BIG PROJECTS

Git, Project Management, Software Development Processes

Software Estimation is Hard

| But not impossible | Estimation is an iterative process | Think in terms of time ranges | Combine with confidence measure | Any guess is better than none |

People use it.
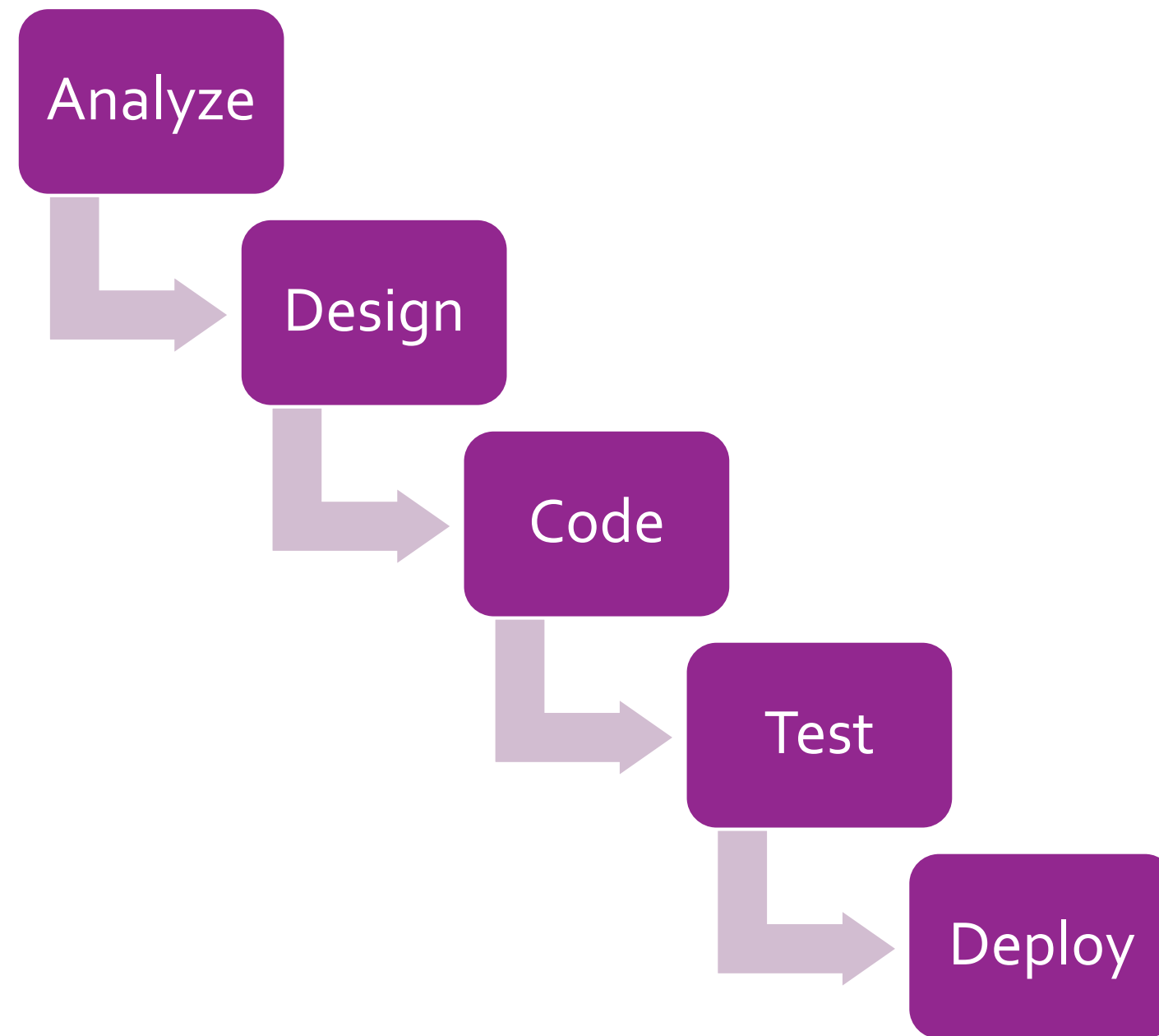
How can I tell if Software is Good?

