Software Development Methodologies – An Overview

- Methodology – A way of completing a task.
- SO → Software Development Methodology → a way of developing software ☺
- There are a number of Software Development methodologies that operate. We will look at a few of the more widely used…

Software Development Methodologies

According to Synopsys[1], the 4 "top" methodologies are:

1. The "**Waterfall**" methodology
   - This is a traditional method of SW Development / Design

2. **Rapid Application Development**
   - Sometimes known as "rapid prototyping" or "rapid application building"

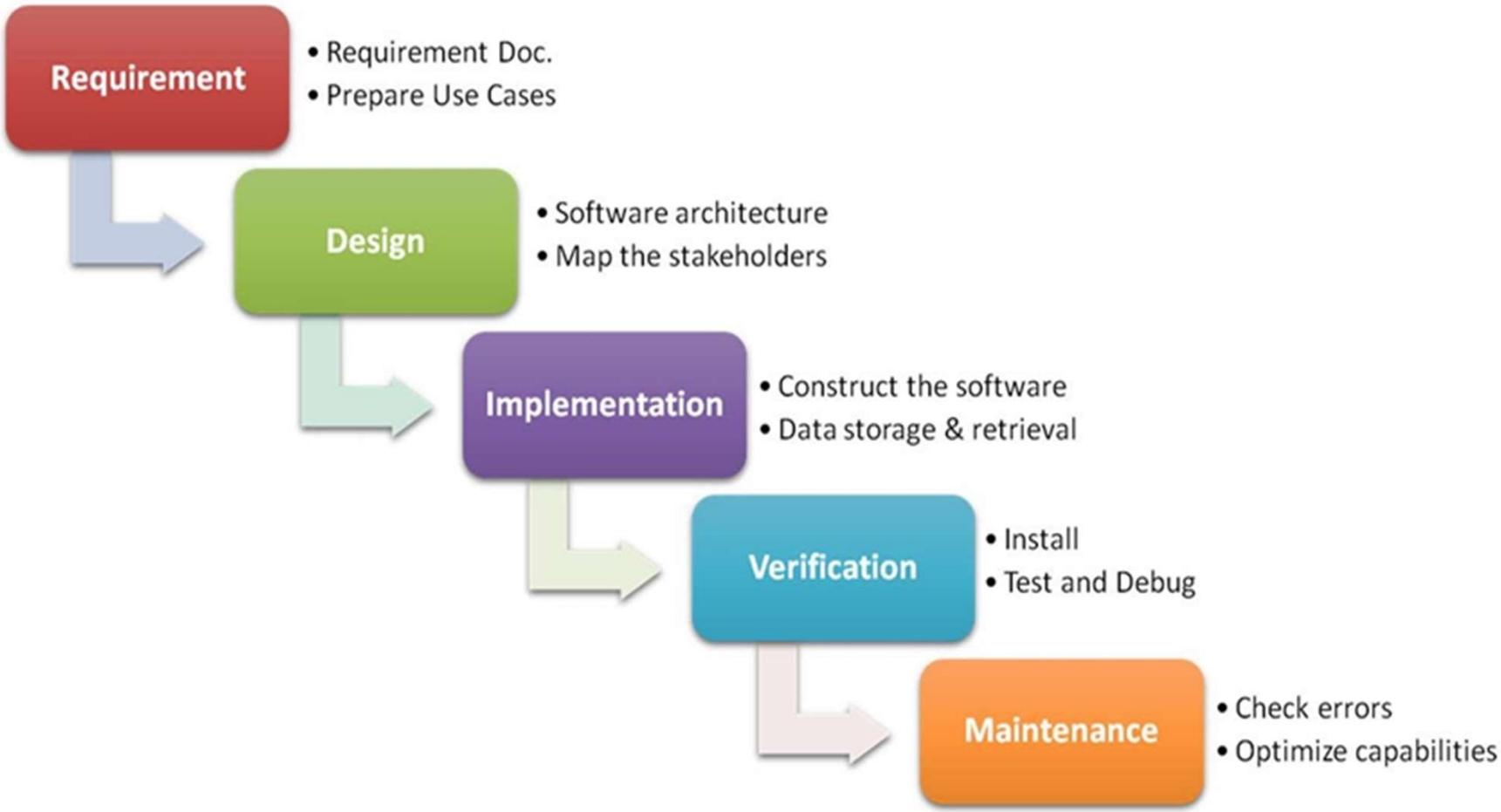
3. **Agile Development**
   - Often associated with "Scrum" technique

