value through software that is delivered to the customer as rapidly as possible.
- Develop recognizing that requirements will change and welcome changes.
- Deliver software increments frequently (weeks not months) to stakeholders to ensure feedback on their deliveries is meaningful.
- Agile team populated by motivated individuals using face-to-face communication to convey information.
- Team process encourages technical excellence, good design, simplicity, and avoids unnecessary work.



Agility Principles

- Working software that meets customer needs is the primary goal.
- Pace and direction of the team's work must be "sustainable," enabling them to work effectively for long periods of time.
- An agile team is a "self-organizing team"— that can be trusted to develop well-structured architectures that lead to solid designs and customer satisfaction.
- Part of the team culture is to consider its work introspectively with the intent of improving how to become more effective its primary goal (customer satisfaction).

Scrum Framework



Scrum Details

- **Backlog Refinement Meeting** Developers work with stakeholders to create product backlog.
- **Sprint Planning Meeting** Backlog partitioned into "sprints" derived from backlog and next sprint defined.
- **Daily Scrum Meeting** Team members synchronize their activities and plan work day (15 minutes max).
- **Sprint Review** Prototype "demos" are delivered to the stakeholders for approval or rejection.
- **Sprint Retrospective** After sprint is complete, team considers what went well and what needs improvement.

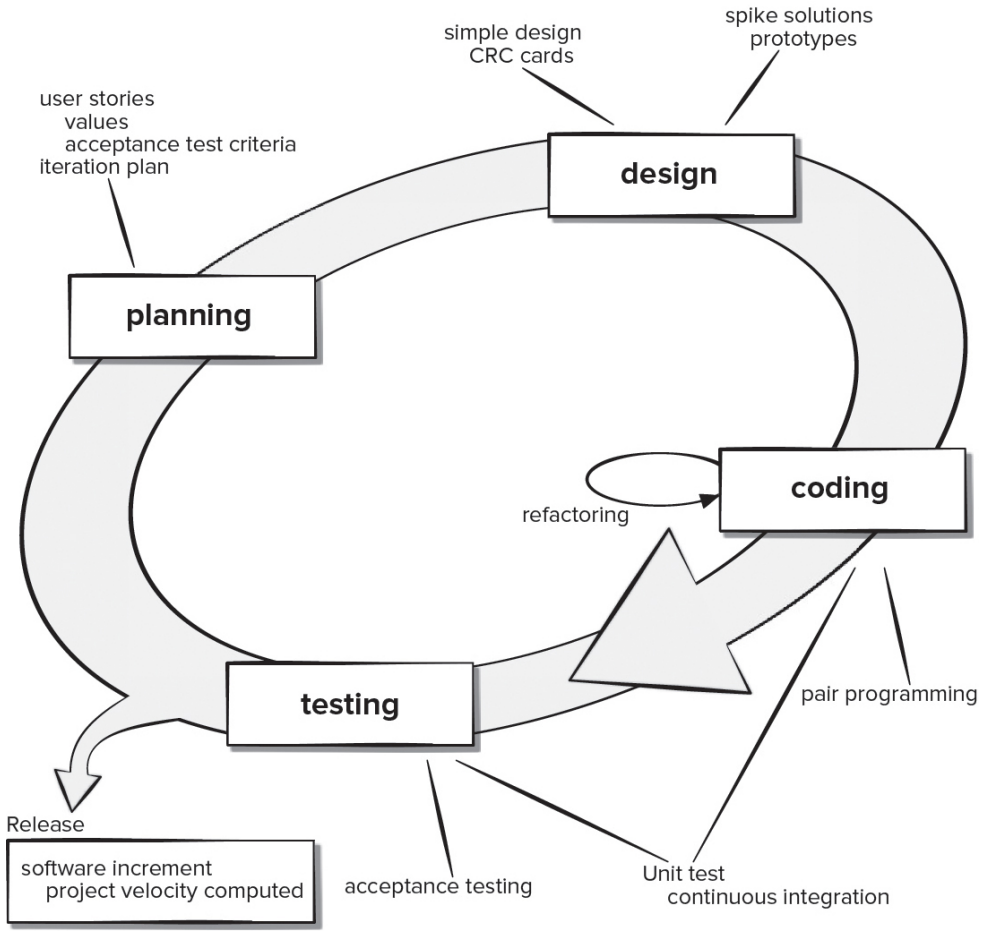
Pros

- Product owner sets priorities.
- Team owns decision making.
- Documentation is lightweight.
- Supports frequent updating.

Cons

- Difficult to control the cost of changes.
- May not be suitable for large teams.
- Requires expert team members.

Extreme Programming (XP) Framework



XP Details

- **XP Planning** – Begins with user stories, team estimates cost, stories grouped into increments, commitment made on delivery date, computer project velocity.
- **XP Design** – Follows KIS principle, encourages use of CRC cards, design prototypes, and refactoring.
- **XP Coding** – construct unit tests before coding, uses pair.
- **XP Testing** – unit tests executed daily, acceptance tests define by customer.