It is cheap to develop.

HOW CAN I TELL IF CODE IS GOOD?

Good Software will Evolve

- There are always bugs.

- There is always room for improvement.

- There will always be requests for new features and functionality.

Waterfall Process

Analyze → Design → Code → Test → Deploy
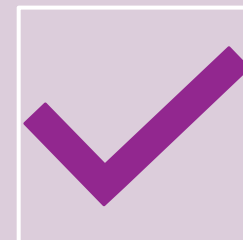
Agile Programming Manifesto

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Extreme Programming (XP)

- An agile software development methodology intended to improve software quality and responsiveness to changing customer requirements.

- Advocates frequent releases in short development cycles, intended to improve productivity and introduce checkpoints at which new customer requirements can be adopted.

What About Scrum?

Scrum Framework © 2020 Scrum.org
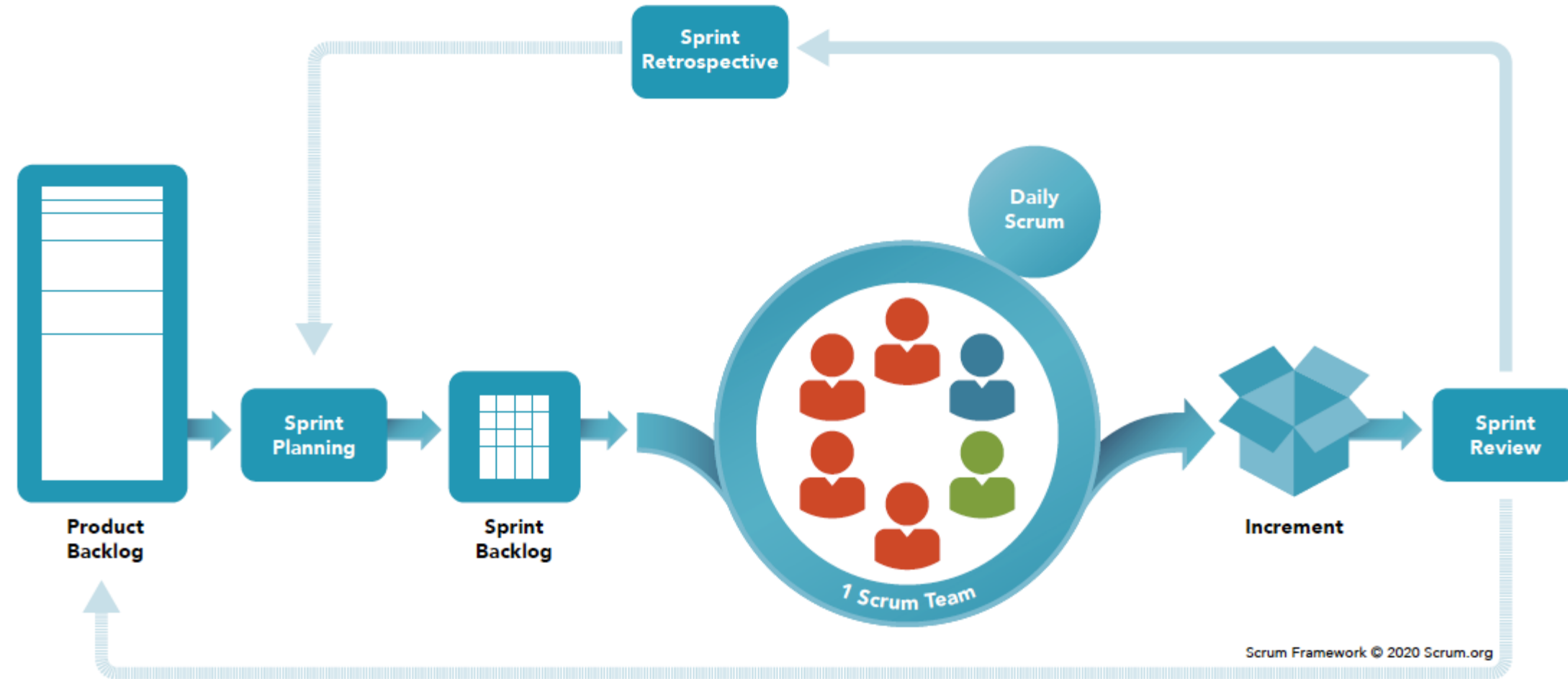
Scrum Teams

