WASHINGTON STATE UNIVERSITY
TRI-CITIES

SOFTWARE PROCESS

Chapter 03

Spring 2023

E

AGILE – WHAT IS IT?

- Customer satisfaction
- Early and continuous delivery
- Embrace change
- Frequent delivery
- Collaboration of businesses and developers
- Motivated individuals
- Face-to-face conversation
- Functional products
- Technical excellence
- Simplicity
- Self-organized teams
- Regulation, reflection and adjustment

E

AGILE MANIFESTO

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes 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.

SCRUM – THE WHAT

- An agile process model
- Scrum – comes from the sport - Rugby
- Incremental development
- Timeboxing
  - Break up your efforts into 2-3 week developments called sprints
  - Fit as many activities as you can into this time window

E

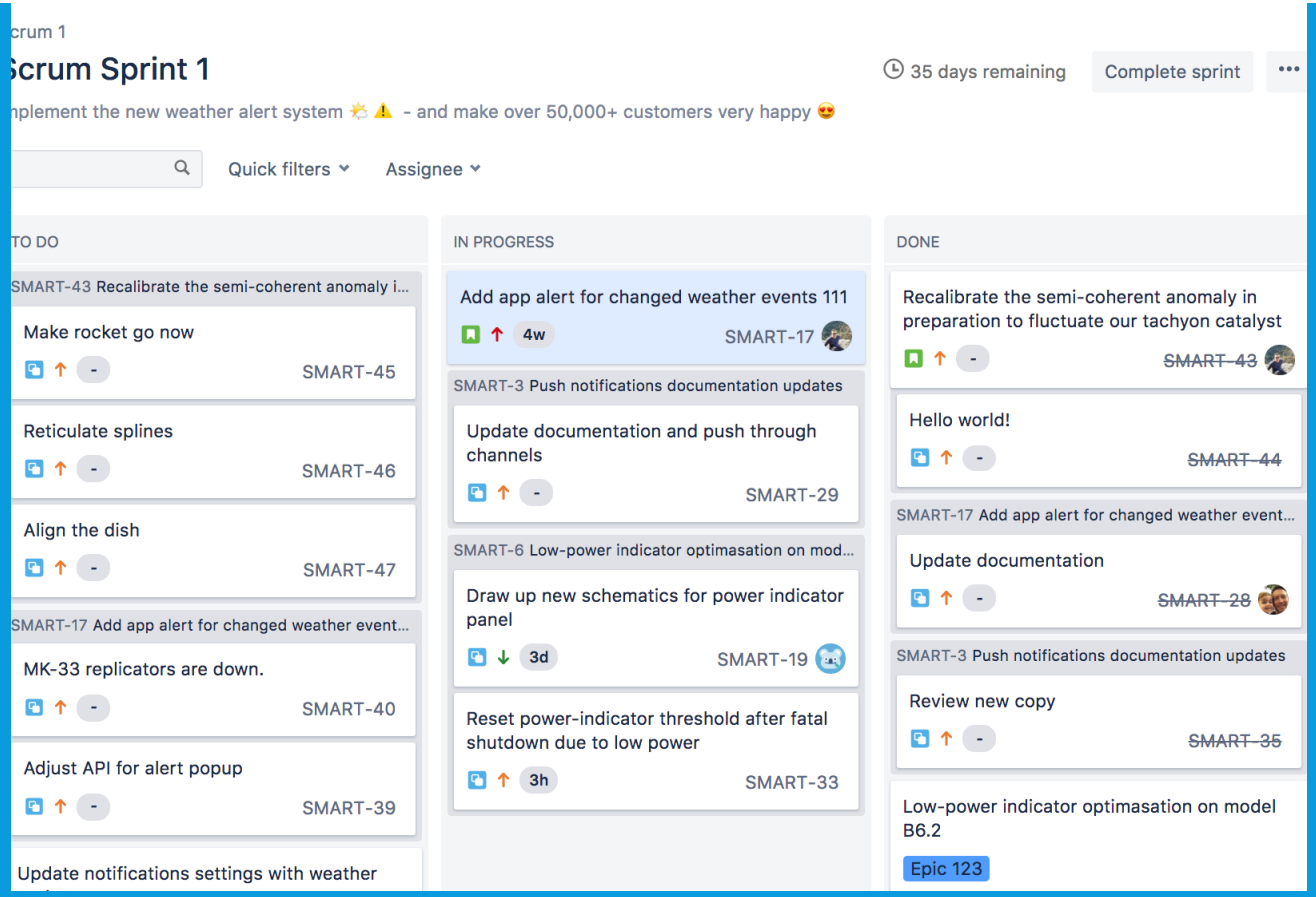
SCRUM – THE WHO

- Product Owner – stakeholder – holds the vision for the product – often works with the client
- Scrum Master – person who keeps the team on pace, works with product owner to develop scope, and ensure tasks are being completed
- Development Team – implements code, works with scrum master to identify risks, bottlenecks, etc.

