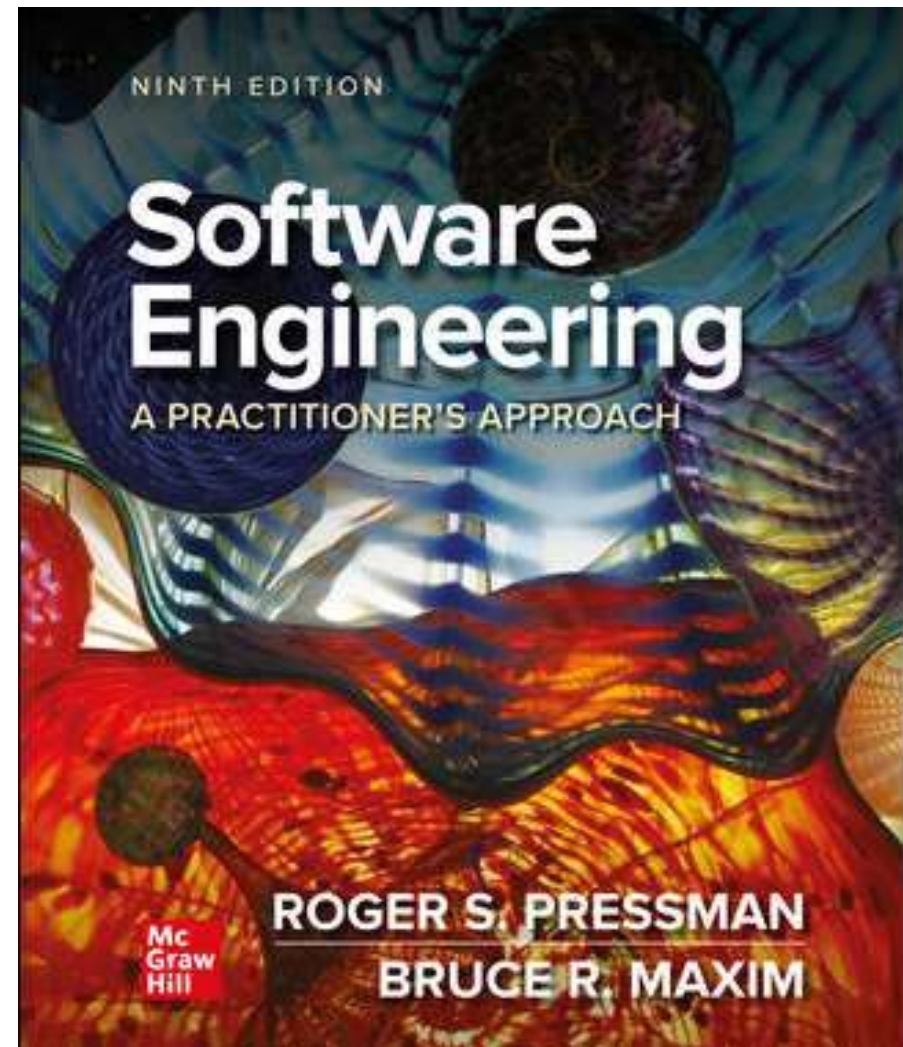


Chapter 6

Principles that Guide Practice

Part Two - Modeling



Principles that Guide Process ¹

- **Principle #1. *Be agile.*** Regards of your process model, let the basic tenets of agile development govern your approach.
- **Principle #2. *Focus on quality at every step.*** The exit condition for every process activity, action, and task should focus on the quality of the work product produced.
- **Principle #3. *Be ready to adapt.*** Dogma has no place in software development. Adapt your approach to constraints imposed by the problem, the people, and the project itself.
- **Principle #4. *Build an effective team.*** Software engineering process and practice are important, but the bottom line is people. Build a self-organizing team.

Principles that Guide Process ²

- **Principle #5. *Establish mechanisms for communication and coordination.*** Projects fail because information falls into the cracks and/or stakeholders fail to coordinate their efforts.
- **Principle #6. *Manage change.*** Approach may formal or informal. You need mechanisms to manage how changes are requested, assessed, approved and implemented.
- **Principle #7. *Assess risk.*** Lots of things can go wrong as software is being developed, establish contingency plans.
- **Principle #8. *Create work products that provide value for others.*** Create only those work products that provide value for other process activities, actions or tasks.