- Consist of one Scrum Master, one Product Owner, and Developers. Within a Scrum Team, there are no sub-teams or hierarchies. It is a cohesive unit of professionals focused on one objective at a time, the Product Goal.

- Cross-functional, meaning the members have all the skills necessary to create value each Sprint. They are also self-managing, meaning they internally decide who does what, when, and how.

- Small enough to remain nimble and large enough to complete significant work within a Sprint, typically 10 or fewer people.

Scrum Challenges

- Very process heavy

- Over-emphasizes big items

- Doesn't consider ROI, or customer satisfaction

- Many meetings

- Loses track of big picture

- Assumes skills are transferable within a team.

- Scrum-masters and product owners have too much influence.

Misconception Sharing of Skills

- Assumption is: "scrum engages groups of people who collectively have all the skills and expertise to do the work and share or acquire such skills as needed"

- Most scrum-masters and product-owners are not developers.

- Scrum teams include product designers and quality assurance engineers, and maybe documentation.

- Makes estimation and sharing of work inconsistent.

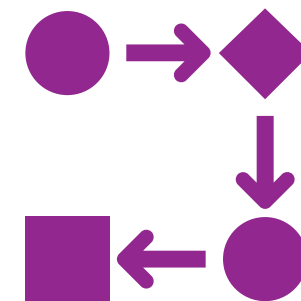- Transfer of skills not always possible

Scrum is a Business

- The scrum alliance is in the business of selling scrum-master certification
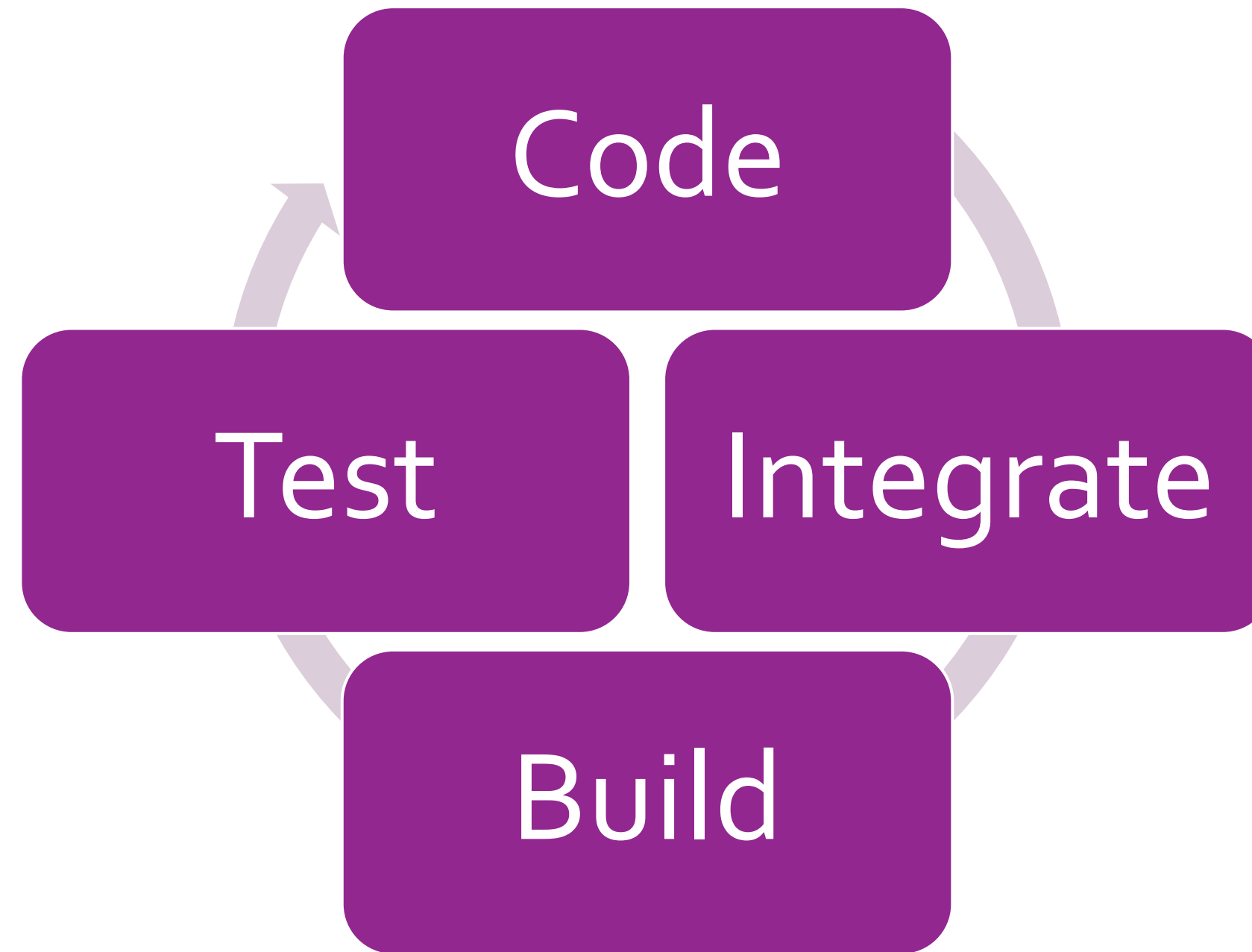
In a nutshell you should be prepared to
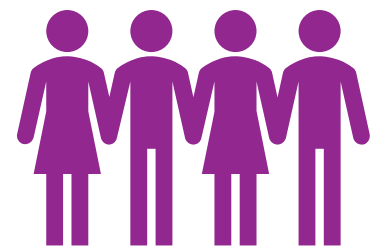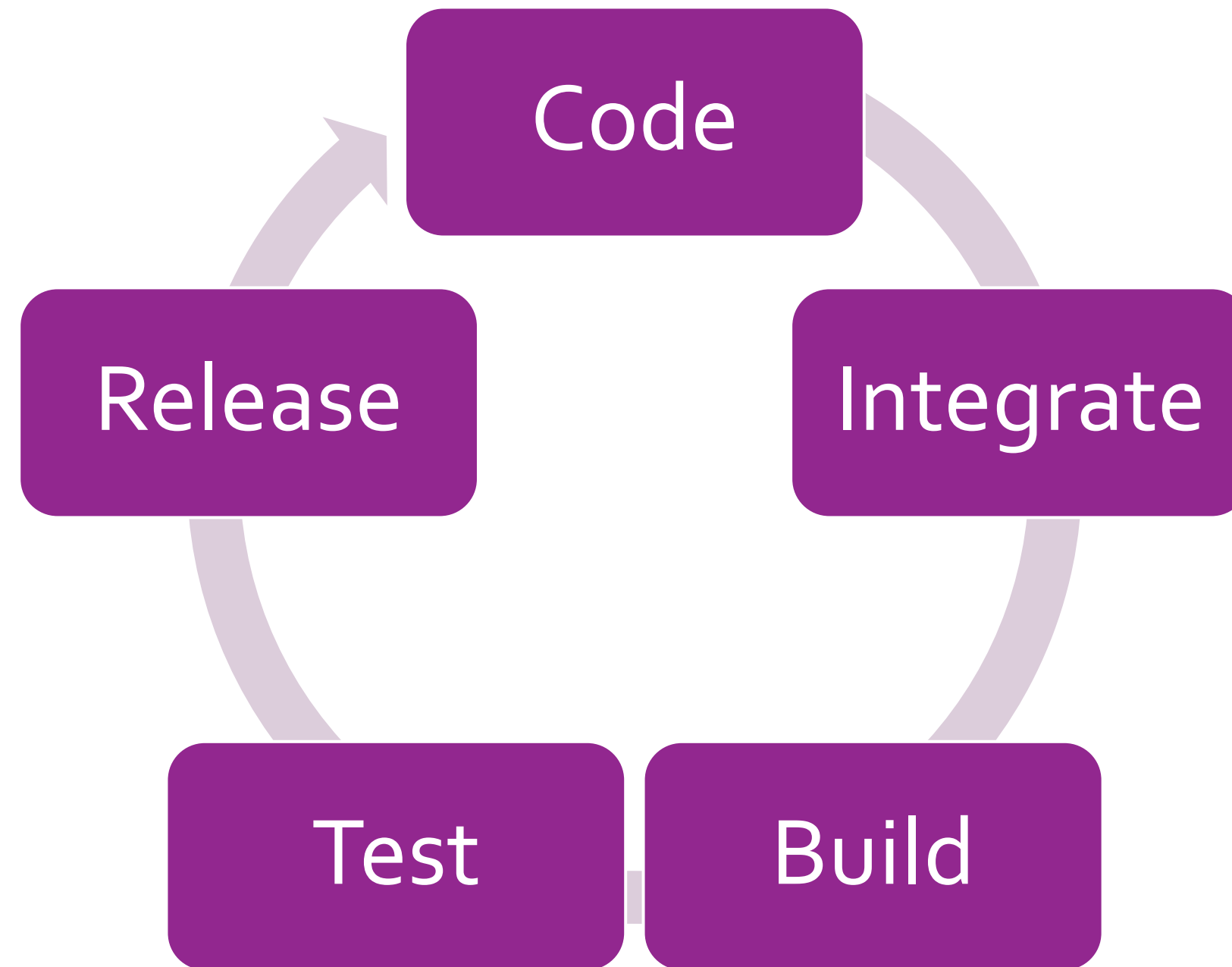
Embrace change   Iterate frequently

