Agile Principles

CEN 4010 Intro to Software Engineering

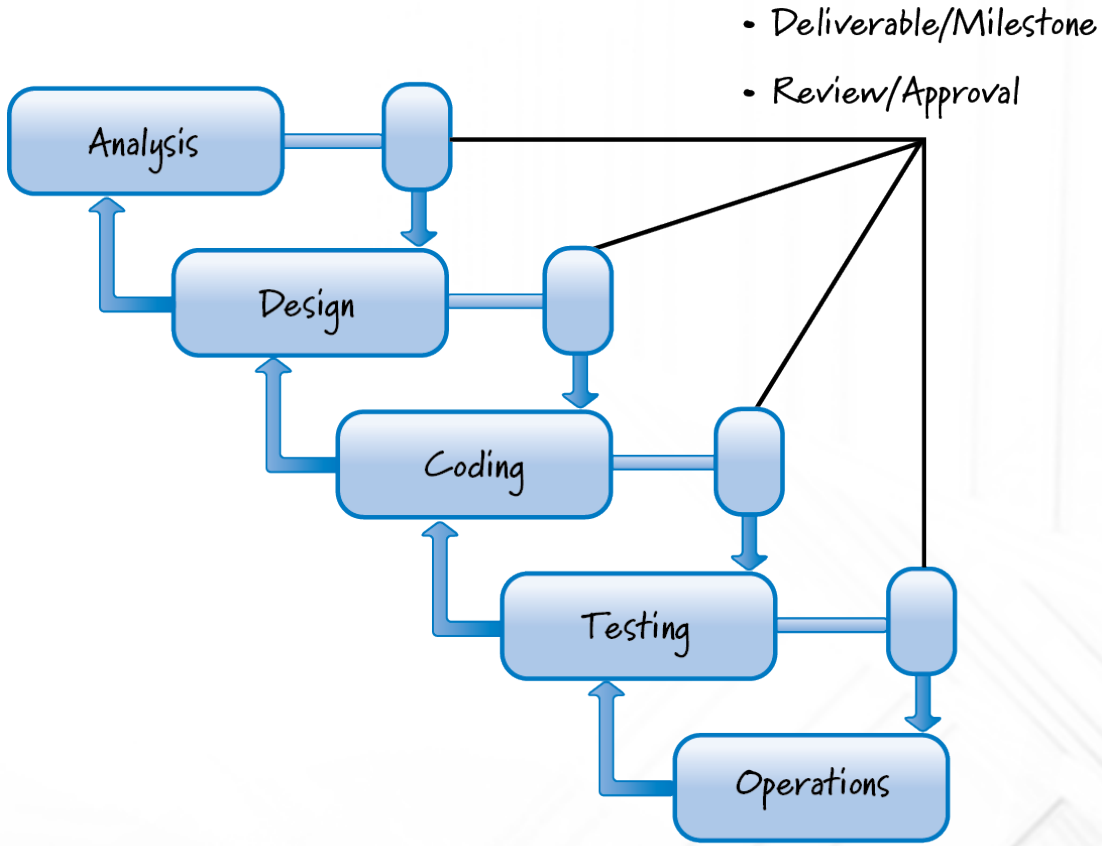
Professor Alex Roque

Plan Drive Sequential Development

- Waterfall is the most popular example. This would work great in a domain where there are few unknowns, such as manufacturing.
- **PDSD** attempts to plan for and anticipate **up front** all of the features a user might want in the end product and to determine how best to build them
  - **Idea** -> Better Plan -> Better Understand -> Better Execution
  - **Works well** if applied to problems that are **well-defined**, **predictable**, and unlikely to undergo **significant change**
- **Scrum** (and other agile practices) is based on the **beliefs** that map well to problems having **enough uncertainty** to make **high levels of predictability difficult**

Plan Drive Sequential Development

- Waterfall model does work, if there are very few undefined problems...(say building a house).