Principles that Guide Practice ¹

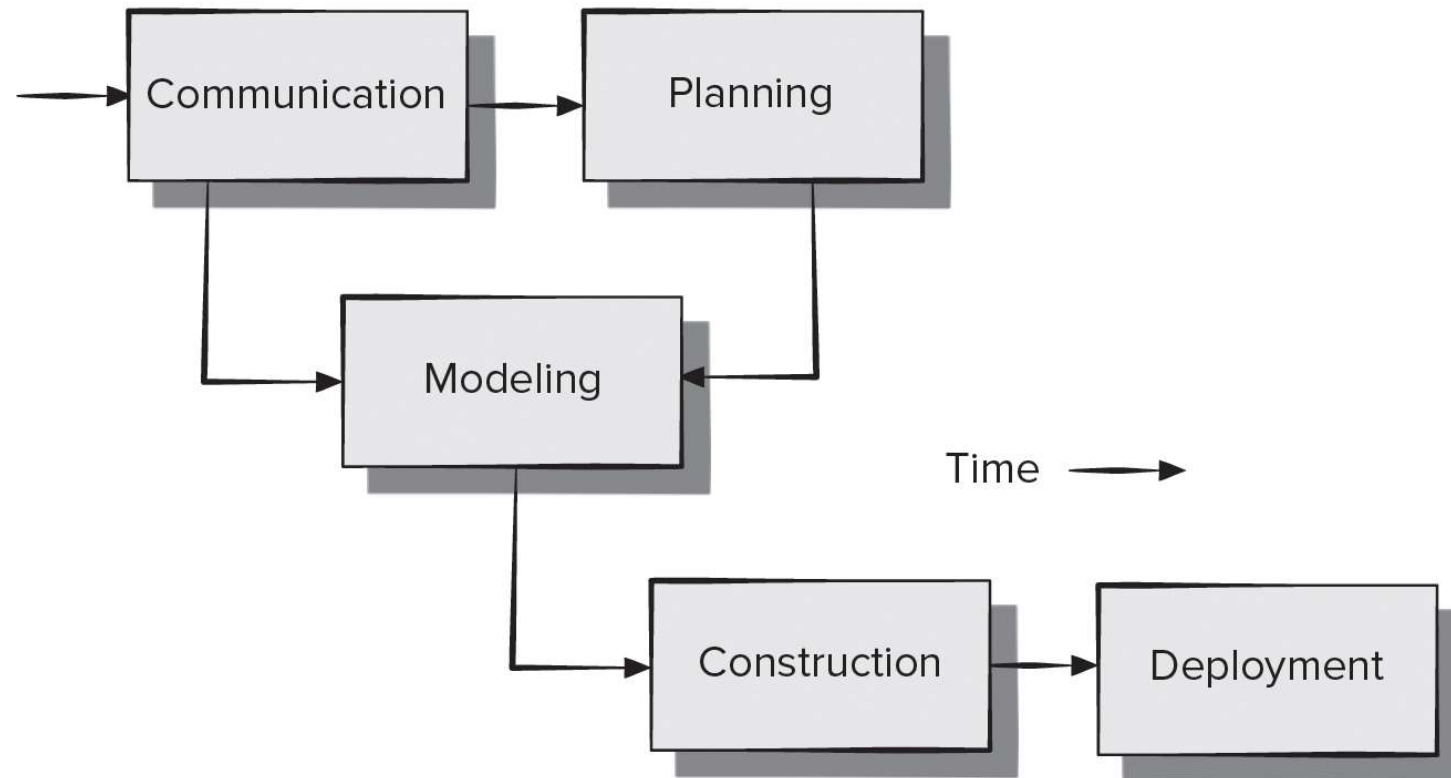
- **Principle #1. *Divide and conquer.*** Analysis and design should always emphasize *separation of concerns* (SoC).
- **Principle #2. *Understand the use of abstraction.*** Abstraction is a simplification of a complex system element used to communicate meaning simply.
- **Principle #3. *Strive for consistency.*** A familiar context makes software easier to use.
- **Principle #4. *Focus on the transfer of information.*** Pay special attention to the analysis, design, construction, and testing of interfaces.

Principles that Guide Practice ²

- **Principle #5. *Build software that exhibits effective modularity.*** Provides a mechanism for realizing the philosophy of Separation of concerns .
- **Principle #6. *Look for patterns.*** The goal of patterns is to create a body of literature to help developers resolve recurring problems encountered in software development.
- **Principle #7. *Use multiple viewpoints.*** Represent the problem and solution from different perspectives.
- **Principle #8. *Some consumes your work products.*** Remember that someone will maintain the software.