4. **DevOps Deployment Methodology**

[1] https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/

Waterfall Methodology

- "SSADM" from the 1970s/80s
  - Structured Systems Analysis & Design Methodology
- Although old, still represents generic approach to design/development
- Known as "waterfall" due to classic diagram describing process…


Requirement
• Requirement Doc.
• Prepare Use Cases
Design
• Software architecture
• Map the stakeholders
Implementation
• Construct the software
• Data storage & retrieval
Verification
• Install
• Test and Debug
Maintenance
• Check errors
• Optimize capabilities

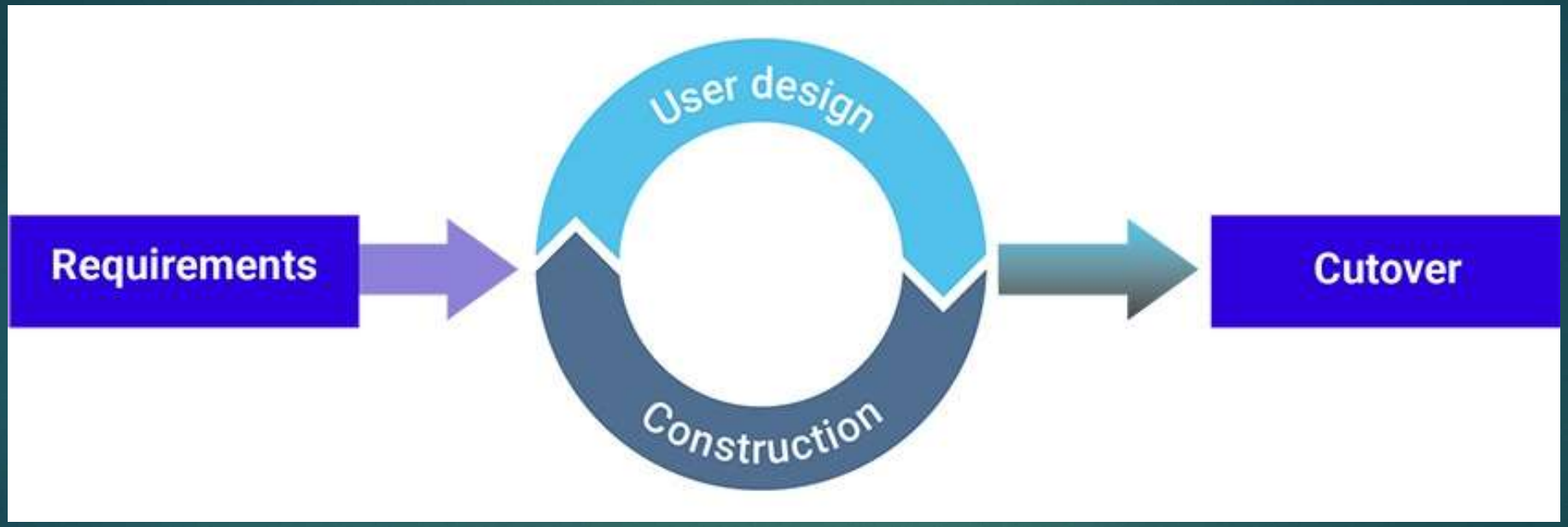
Why Waterfall ???

- The Waterfall methodology is a well-known process.
- Easily managed. Suited to well-defined development projects where requirements are clear.
- If requirements are “stable” → usually fast and widely understood
- May be suited to inexperienced teams/managers

Why not Waterfall??

- Very rigid and inflexible
- Likely to lead to failure if requirements change or were not well-defined from start
- Although well-known, not often used in "real-world" projects where flexibility is often essential
  - "No plan ever survives first contact with the enemy" ☺