SCRUM – SPRINT BOARD / BACKLOGS

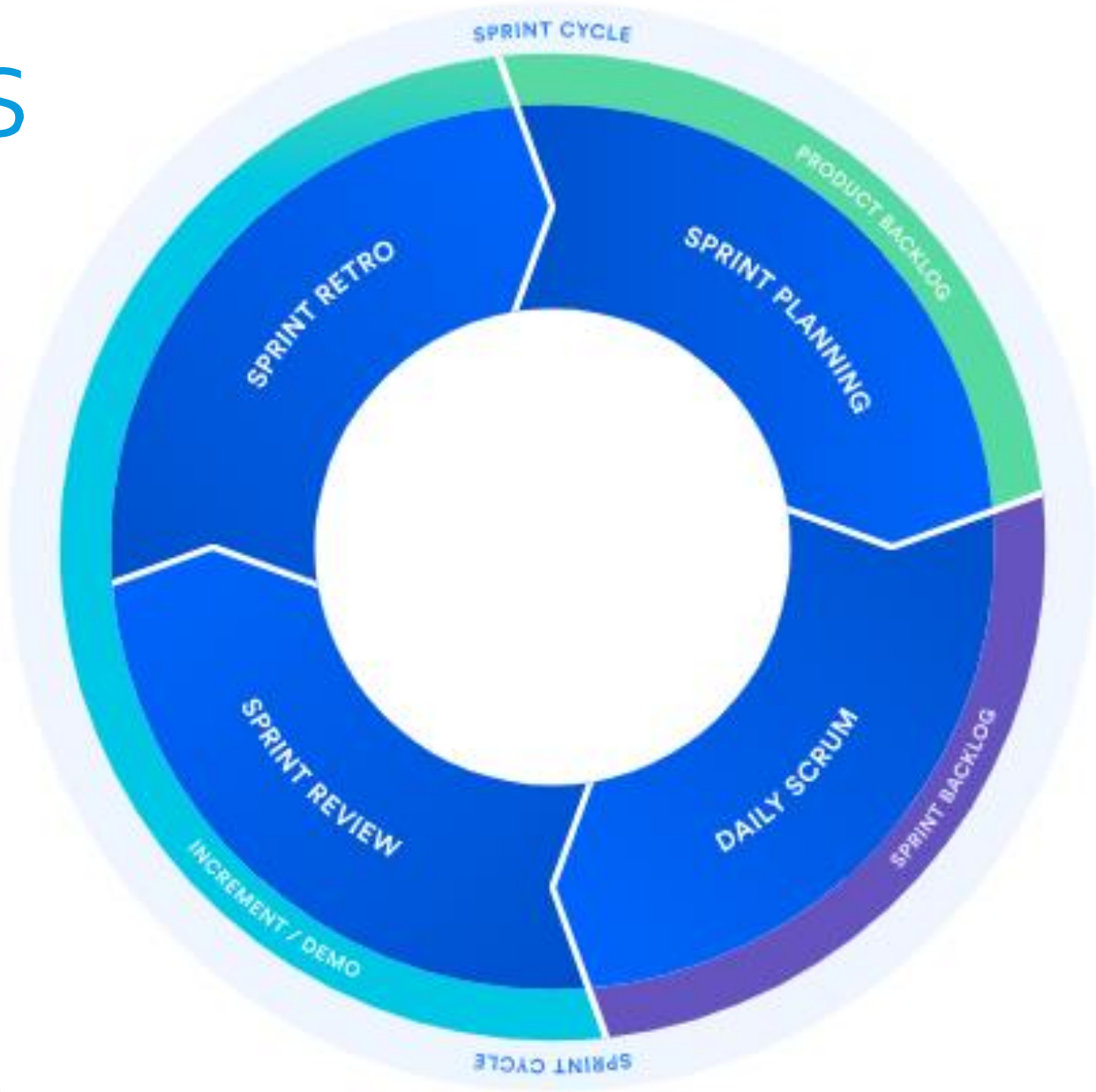
- **Sprint board** – an electronic board that displays tickets in swim lanes where each column is the state of a particular ticket

- **Backlog** – a collection of tickets presented on a board often which are organized based on priority. The topmost items are often the items that are highest priority



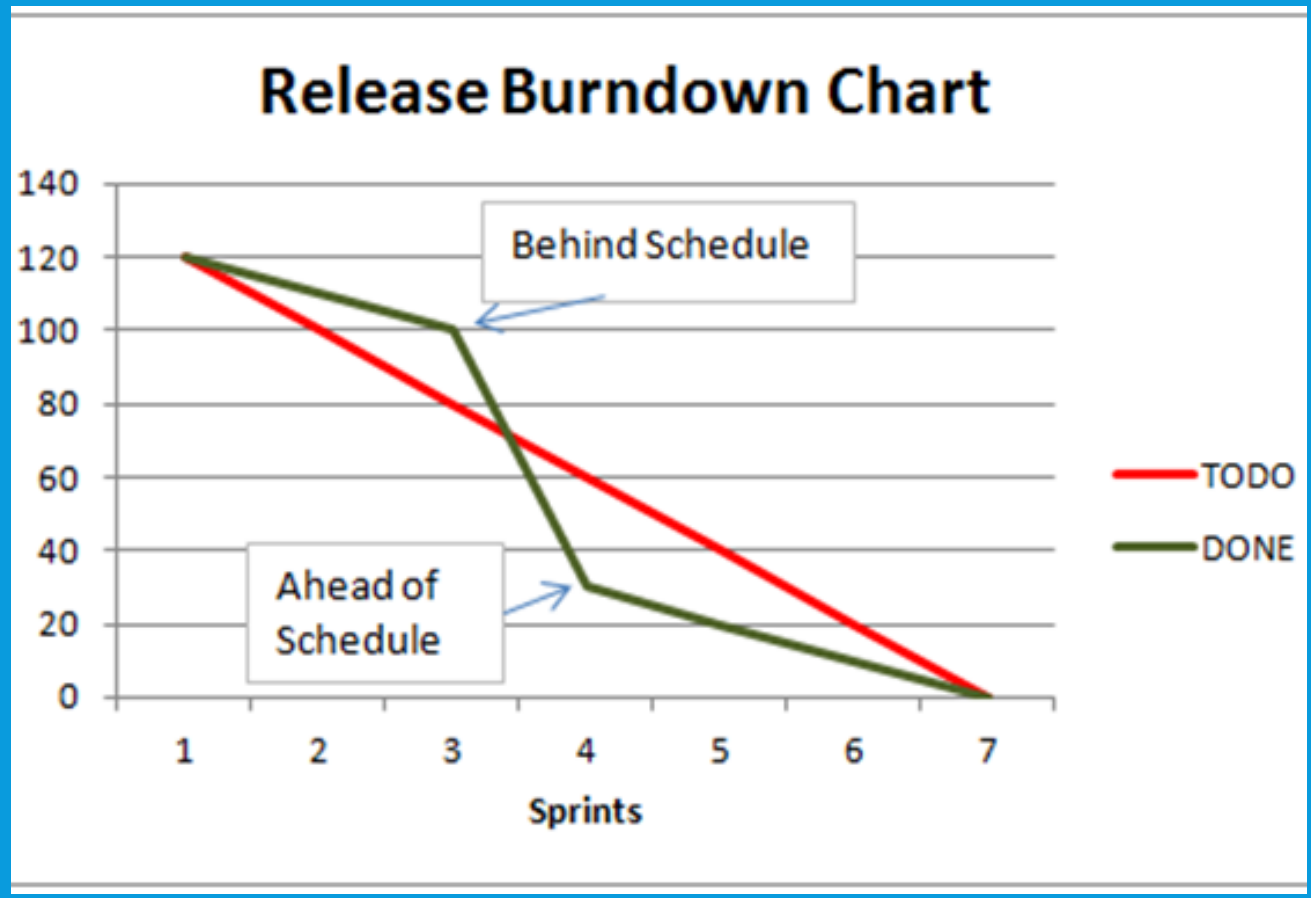
SCRUM – THE PROCESS

- Planning
  - Backlog grooming
  - Integration of new or planned tasks
- Daily Scrum
  - Updates / meetings that are performed - **standups**
- Review
  - Demos, discussion about how things were integrated
- Retrospective
  - Discussions about how to fix the process



SCRUM – TIME AND CHANGES

- One key thing about Scrum – once a sprint starts, the team must carry out that sprint - there is no room for adjusting the sprint
- Time is measured in "story points"
  - 1, 2, 4, 8, 16
- Velocity – metric for measuring how fast things are moving



E

KANBAN – THE WHAT

- Another agile / incremental approach
- No-timeboxing – focused on continuous delivery
- Toyota – 1940's – ***just in time*** manufacturing
  - Lean manufacturing

- More amenable towards web-development where deployment of software is much easier, e.g. when compared to desktop apps

E

KANBAN – WHO / HOW / TIMING

- Kanban Board
  - Requested
  - In-Progress
  - Done
- Kanban does not have required roles
- Kanban measures progress using three metrics vs. velocity

KANBAN – TIME METRICS

Metric	Definition	Why?
Lead Time	time between customer request to delivery	Is the customer waiting a long time?
Cycle Time	amount of time it takes to go from work started to work delivered	How long is taking us to get this out?
WIP	number of task items that a team is currently working on	Are we overworking our team?