Simplified Process Framework

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



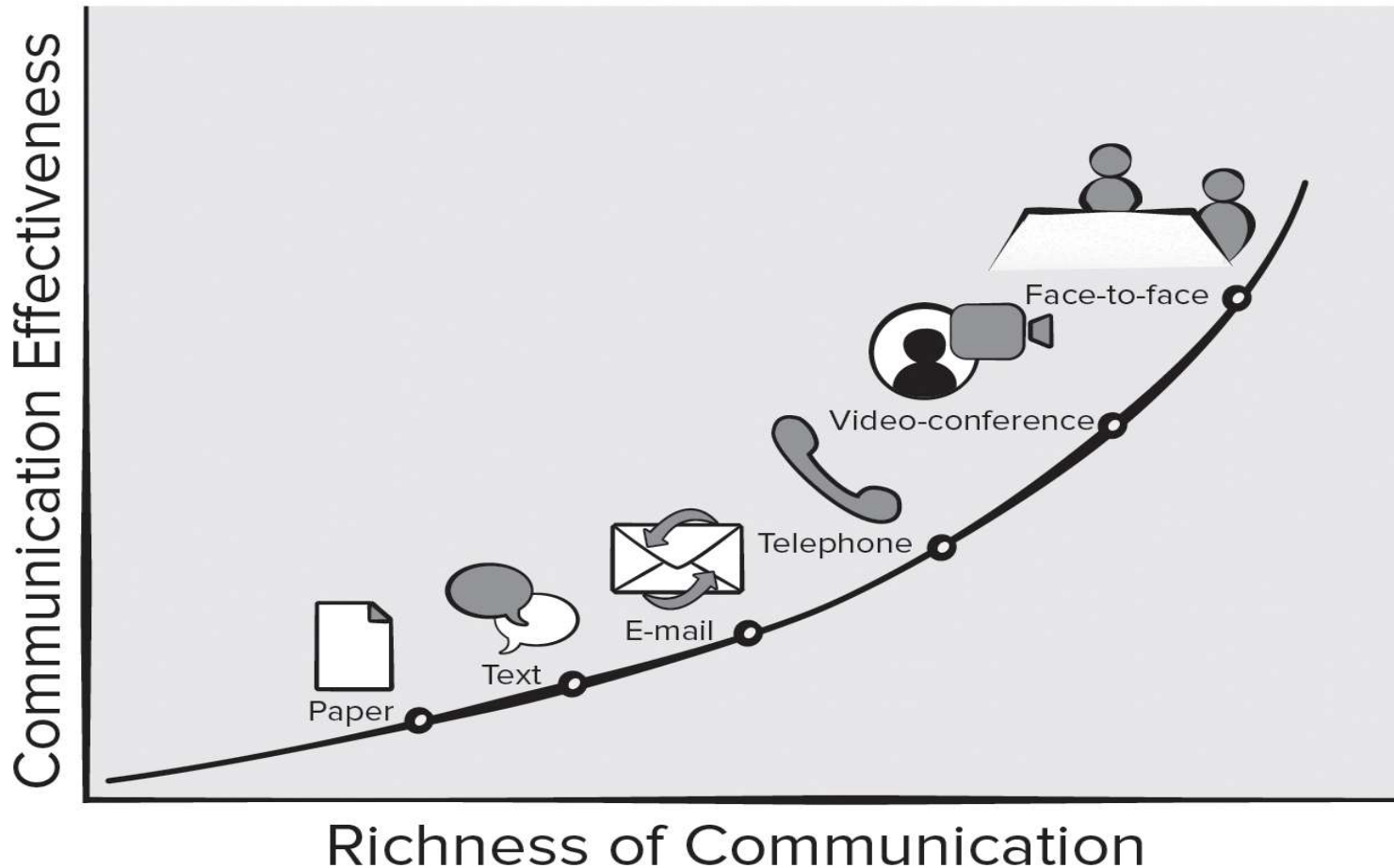
[Access the text alternative for slide images.](#)

Communications Principles ₁

- **Principle #1. *Listen.*** Try to focus on the speaker's words, not formulating your response to those words.
- **Principle # 2. *Prepare before you communicate.*** Understand a problem before meeting with others.
- **Principle # 3. *Someone should facilitate the activity.*** Every communication meeting should have a leader to keep the conversation moving in a productive direction.
- **Principle #4. *Face-to-face communication is best.*** Visual representations of information can be helpful.
- **Principle # 5. *Take notes and document decisions.*** Someone should serve as a “recorder” and write down all important points and decisions.