Agile Manifesto Revisited

Individuals and interactions	over	Process and tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

Agile Principles

1. Variability and Uncertainty
2. Prediction and Adaptation
3. Validated Learning
4. Work In Process (WIP)
5. Progress
6. Performance

Principle 1: Variability and Uncertainty

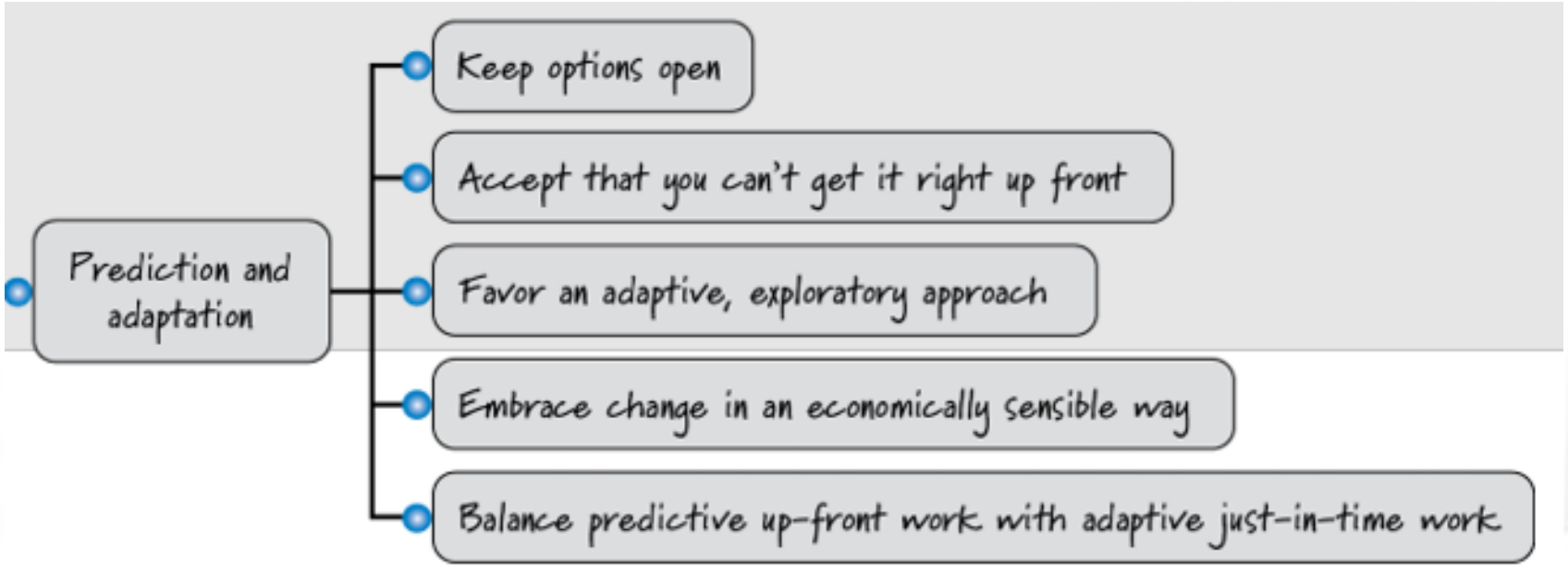
- Plan-Driven processes treat product development like manufacturing – **they shun variability** and encourage **conformance** to a defined process
- Product development is not like product manufacturing
  - Create a **single instance** of a product rather than manufacture **multiple instances**
- Employ **Iterative** and **Incremental Development**
  - **Iterative** acknowledges that we will probably get things wrong before right
  - **Incremental** based on age-old principle of build some before you build it all
- Leverage Variability through **Inspection, Adaptation,** and **Transparency**
- Reduce All Forms of **Uncertainty Simultaneously**
  - **End** uncertainty – uncertainty surrounding the features of the final product
  - **Means** uncertainty – uncertainty surrounding the process and technologies used
  - **Customer** uncertainty – uncertainty about who wants/needs (start-ups assume)

Principle 1: Variability and Uncertainty

- Comparison of Plan-Driven and Scrum Processes

Dimension	Plan-Driven	Scrum
Degree of process definition	Well-defined set of sequential steps	Complex process that would defy a complete up-front definition
Randomness of output	Little or no output of variability	Expect variability because we are not trying to build the same thing over and over
Amount if feedback used	Little and late	Frequent and early