Metric	Definition	Why?
Velocity (Scrum)	The number of story points being accomplished per unit time	How fast are we moving?

INCREMENTAL

Good For

- Projects with changing requirements
- Projects where the client is unsure what they need
- Customer needs a small aspect of the software

Bad For

- Unmotivated teams
- Large organizations that may have their own policies that conflict with the more *"informal"* agile methods
- "Process" not visible to managers because activities can be concurrent
- System structure can degrade over time

DIFFERENCES?

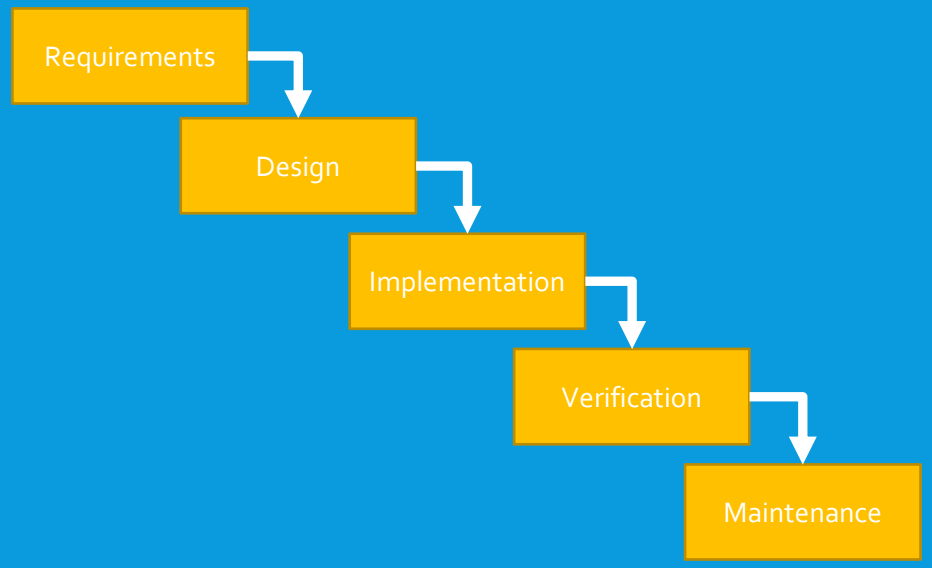
Waterfall

- Favors documentation and complete analyses of requirements and design
- Product delivered at the end of the whole process

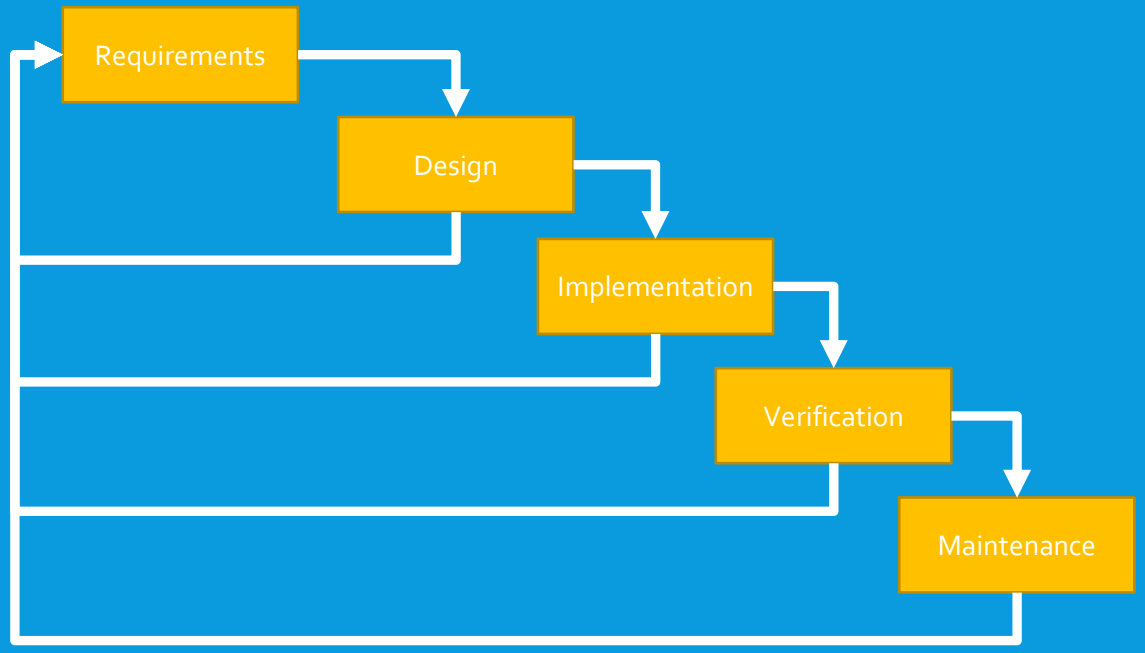
Incremental

- Reduction of costs for changing requirements because documentation and analyses are typically fluid
- Product can be delivered in stages