Communications Mode Effectiveness

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



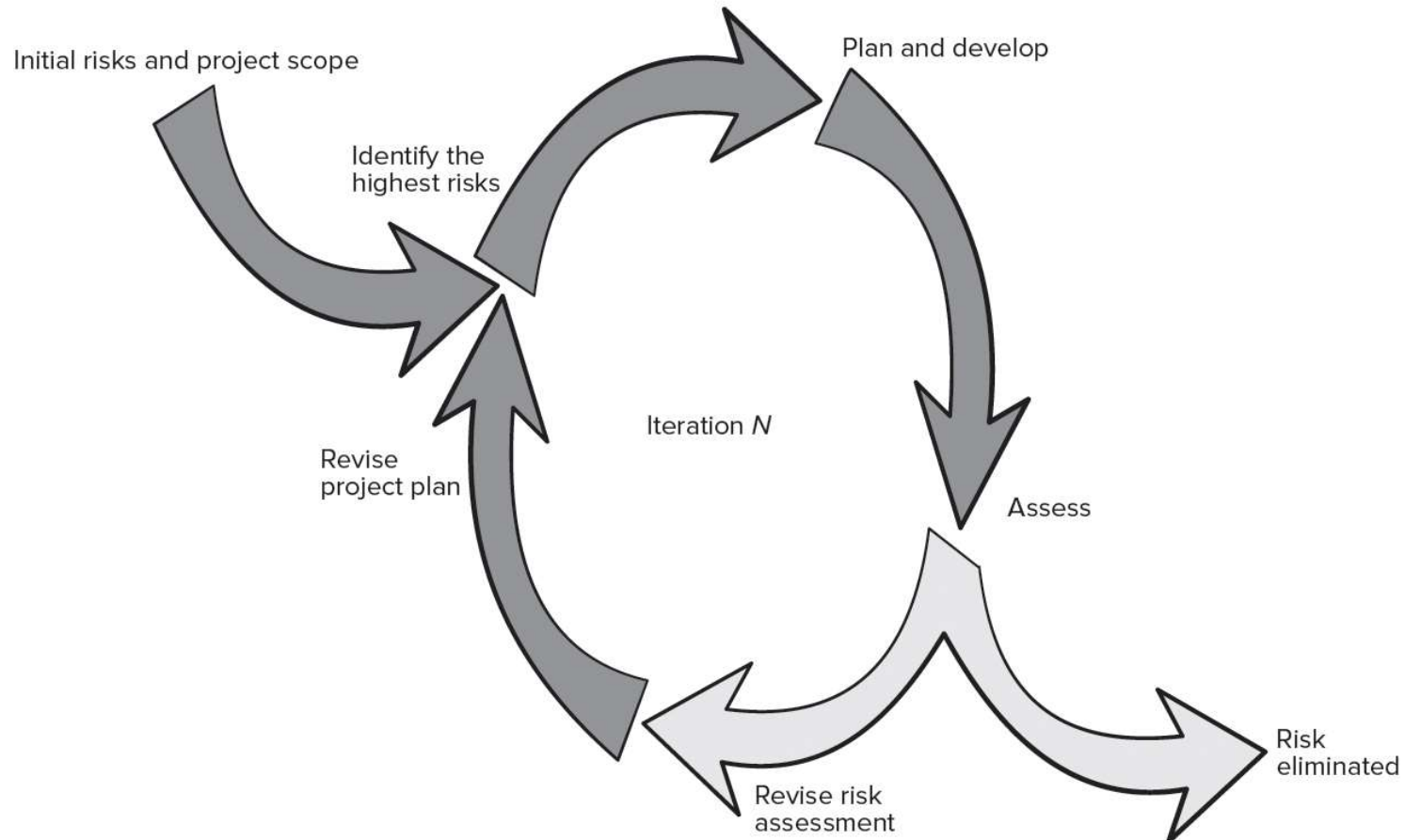
[Access the text alternative for slide images.](#)