Continuous Integration (CI)

Continuous Deployment (CD)

Git Primer

Git is a popular source code management software (aka version control software)

Git was first authored by Linus Torvalds in 2005 for development of Linux

A gross oversimplification of Git

- You have a copy of a repository on a server

- Individuals can have copies on their machine

- You can pull the most recent version from the server to your client

- If there is a difference the changes are merged (automatic or manual)

- You can push changes from your client to the server

- On the server changes may happen automatically or they may need to be reviewed

Git and Commits

- Commits are snapshots of a project

- This is how you save work to a project

- They are associated with a unique hash code and commit message

- Commits are created with the git commit –m "Message"

- Commits consist of the contents of the staging area

- If the staging area is empty, no commit can be created

Staging Area

The command "git add ." adds changed files to a staging area

This is required before making a commit

The staging are is the list of tracked files that will be added to the next commit

Clearing staging area ("git reset") doe not lose your work

Branches

Different versions of the source code can exist simultaneously

These are called branches

Branches have their own history

Branches can be merged

A branch is a time-line of commits

Pull Request (An unfortunate name)

- A notification of intent to merge two branches
- It is done on the server
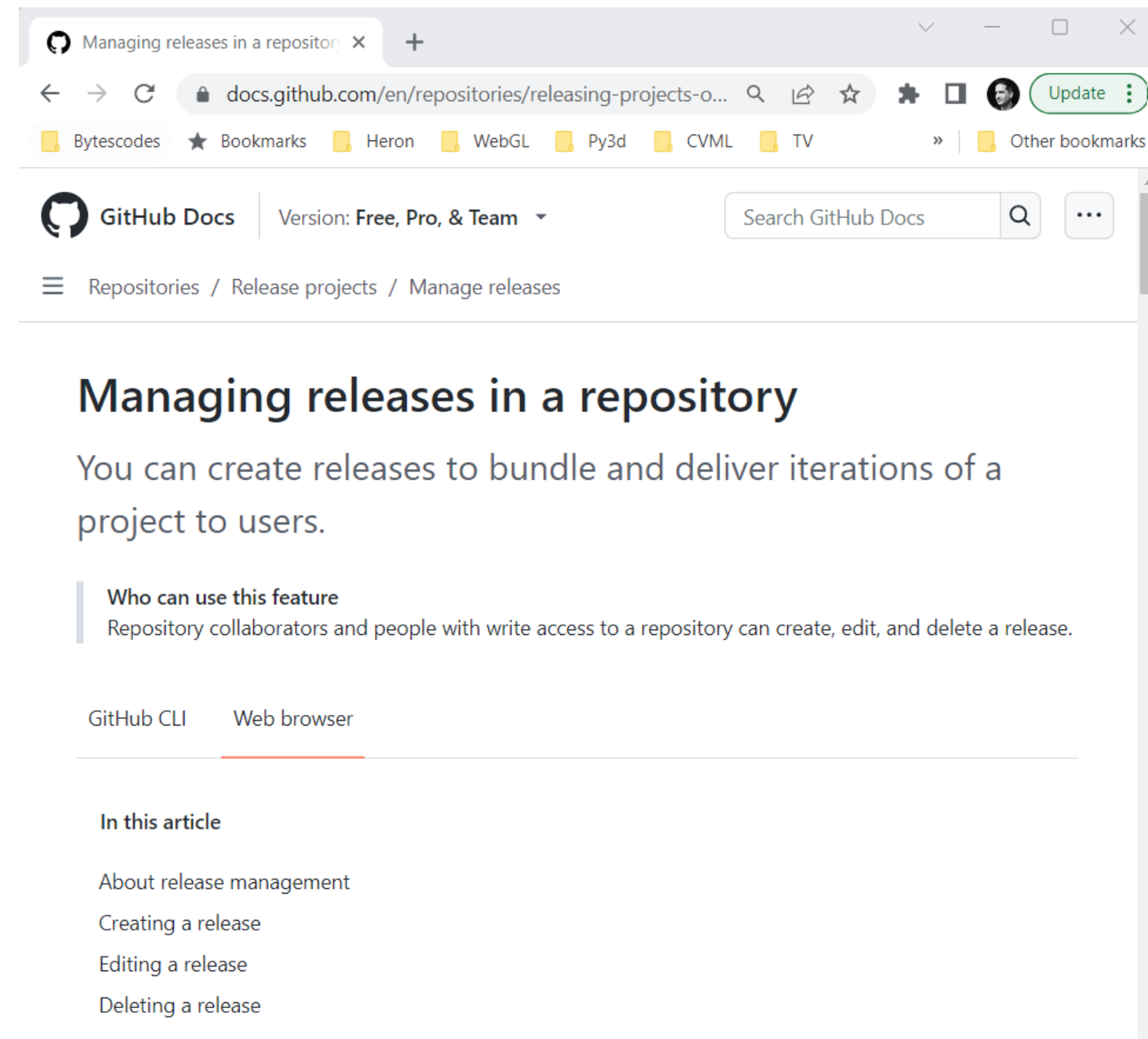- Provides an opportunity for a code review
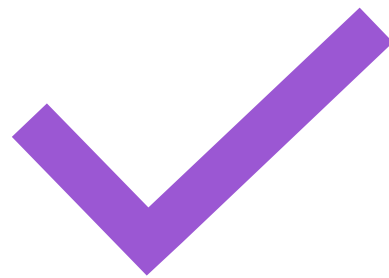- May trigger automated validity checks (CI)
- May trigger automated deployment steps (CD)