Waterfall



Incremental

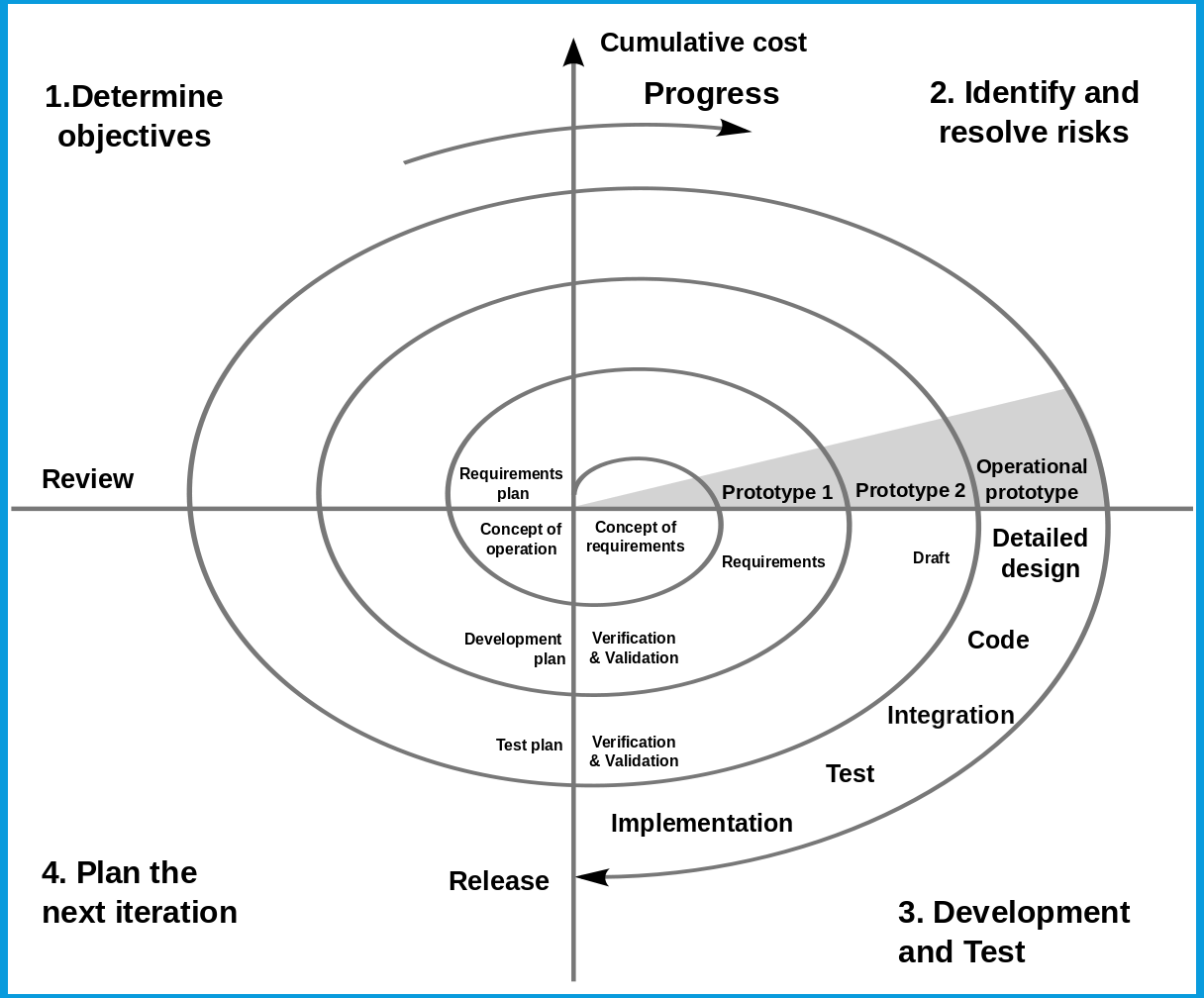


At any phase of the approach – requirements analysis and specification may be revisited

INCREMENTAL

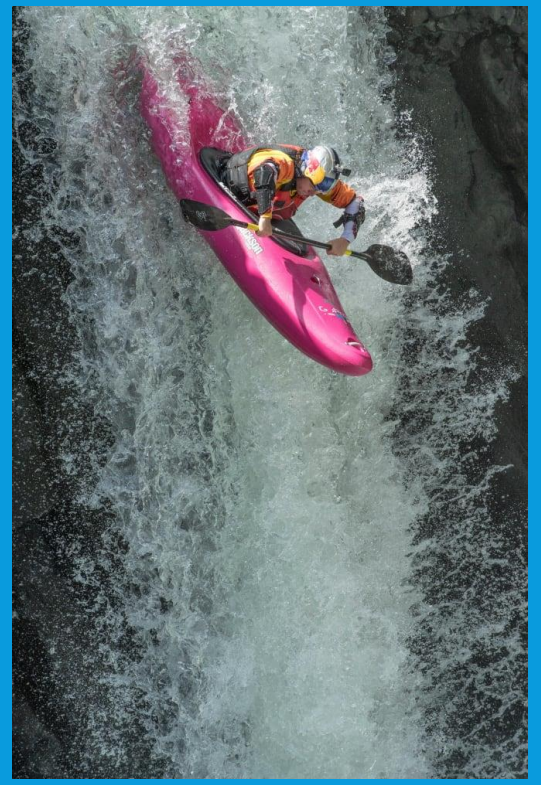
The spiral model is one of the best ways to understand how incremental models work

1. Determine Objectives
2. Identify Resol



THIS CLASS?

- We are going to use the **Waterfall Process**
- Why?
  - 3 months (15 weeks)
  - Competing priorities (classes)
  - Learning – going through the simplest process first gets your feet wet (no pun intended)



Most of what you will all do

PROCESS ACTIVITIES

SOFTWARE SPECIFICATION

- Process of understanding the system

- What are the constraints?
- What systems already exist?

- Often times company's / teams will perform market research, feasibility studies

- What we want at the end of the day – "a document" – software requirements specification (SRS) – that captures the stakeholder / clients requirements.

SOFTWARE SPECIFICATION ACTIVITIES

- Requirements Elicitation and Analysis – process of deriving the system requirements
  - Observation of existing systems
  - Discussion with users, customers, existing systems
- Requirements Specification – activity of translating requirements into a document
- Requirements Validation – Activity to check that the requirements are real, consistence, and complete
  - Completed internally
  - Work with customers to perform activity

PROJECT

- You'll be given a project description
- You'll have a few class periods to come and ask questions – elicit requirements
- You'll have some time to write these requirements
- You'll have some class periods to validate your specification

SOFTWARE DESIGN AND IMPLEMENTATION

- Process of building the executable
- Translation of requirements into design and then implementation

- Design – description of the software
  - Behavior – flow of data, sequences of events
  - Structure – relationships between components