Communications Principles ²

- **Principle # 6. *Strive for collaboration.*** Consensus occurs when collective team knowledge is combined.
- **Principle # 7. *Stay focused, modularize your discussion.*** The more people involved in communication the more likely discussion will bounce between topics.
- **Principle # 8. *If something is unclear, draw a picture.***
- **Principle # 9. (a) *Once you agree to something, move on;* (b) *If you can't agree to something, move on;* (c) *If a feature or function is unclear and cannot be clarified at the moment, move on.***
- **Principle # 10. *Negotiation is not a contest or a game. It works best when both parties win.***

Iterative Planning Process

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

Planning Principles ¹

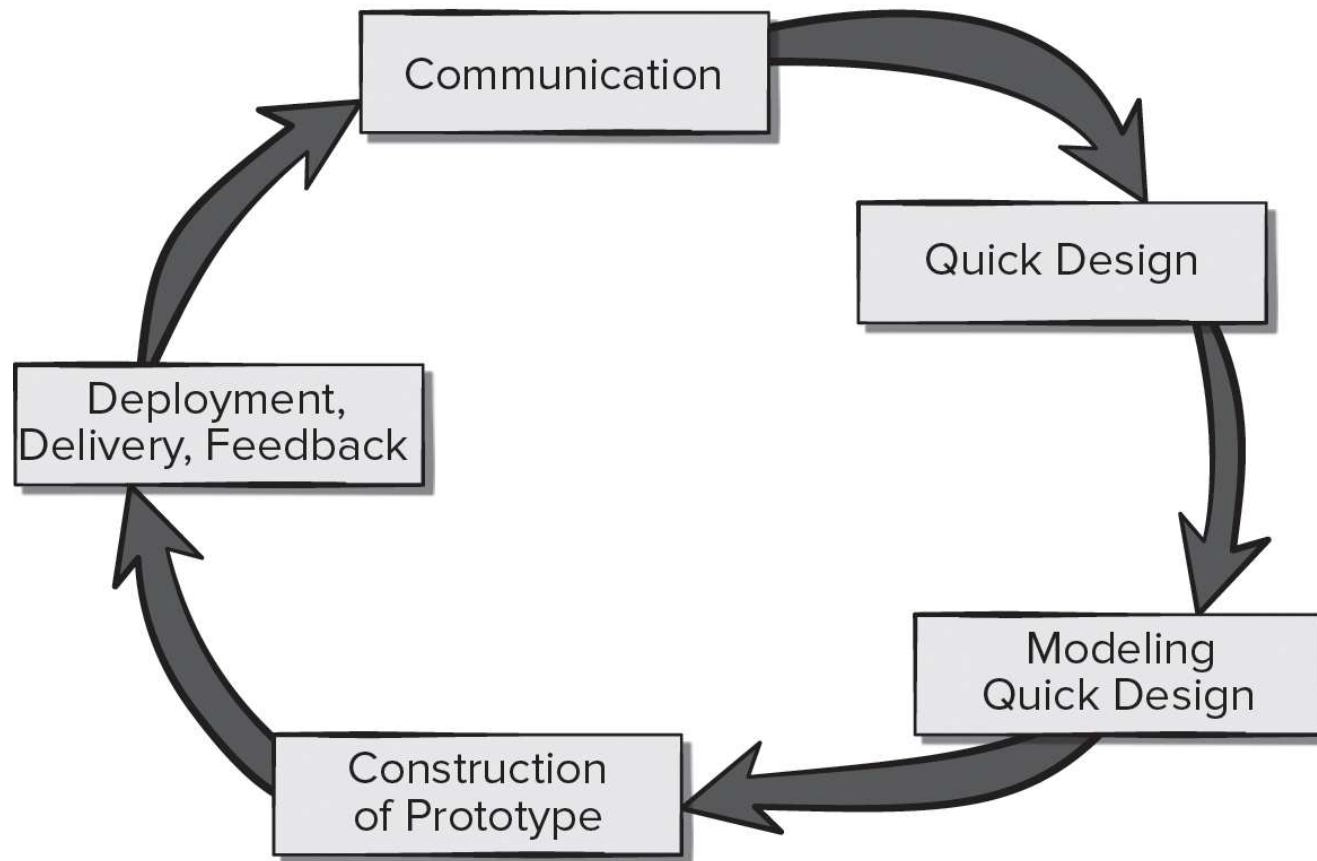
- **Principle #1. *Understand the scope of the project.*** Scope provides the software team with a destination as the roadmap is created.
- **Principle #2. *Involve the customer in the planning activity.*** They define priorities and project constraints.
- **Principle #3. *Recognize that planning is iterative.*** A project plan is likely to change as work begins.
- **Principle #4. *Estimate based on what you know.*** Estimation provides an indication of effort, cost, and task duration, based on team's current understanding of work.
- **Principle #5. *Consider risk as you define the plan.*** Contingency planning is needed for identified high impact and high probability risks.