Making a Release
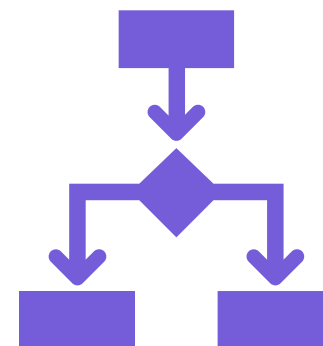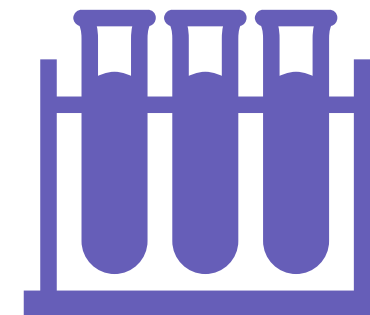
What is "Integration"

We integrate new features into a working project

Involves merging code and resolving conflict (when multiple team works on it)

Testing all features to make sure they don't break (integration testing)

NOT JUST FOR TEAMS

Adding new functionality, or refactoring

ALTERNATING APPROACH

Multi-tasking is hard but it is important to make progress on different things

Alternate Between

Small and big.

Front end and back end.

Hard and easy.

Testing and implementing.

Refactoring and writing new code.

Improving and adding.

Coding and documenting.

Researching and doing.

Known and unknown.

Safe and risky.

Two Parallel Streams

Looks like finished product

Come up with a plan

Faked implementations

Find common systems

Fill out bit by bit

Tackle hard problems

Expect to

Make changes → Uncover things as we go → Discover incorrect assumptions → Discover missing requirements

Find tasks that are harder than we thought → Find features that are easier to implement → Find repetition
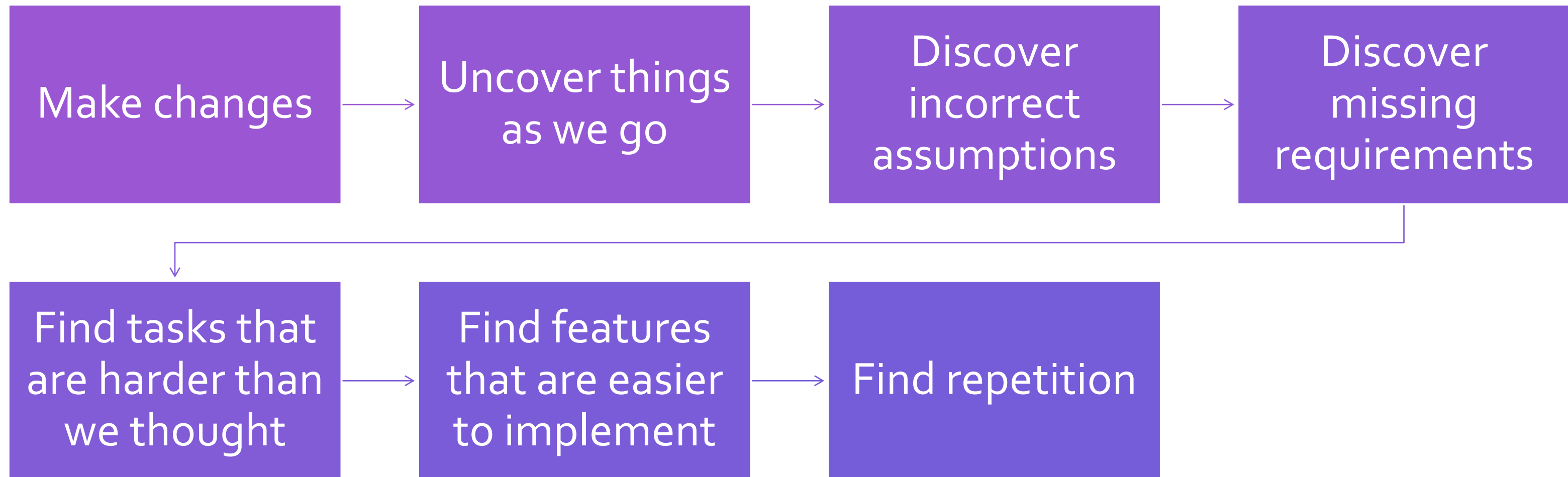
Software Development is Iterative

You don't know what you don't know until you try to build and elaborate

Project Management Tips

- Start from the time you have and work backwards

- Give yourself padding for unpleasant surprises'

- Aim to be done in advance

- Identify the risks

- Identify any dependencies

You Have Four Weeks

- April 4 – General outline of code. Spikes completed.