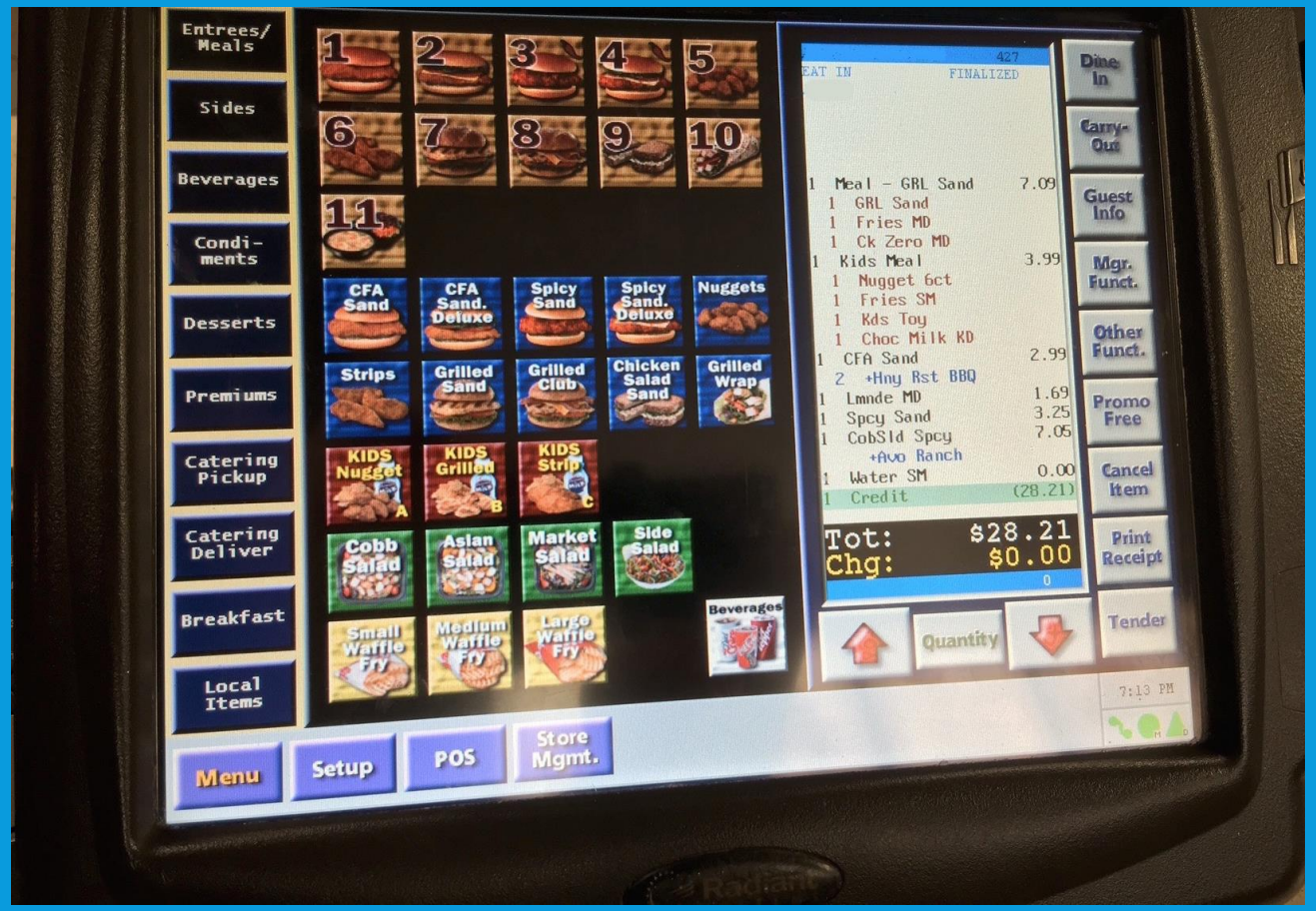
- Implementation – "coding"
-

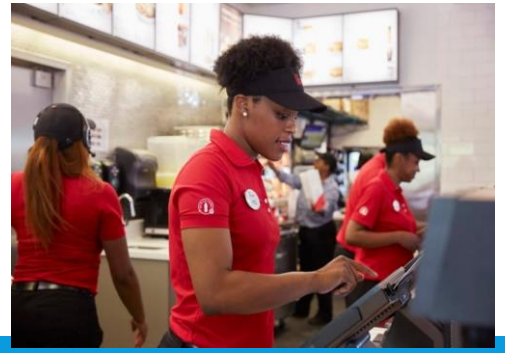
SOFTWARE DESIGN AND IMPLEMENTATION

- Design and implementation can happen concurrently
  - Mostly in agile models
- Design then implementation
  - Waterfall

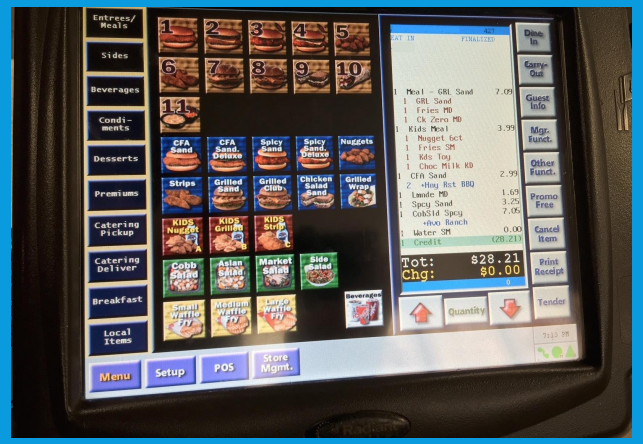
- Main Activities
  - Architectural Design – overall structure – layout of principle components
  - Database Design – structure of data, relationships, etc.
  - Interface Design – interfaces between components
  - Component Selection and Design – identification of reusable components


Chick-fil-A


Entrees/ Meals
Sides
Beverages
Condi- ments
Desserts
Premiums
Catering Pickup
Catering Deliver
Breakfast
Local Items
1
2
3
4
5
6
7
8
9
10
11
CFA Sand
CFA Sand. Deluxe
Spicy Sand
Spicy Sand. Deluxe
Nuggets
Strips
Grilled Sand
Grilled Club
Chicken Salad Sand
Grilled Wrap
KIDS Nugget
KIDS Grilled
KIDS Strip
Cobb Salad
Asian Salad
Market Salad
Side Salad
Small Waffle Fry
Medium Waffle Fry
Large Waffle Fry
Beverages
EAT IN
FINALIZED
1 Meal - GRL Sand 7.09
1 GRL Sand
1 Fries MD
1 Ck Zero MD
1 Kids Meal 3.99
1 Nugget 6ct
1 Fries SM
1 Kds Toy
1 Choc Milk KD
1 CFA Sand 2.99
2 +Hny Rst BBQ
1 Lmnde MD 1.69
1 Spcy Sand 3.25
1 CobSld Spcy 7.05
+Avo Ranch
1 Water SM 0.00
1 Credit (28.21)
Tot: $28.21
Chg: $0.00
Dine In
Carry- Out
Guest Info
Mgr. Funct.
Other Funct.
Promo Free
Cancel Item
Print Receipt
Tender
Quantity
7:13 PM
Menu
Setup
POS
Store Mgmt.
Radiant