Principle 2: Prediction and Adaptation

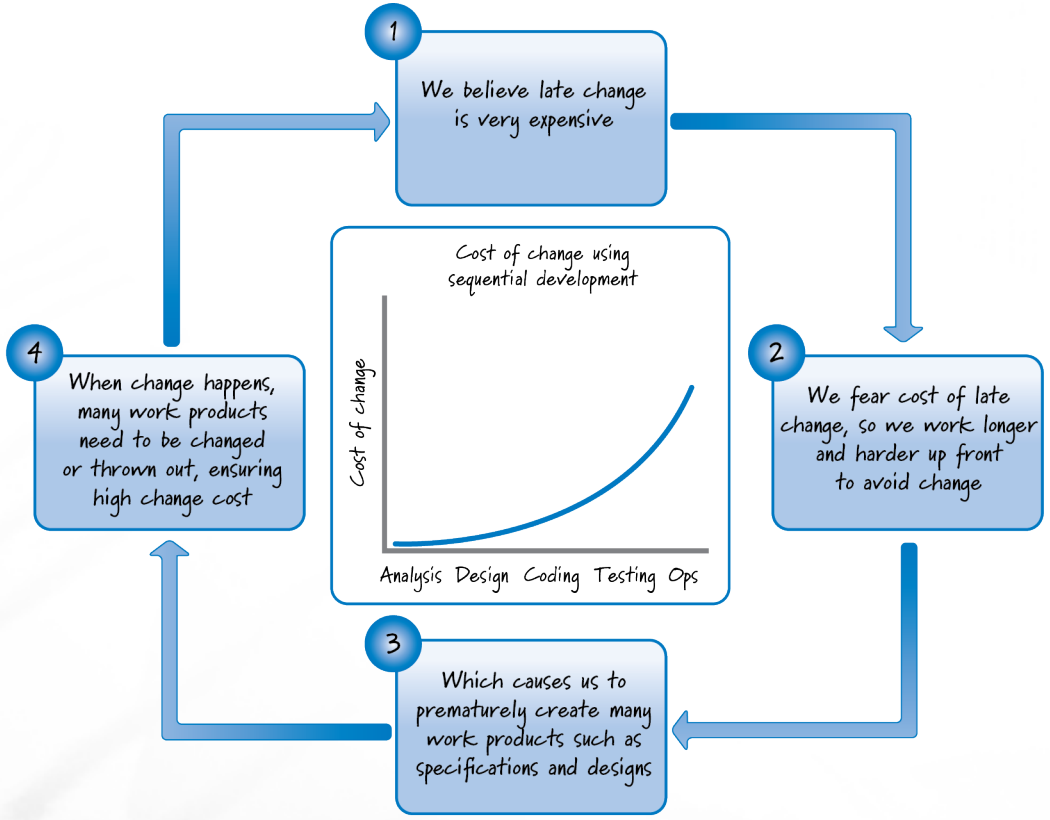


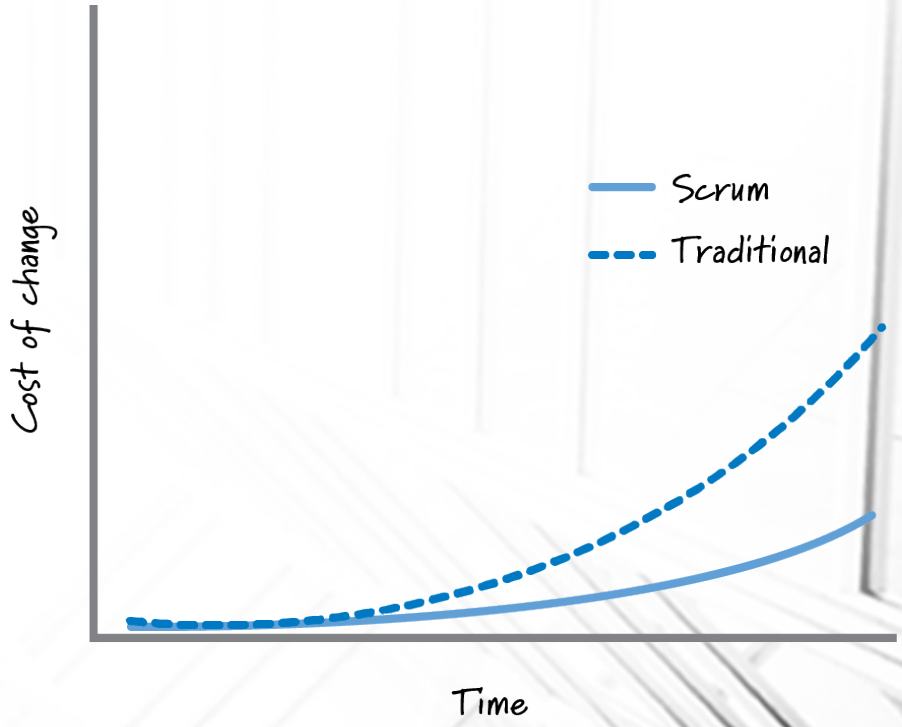
Principle 2: Prediction and Adaptation

- Keep options Open: The strategy in Scrum is to keep options open and delay commitment until we have the most information available. A decision should be made when the cost of not making a decision becomes greater than the cost of making a decision.
- Accept that you cant get it right up front: We know in scrum that we most likely won't have all the decisions up front due to variability and complexity of the project.
- Adaptive, exploratory approach: Scrum favors exploration to gain knowledge (building proof of concepts, studies or prototypes) so it favors an exploratory approach.