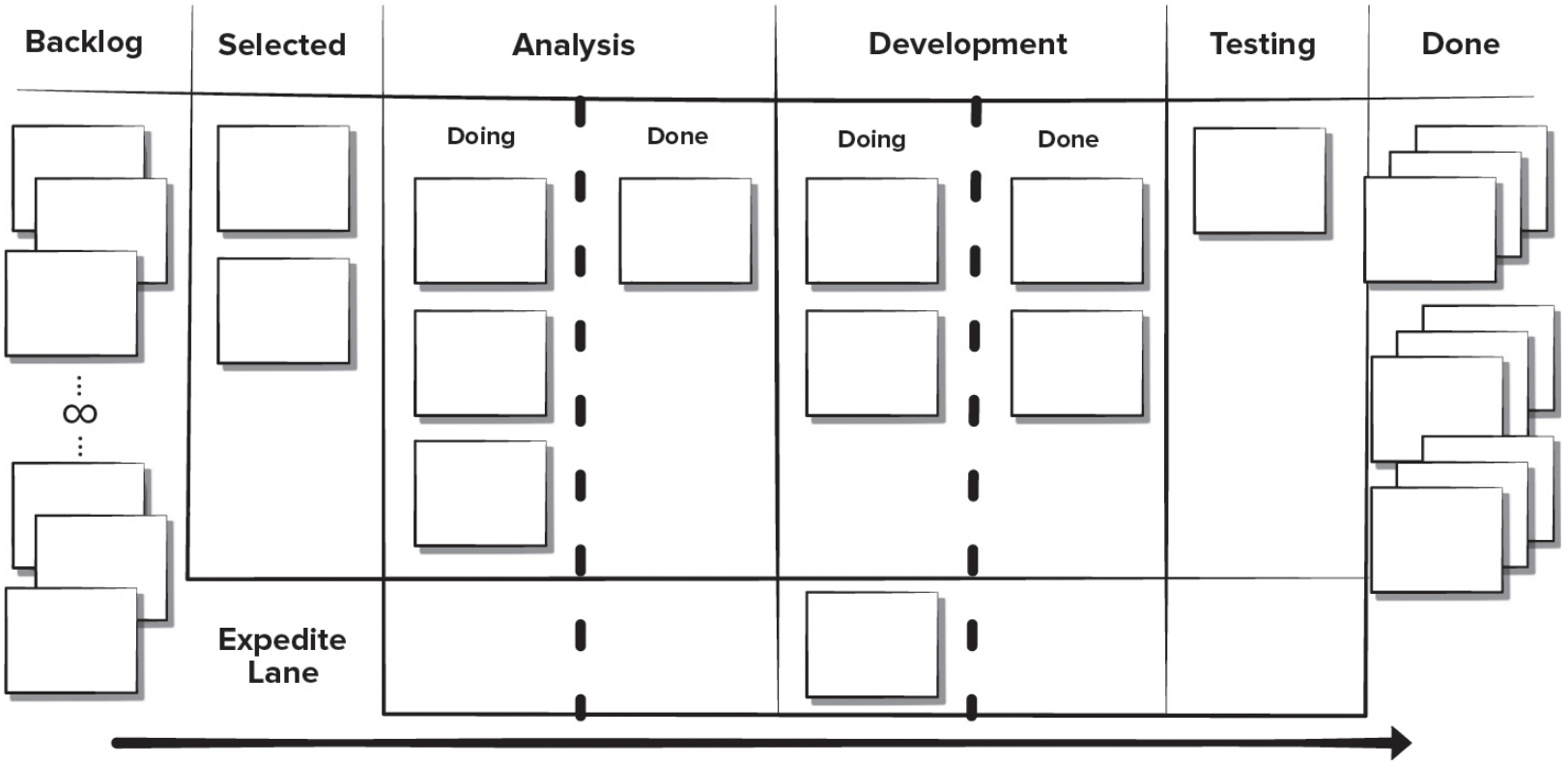
Pros

- Emphasizes customer involvement.
- Establishes rational plans and schedules.
- High developer commitment to the project.
- Reduced likelihood of product rejection.

Cons

- Temptation to "ship" a prototype.
- Requires frequent meetings about increasing costs.
- Allows for excessive changes.
- Depends on highly skilled team members.

Kanban Framework



Kanban Details

- Visualizing workflow using a Kanban board.
- Limiting the amount of *work in progress* at any given time.
- Managing workflow to reduce waste by understanding the current value flow.
- Making process policies explicit and the criteria used to define "done".
- Focusing on continuous improvement by creating feedback loops where changes are introduced.
- Make process changes collaboratively and involve all stakeholders as needed.