Rapid App Development



https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/

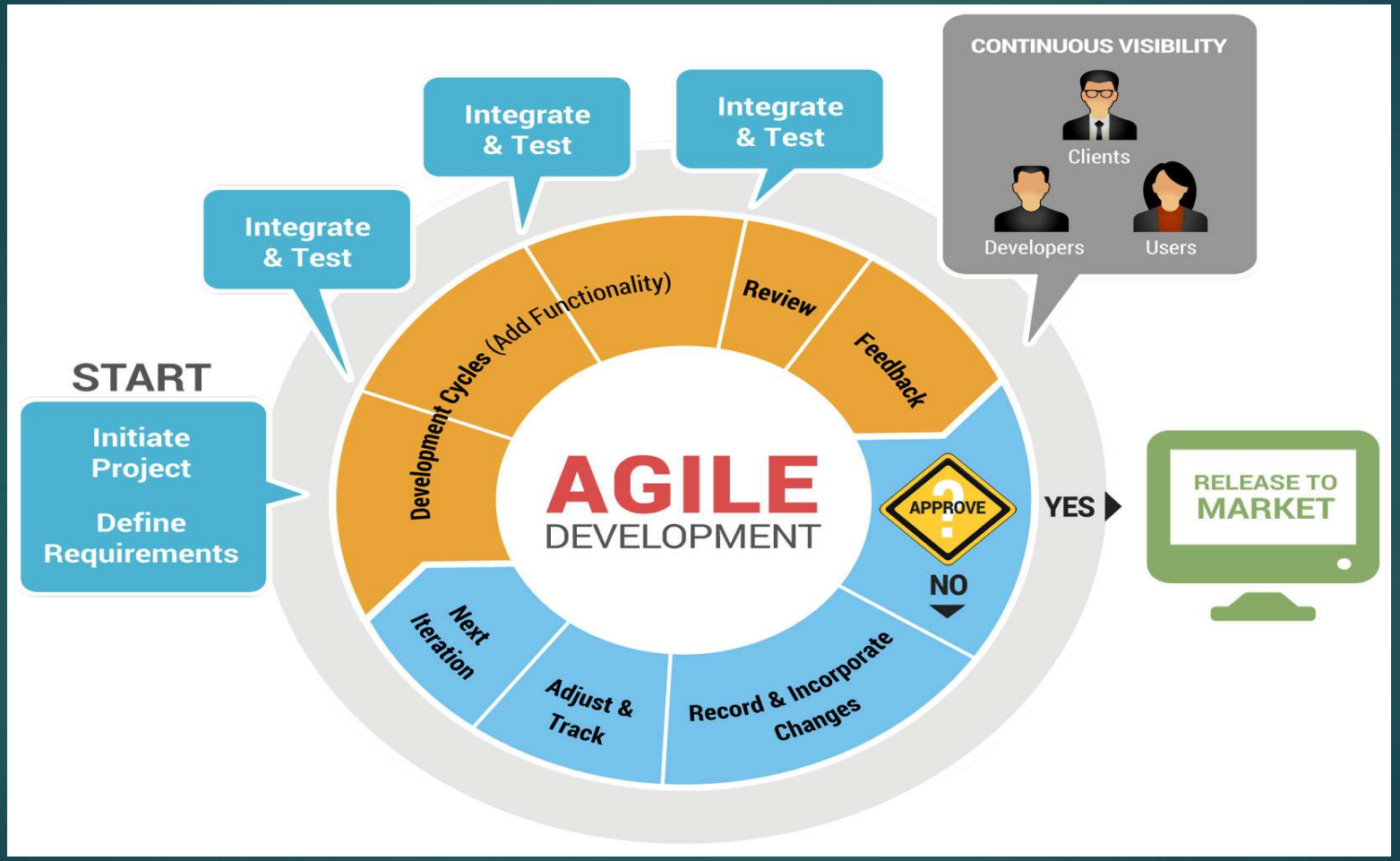
Why RAD?

- Allows Dev team to adjust / react to requirements changes
- Fast development of prototype versions → allows for better user involvement
- More likely to have satisfied user at end of process
- Fast process so better chance of being on time → once the users don't change too much!!

Why not RAD??

- Need a stable User group. If users change then requirements may change
  - Vulnerable to "scope creep" – project never actually finishes
- Can lead to a bad design due to "hack and test" approach
  - Over-focus on "fix the prototype"
- Really only works with small or medium-size development teams
- Needs team with "deep knowledge" of application area

Agile Development



https://www.meta5.us/wp-content/uploads/2014/07/z01_agile_image_02.jpg

What is Agile?

- "The main goal of agile methods is minimizing the risk by developing software in short timeboxes, called iterations, which typically last one to four weeks.

Each timebox is like a mini software project that includes all the tasks necessary to release the mini-increment of new functionality"

https://dev.to/iriskatastic/top-6-software-development-methodologies-9b

"Agile is a time boxed, iterative approach to software delivery that builds software incrementally from the start of the project, instead of trying to deliver it all at once near the end" (Agile in a Nutshell)

http://www.agilenutshell.com/

Why Agile?

- Designed to minimize risk (e.g. cost / changing requirements) by developing s/w in increments based on functionality
- Iterative approach helps find/fix problems with each iteration
- Helps correct / refine user expectations

Why not Agile??

- Needs a LOT of communication between team members
- Relies on user time-commitment that may not be possible
- Labour intensive – team must complete each "feature" within given time frame for user sign-off.
- Similar to RAD → not always efficient in large organisation