Principle 2: Prediction and Adaptation

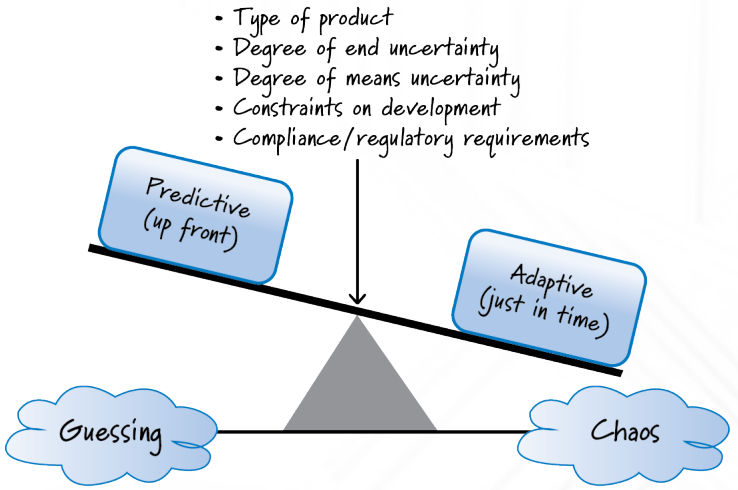
- Embrace Change in an Economically Sensible Way (What happens when there is a change)





Principle 2: Prediction and Adaptation

- Since Scrum produces artifacts in every sprint (requirements, designs and test cases) when a change occurs in each sprint there are typically far fewer artifacts, thus we embrace change in an economically sensible way.
- In Scrum, we acknowledge that we cannot get all the requirements up front, so we balance an early set of requirements and just in-time work to keep us from falling into chaos.