- April 11 (last day of classes) – All major features implemented.

- April 18 – Everything works, maybe a few bugs.

- April 25 – Finished Reports

Time Management

- There is a clock ticking.

- Software is notoriously hard to estimate.

- Even professionals are terrible at it.

- Can't avoid it, because it is business!

"Best is Enemy of Good" - Voltaire

- « Dans ses écrits, un sage Italien Dit que le mieux est l'ennemi du bien. »

- "In their writing a wise Italian said that best is the enemy of good."

Risk Management

- What are the biggest unknowns?

- These are the things that are hardest to estimate in terms of time.

MIT/Stanford Approach

- **Simplicity** - The design must be simple, both in implementation and interface. It is more important for the interface to be simple than the implementation.

- **Correctness** - The design must be correct in all observable aspects. Incorrectness is simply not allowed.

- **Consistency** - The design must be consistent. A design is allowed to be slightly less simple and less complete to avoid inconsistency. Consistency is as important as correctness.

- **Completeness** - The design must cover as many important situations as is practical. All reasonably expected cases must be covered. Simplicity is not allowed to overly reduce completeness.

Worse is Better Approach

- **Simplicity** - The design must be simple, both in implementation and interface. It is more important for the implementation to be simple than the interface. Simplicity is the most important consideration in a design.

- **Correctness** - The design should be correct in all observable aspects. It is slightly better to be simple than correct.

- **Consistency** - The design must not be overly inconsistent. Consistency can be sacrificed for simplicity in some cases, but it is better to drop those parts of the design that deal with less common circumstances than to introduce either complexity or inconsistency in the implementation.

- **Completeness** - The design must cover as many important situations as is practical. All reasonably expected cases should be covered. Completeness can be sacrificed in favor of any other quality. In fact, completeness must be sacrificed whenever implementation simplicity is jeopardized. Consistency can be sacrificed to achieve completeness if simplicity is retained; especially worthless is consistency of interface.

For the class project

Implementation matter more than interface

Simplicity in implementation first, interface second

Consistency is a key component of simplicity

Completeness does not matter

Correctness matters in implementation

Trade-Offs

Time is fixed

Quality is desired

Functionality is variable

What is Functionality?

- How many features there are

- The range of possible interactions

- The capabilities of each one

What is Quality

- Easy to use
- Discoverable
- Accessible
- Internationalized
- Robust
- Reliable
- Consistent
- Appealing
- Fun
- Performant

Typical Scale of Issues/Bugs

- Trivial

- Minor

- Major

- Critical

- Showstopper

Spikes

- "I would often ask Kent [Beck], 'What is the simplest thing we can program that will convince us we are on the right track?' Such stepping outside the difficulties at hand often led us to simpler and more compelling solutions. Kent dubbed this a Spike. I found the practice particularly useful while maintaining large frameworks."

- https://wiki.c2.com/?SpikeSolution - Ward Cunningham

Technical Spikes / Investigations

- This is part of risk management and time management

- Implementing all features in context of full program can make it harder

- Consider working on a separate project

- Maybe it is just a series of unit tests

- Sometimes we put in back-doors to an application

- Involves some reading and writing

- Start with a hypothesis

- Find out what you can about the problem and solutions

Application Back-door

- Like game cheat codes.

- Allow game developers to test features, without having to play whole level

- Usually disabled in production, but maybe leave it in as an easter-egg.

Ask yourself

- What is the simplest thing that could possibly work.

Ask others

- Do you have any suggestions or ideas?

Menus, Accelerator and Shortcut Keys

-

- https://learn.microsoft.com/en-us/dotnet/desktop/winforms/controls/how-to-provide-standard-menu-items-to-a-form