User Management

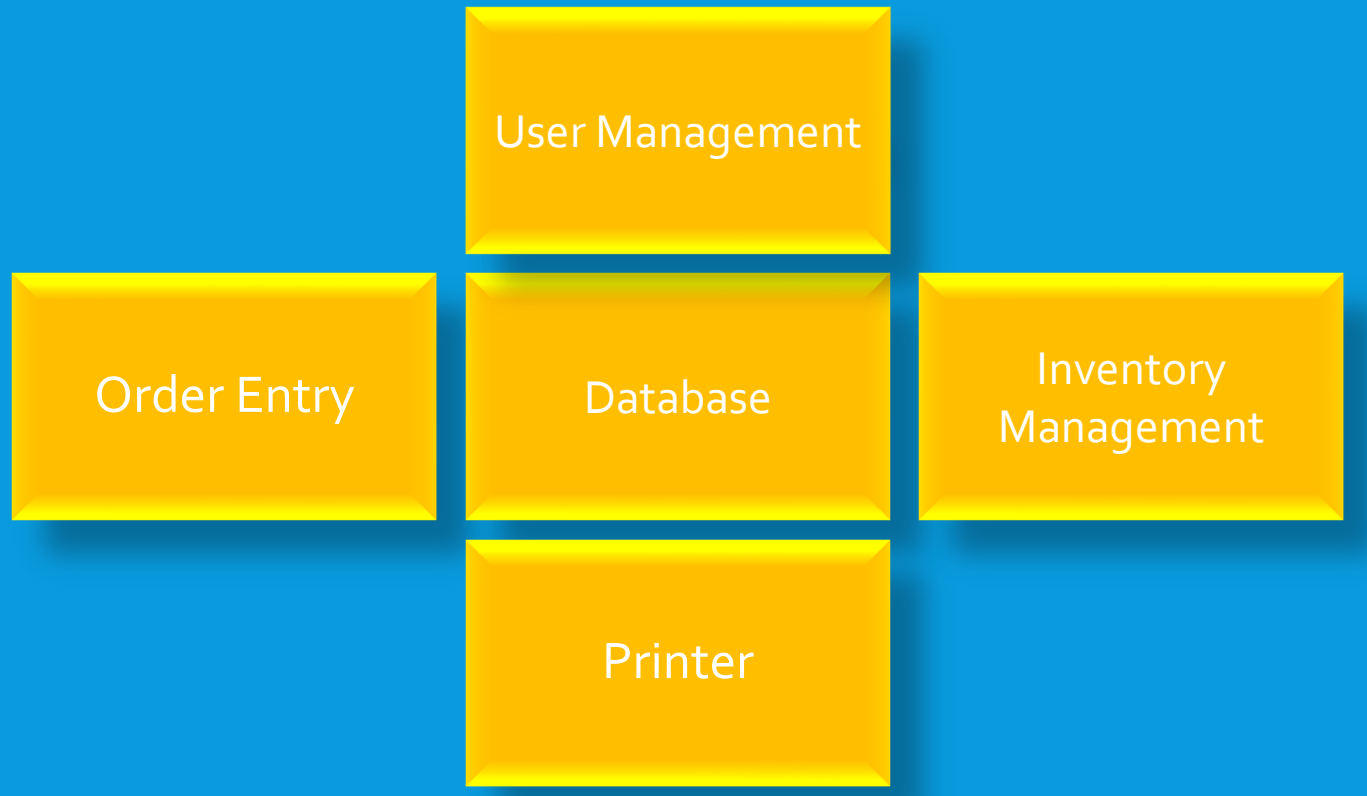


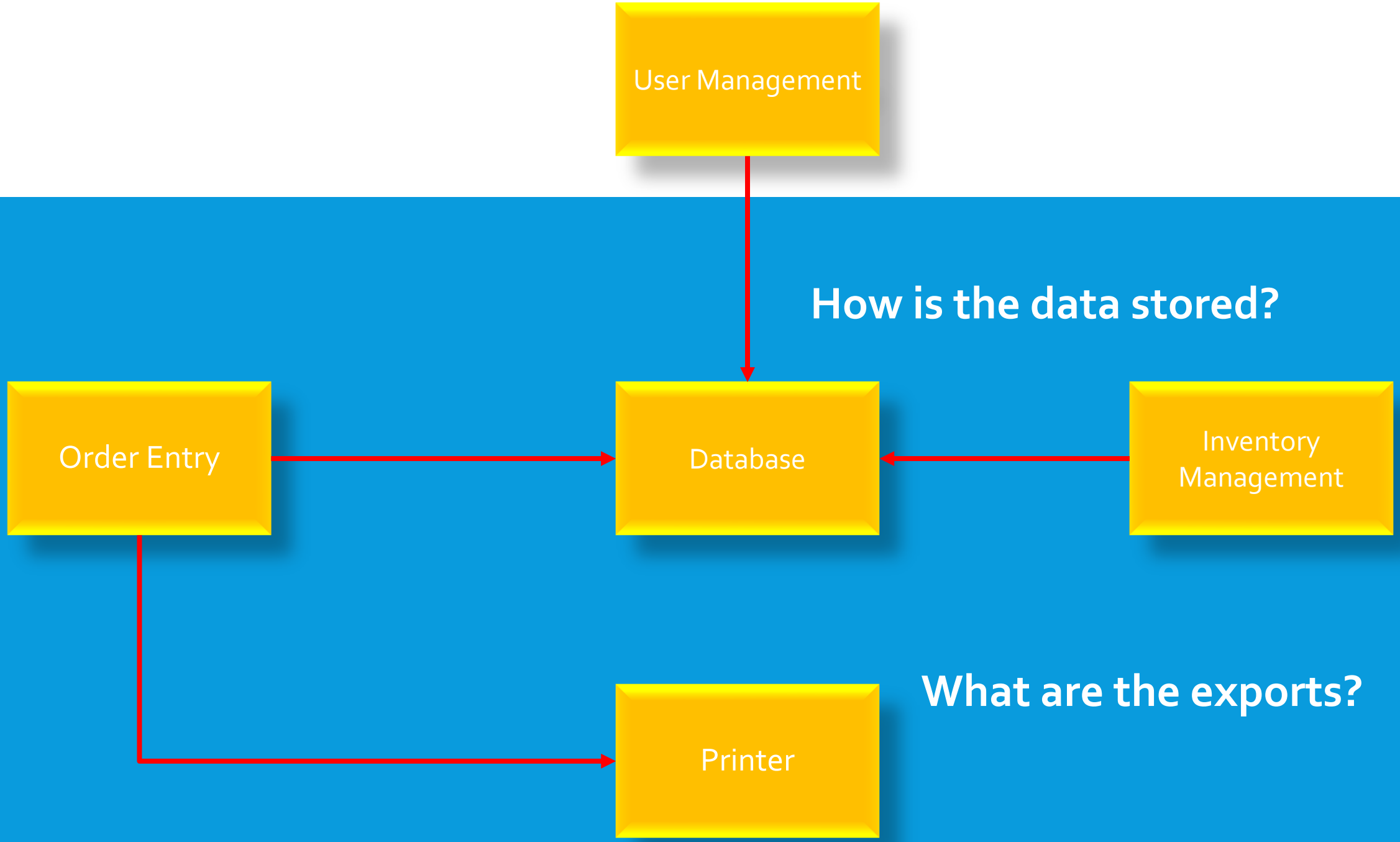
Order Entry

Database

Inventory Management

Printer


User Management
Order Entry
Database
Inventory Management
Printer


User Management
How is the data stored?
Order Entry
Database
Inventory Management
What are the exports?
Printer

SOFTWARE VALIDATION

- V&V – Validation and Verification

- Validation – Are the requirements correct?
- Verification – Did we build it correctly?

SOFTWARE V&V

- Component Testing – early stages of implementation
- System Testing – later stages of implementation
- Customer Testing – before delivery / as part of delivery

SOFTWARE V&V – COMPONENT TESTING

- Focuses on individual components
- Unit Testing – a method of verifying that individual classes, components function *as required*
- Use of simulated data often part of this testing process
- Answers Questions
  - Does the printer code work correctly?
  - Does our order system add correctly?