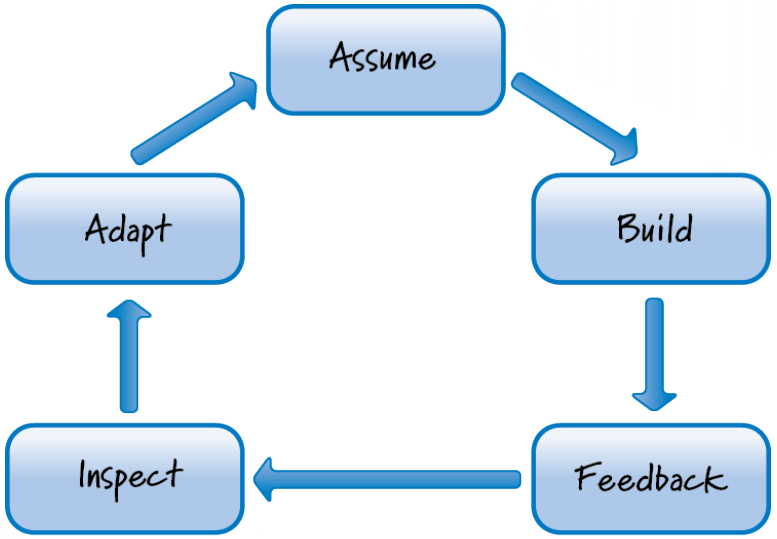


Principle 3: Validated Learning

- We need to obtain learning as fast as possible to validate assumptions.

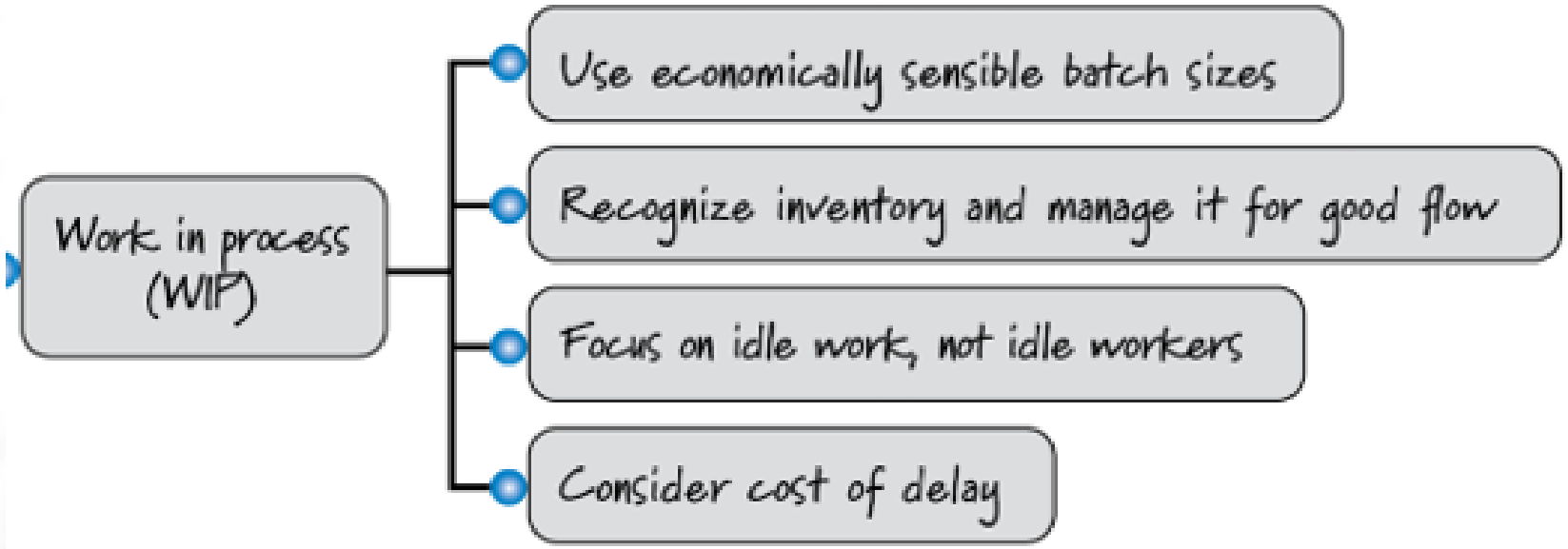
Principle 3: Validated Learning

- Assumptions constitute risk, so we have to make sure we validate assumptions to remove risk as early as possible.
- Learning loops allow a pathway for learning to be implemented back into the development flow. The daily scrum and sprint review allow for this.



Principle 4: Work in Process

- Work in Process is any work that has been started but has not finished yet.
- This is type of work should be managed, as there is a limit of much WIP can occur at the same time.
- Small batches are preferred over large batches.