Planning Principles ²

- **Principle #7. *Adjust granularity as you define the plan.*** *Granularity* refers to the level of detail that is introduced as a project plan is developed.
- **Principle #8. *Define how you intend to ensure quality.*** Your plan should identify how the software team intends to ensure quality.
- **Principle #9. *Describe how you intend to accommodate change.*** Even the best planning can be obviated by uncontrolled change.
- **Principle #10. *Track the plan frequently and make adjustments as required.*** Software projects fall behind schedule one day at a time.

Software Modeling

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

Agile Modeling Principles ¹

- Principle #1. *The primary goal of the software team is to build software not create models.*
- Principle #2. *Travel light – don't create more models than you need.*
- Principle #3. *Strive to produce the simplest model that will describe the problem or the software.*
- Principle #4. *Build models in a way that makes them amenable to change.*
- Principle #5. *Be able to state an explicit purpose for each model that is created.*

Agile Modeling Principles ²

- **Principle #6. Adapt the models you create to the system at hand.**
- **Principle #7. Try to build useful models, forget about building perfect models.**
- **Principle #8. Don't become dogmatic about model syntax. Successful communication is key.**
- **Principle #9. If your instincts tell you a paper model isn't working you may have a reason to be concerned.**
- **Principle #10. Get feedback as soon as you can.**

Construction Principles - Coding ¹

Preparation Principles: Before you write one line of code, be sure you:

- **Principle 1.** *Understand the problem to be solved.*
- **Principle 2.** *Understand basic design principles and concepts.*
- **Principle 3.** *Pick a programming language that meets the needs of the software to be built.*
- **Principle 4.** *Select a programming environment that provides tools that will make your work easier.*
- **Principle 5.** *Create a set of unit tests that will be applied once the component you code is completed.*

Construction Principles - Coding ²

Coding Principles: As you begin writing code, be sure you:

- **Principle 6.** *Constrain your algorithms by following structured programming practice.*
- **Principle 7.** *Consider the use of pair programming.*
- **Principle 8.** *Select data structures that will meet the needs of the design.*
- **Principle 9.** *Understand the software architecture and create interfaces that are consistent with it.*