SOFTWARE V&V – SYSTEM TESTING

- Focuses on integration of components
- Do your interfaces work correctly?
- Also uses simulated data
- Answers Questions
  - When an order is submitted, does the inventory amounts update correctly?

SOFTWARE V&V – CUSTOMER TESTING

- *Beta* testing

- Does the customer find any issues?
- Does the customer *accept the product?*

- Answers Questions
  - Does the system meet the customers needs?

SYSTEM V&V - CONCEPTS

- In waterfall – you are going to implement the project, and perform testing as the last stage before release. When you get to customer testing you are basically delivering the product

- In incremental – you may deliver a product (beta) version to the user for verification – and the product may not be complete. You may use customer interaction to gain further feedback and help refine requirements or generate new ones.

SOFTWARE V&V - TOOLING

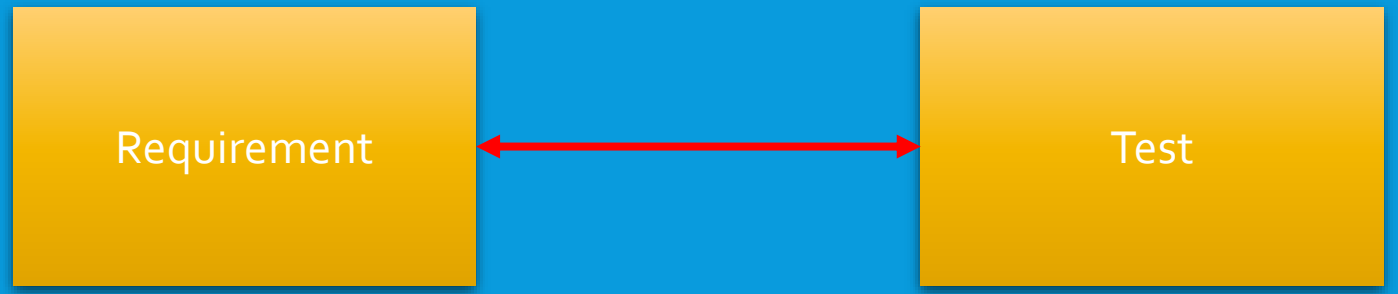
- You'll later learn about different tools for testing (this course and others)
- Some Key Tools
  - Track Your Work - Bug Tracking System (e.g. Trello, JIRA) to help track defects / bugs
  - Write Software Tests - Testing Frameworks for Component and System Testing (e.g. NUnit, JUnit, MSTest)
  - Track Manual Tests Testing Plug-ins (e.g. Zephyr) for managing and tracking test data

SOFTWARE V&V - TRACEABILITY

- 6-month Principle
  - 6-months from now you're going to look at code and go…What The Fudge? (WTF)

- One of the hardest things to maintain are tests.
  - Component Tests (e.g. whitebox) that are not properly documented
  - Unit tests **WILL** break – and that's a good thing
  - What are they? Why are they important?

SOFTWARE V&V - TRACEABILITY

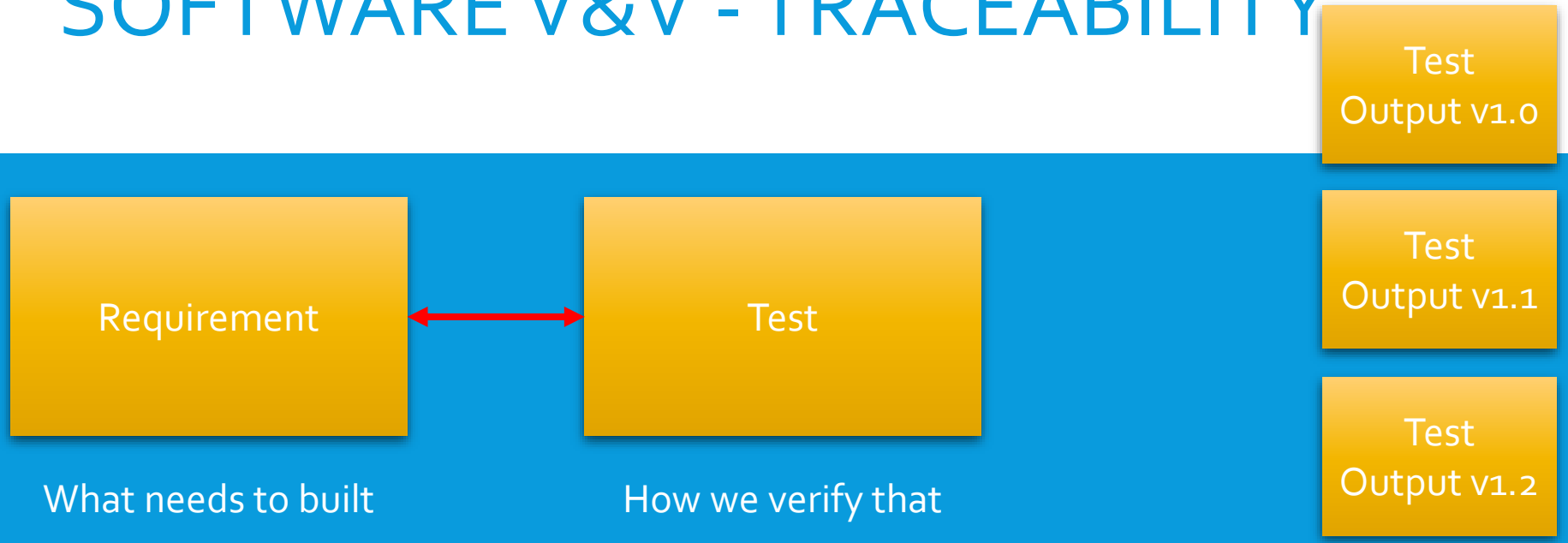


What needs to built

How we verify that

Traceability is one of the most key aspects in software engineering

SOFTWARE V&V - TRACEABILITY



Good teams will also track test data per version (inputs and outputs)

SOFTWARE EVOLUTION

- Software changes
- Requirements change
- Tests will have to change
- Systems become more and more complex
- Versioning and Tracking IS IMPORTANT