Principle 4: Work in Process

Benefit	Description
Reduced cycle time	Smaller batches yield smaller amounts of work waiting to be processed, which in turn means less time waiting for the work to get done. So, we get things done faster.
Reduced flow variability	Think of a restaurant where small parties come and go. Now imagine a large tour bus unloading and the effect it has on the flow in the restaurant.
Accelerated feedback	Small batches accelerate fast feedback, making the consequences of a mistake smaller.
Reduced Risk	Small batches represent less inventory that is subject to change. Smaller batches are also less likely to fail.
Reduced Overhead	There is overheard in managing large batches – for example, maintaining a list of 3,000 work items requires more effort than a list of 30.
Increased motivation and urgency	Small batched provide focus and a sense of responsibility. It is much easier to understand the effect of delays and failure when dealing with small versus large batches.
Reduced cost and schedule growth	When we're wrong on big batches, we are wrong in a big way with respect to cost and schedule. When we do things on a small scale, we wont be wrong by much.

Principle 4: Work in Process

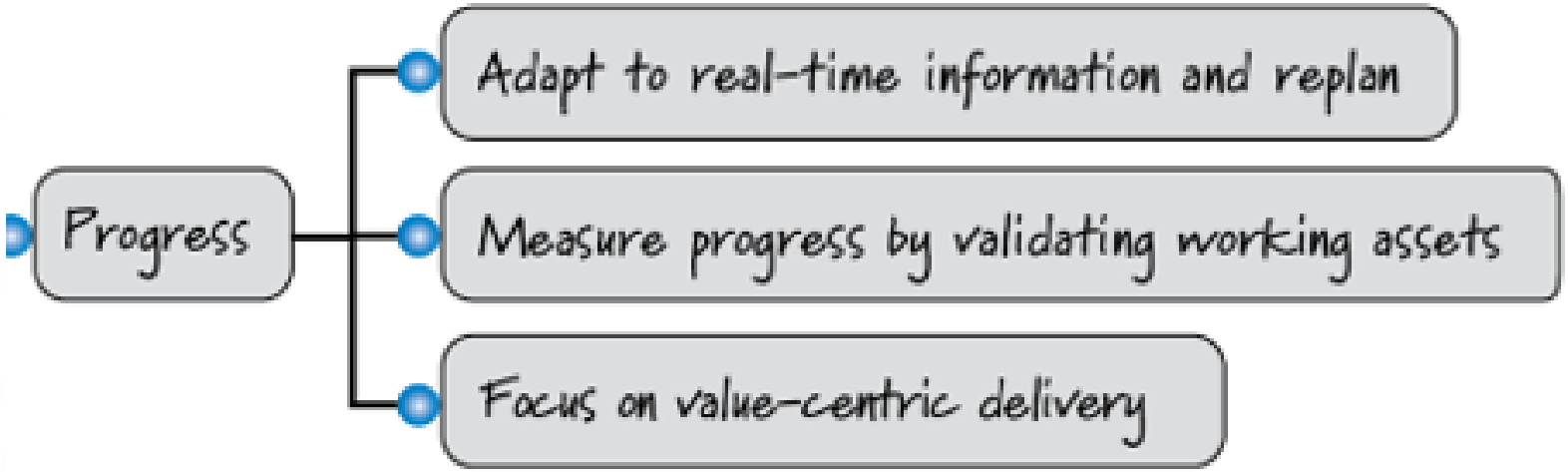
- Focus on Idle Work, not Idle Workers
- As a matter of fact 100% utilization gets us into trouble because we don't have the bandwidth to deal with urgencies when they do happen.
- Having a resource fully utilized over multiple tasks slows them down on each task. Context switching between tasks has an associated overhead costs.
- In scrum, we want to focus on finding the bottlenecks of flow of work and eliminating them, instead of keeping everyone fully utilized.

Principle 4: Work in Process

- Cost of Delay is the financial cost associated with delaying work
- Focusing on Idle Workers instead of Idle Work does not help in decreasing the Cost of Delay.
- Suppose we could hire a documentation specialist for a year salary (say $70k) or we could hire him after the product has been developed and before it goes to market.
- If completing documentation takes 2 months, and each month that the product is not in market its costing us $100k, then hiring the documentation specialist at the end of development will cost us $200k in delay, versus hiring him for the whole year to have the documentation ready at dev complete.