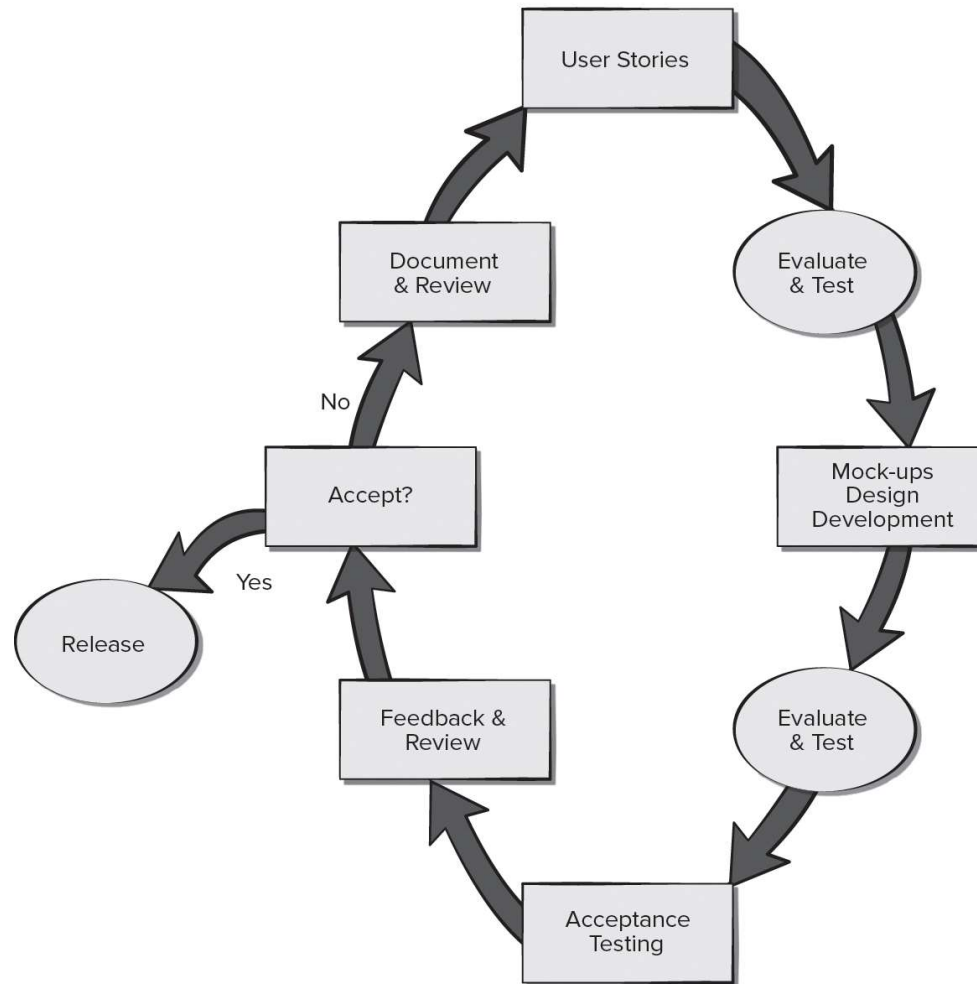
Construction Principles - Coding ³

Validation Principles: After you've completed your first coding pass, be sure you:

- **Principle 10.** *Conduct a code walkthrough when appropriate.*
- **Principle 11.** *Perform unit tests and correct errors you've uncovered.*
- **Principle 12.** *Refactor the code to improve its quality.*

Agile Testing

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

Testing Principles ¹

- **Principle #1.** *All tests should be traceable to customer requirements.*
- **Principle #2.** *Tests should be planned long before testing begins.*
 1. Testing is a process of executing a program with intent of finding an error,
 2. A good test case is one that has a high probability of finding an as-yet-undiscovered error.
 3. A successful test is one that uncovers an as-yet-undiscovered error.
- **Principle #3.** *The Pareto principle applies to software testing.*

Testing Principles ²

- Principle #4. *Testing should begin “in the small” and progress toward testing “in the large.”*
- Principle #5. *Exhaustive testing is not possible.*
- Principle #6. *Testing effort for each system module commensurate to expected fault density.*
- Principle #7. *Static testing can yield high results.*
- Principle #8. *Track defects and look for patterns in defects uncovered by testing.*
- Principle #9. *Include test cases that demonstrate software is behaving correctly.*

Software Deployment Actions

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



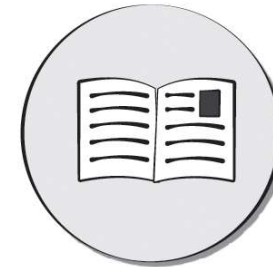
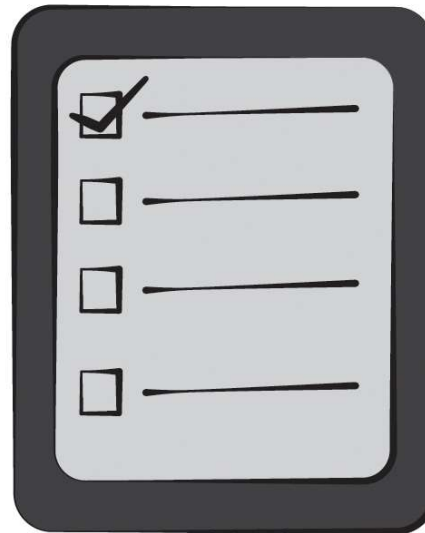
Assemble Deployment Package



Establish Support Regimen



Manage Customer Expectations



Provide Instructional Materials to End Users

[Access the text alternative for slide images.](#)

Deployment Principles ¹

- Principle #1. *Customer expectations for the software must be managed.*
- Principle #2. *A complete delivery package should be assembled and tested.*
- Principle #3. *A support regime must be established before the software is delivered.*
- Principle #4. *Appropriate instructional materials must be provided to end-users.*
- Principle #5. *Buggy software should be fixed first, delivered later.*



Because learning changes everything.®

www.mheducation.com