VERSIONING

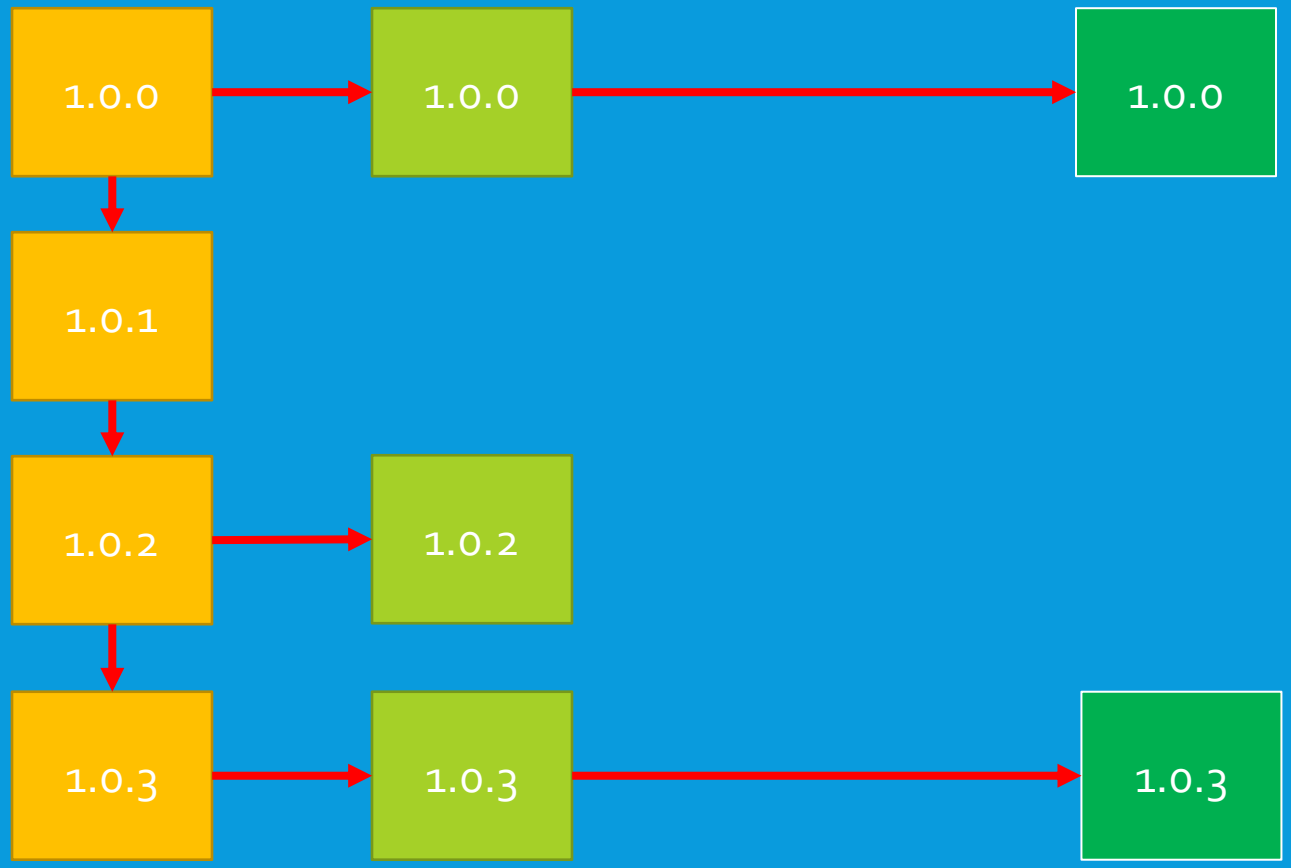
- There are different ways to version your software –
  - Major.Minor.Patch (https://semver.org/)
  - Major.Minor.Build
  - Major.Minor.Revision
  - Date
- Honestly, there is no right way – what is important is to capture differences in the software
- Versions really only mean something to the development team

VERSIONING

Major.Minor.Patch

- MAJOR version when you make incompatible API changes,
- MINOR version when you add functionality in a backwards compatible manner
- PATCH version when you make backwards compatible bug fixes.
- https://semver.org/

HOW DOES VERSIONING RELATE TO PROCESS?



Your software verification phase may have multiple stages

It's important to track the candidates for releases – so that defects can be properly tracked and verified.

TESTING AND RELEASE

- We typically have concurrent dev, test, and release/acceptance activities occurring
- It's really important to communicate and stay organized

COPING WITH CHANGE

- Change is inevitable – and staying organized is key

- Approach software development KNOWING and ACCEPTING that requirements will change based on
  - Business needs
  - Competition
  - Priorities

PROTOTYPING

- System Prototyping – a way to handle expected or anticipated changes
  - Focuses on small parts of a system – how might it look? Is this feasible? Show a customer and get their feedback

- Incremental Delivery - Delivering prototypes to customer (CD)

- Amazon, Google, etc. will actually test features on you. You may see a different site than your friends and may not even know it.

PROTOTYPING – WAIT WHAT IS IT?

- Early version of software or feature that is used to demonstrate proof of concepts.
- Good way to get something in front of a user (stakeholder more often than not) so they can provide feedback – you can collect useability data
- Used to help control costs – sometimes some of the key steps in the engineering process are minimized (e.g. design)

PROTOTYPING – THE PITFALL

- Sometimes prototypes are just adopted as the end product. The customer likes the prototype and it may be accepted as useable
  - The concept of a throwaway prototype is not enacted
- The design, however, may not be extensible (i.e. easy to modify)
- When you develop a prototype attempt to still develop as though you are writing an end product

PROCESS IMPROVEMENTS

- Process improvement is the act of trying to reduce costs and deliver quicker
- Sommerville describes two
  - Process Maturity – focuses on improving process and project management
  - Agile – focuses on iterative development and reduction of overheads

PROCESS MATURITY

- Process Measurement
  - Establish attributes to measure how well you are maintain cost and schedule
- Process Analysis
  - Identification of weaknesses and bottlenecks
- Process Change
  - Propose changes to the process to fix identified weaknesses

CAPABILITY MATURITY MODELS

- Initial - CMM1
  - goals associated to process area are satisfied and communicated to the team
- Managed - CMM2
  - goals are associated with process area, organized policies are in place that define when each process should be used
- Defined – CMM3
  - Organizational standardization and deployment of processes
  - Process assets and measurements are collected and used to make improvements qualitatively

CAPABILITY MATURITY MODELS

- Quantitatively Managed - CMM4
  - Statistical methods are used to control sub-processes – used in process management
  - Compared to Defined – (CMM 3) processes are quantitatively vs. qualitatively predictable.
- Optimizing – CMM5
  - Use process and product measurements to drive process improvements. Trends are established an analyzed to adapt to changing business needs
  - Focuses on shifting the trend in performance – vs. – fixing just special cases identified (CMM4)

EXPERIENCE…

- No one software process works for all projects
- No one software process works for all teams
- Software processes should be allowed to evolve
  - New clients
  - New requirements
  - Competition
- Software processes should be reviewed regularly
- **DEFINE** your process – communicate – and get buy in!

EXPERIENCE…

- Software processes are intended to keep teams organized and coordinate activities across teams or people
- Tooling has come a long way – use electronic tools where you can
- Establish patterns to create repeatable work – use best practices, workflows, that are established within industry
  - DON'T CREATE YOUR OWN! Not right now anyway

COMMUNICATE, COMMUNICATE, COMMUNICATE

QUESTIONS?