Principle 5: Progress

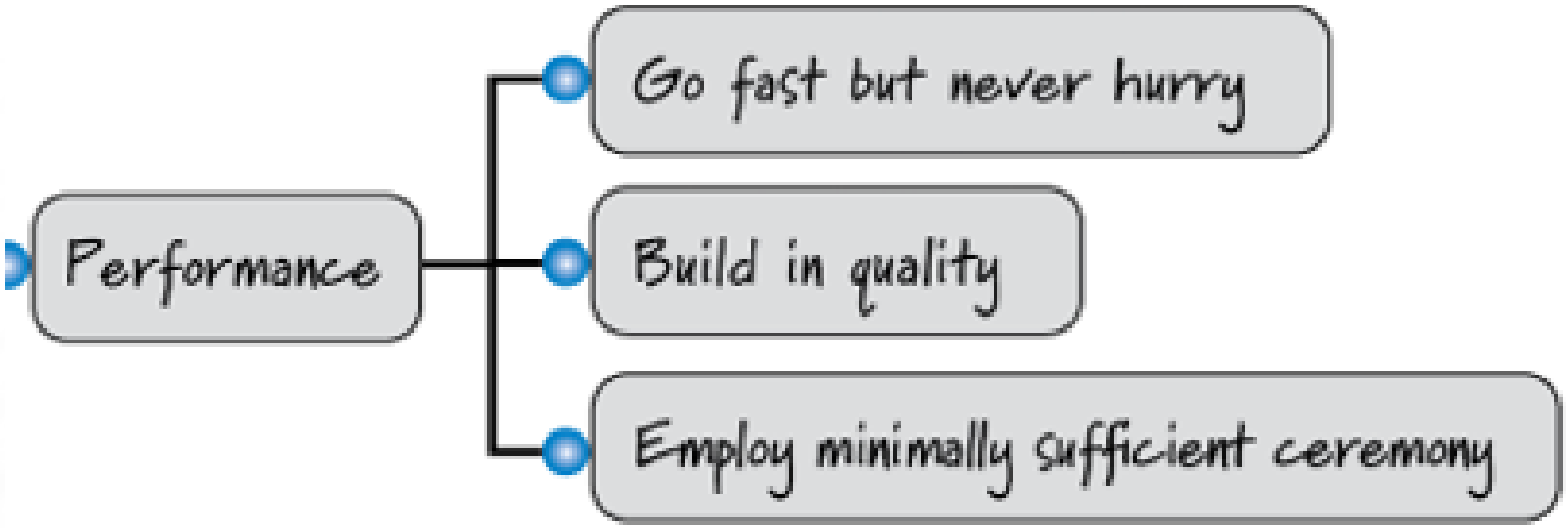
- It's important understand why scrum focuses on progress…



Principle 5: Progress

- With Plan driven development, too much focus is placed on the plan.
- In Scrum, we want to be flexible because the plan may lead us down the wrong path, so its important to acknowledge if something is not working and replan (think retrospectives).
- To measure true progress in scrum, we focus on the software that we are producing for customers and validate the assumptions of the customer along the way (think reviews).
- With plan driven development, we produce artifacts that might be considered waste for the customer (specifications, documentation, etc.) so we are not focusing on delivering the value for customers. Scrum focuses on delivering value

Principle 6: Performance



Principle 6: Performance

- Be nimble, adaptable and speedy, but make sure there is a sustainable pace. Do not sacrifice quality (very important).
- Quality is built in to the sprint. The testing is not done at the very end of product delivery. Since team members are expected to be cross functional, testing occurs throughout the sprint. We should have working, shippable product at the end.

Principle 6: Performance

- Process Formality: Scrum does not promote “doing things for the sake of doing things”
- Plan driven development usually prefers to have a very rigid process that must take place.
- Scrum is not anti-process or documentation, but allows teams to decide when items can be done informally.