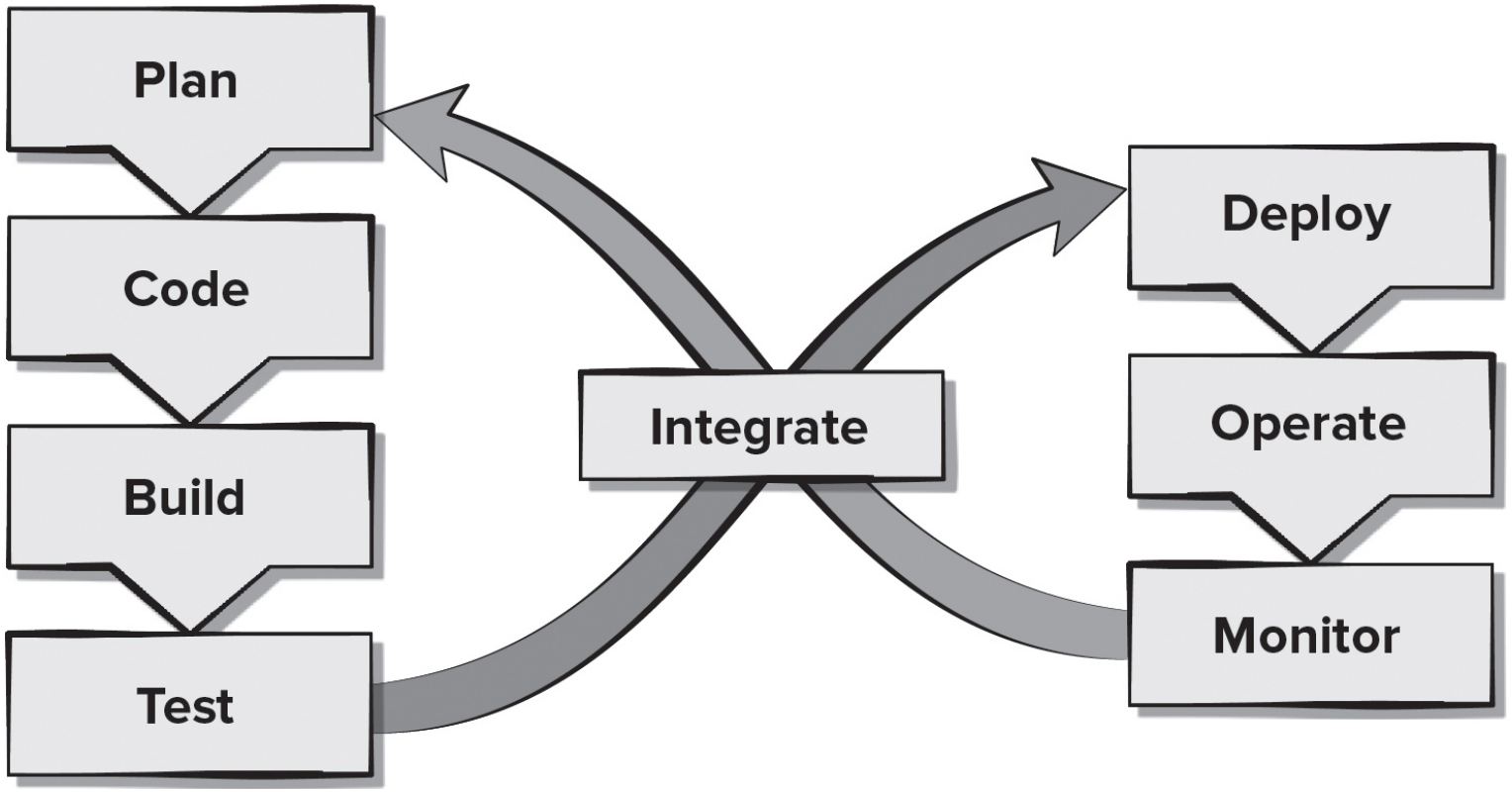
Pros

- Lower budget and time requirements.
- Allows early product delivery.
- Process policies written down.
- Continuous process improvement.

Cons

- Team collaboration skills determine success.
- Poor business analysis can doom the project.
- Flexibility can cause developers to lose focus.
- Developer reluctance to use measurement.

DevOps



DevOps Details

- **Continuous development.** Software delivered in multiple sprints.
- **Continuous testing.** Automated testing tools used prior to integration.
- **Continuous integration.** Code pieces with new functionality added to existing code running code.
- **Continuous deployment.** Integrated code is deployed to the production environment.
- **Continuous monitoring.** Team operations staff members proactively monitor software performance in the production environment.

Pros

- Reduced time to code deployment.
- Team has developers and operations staff.
- Team has end-to-end project ownership.
- Proactive monitoring of deployed product.

Cons

- Pressure to work on both old and new code.
- Heavy reliance on automated tools to be effective.
- Deployment may affect the production environment.
- Requires an expert development team.