Software Maintenance & Support

Many processes include both Software Maintenance and Software Support in a single Maintenance category. This is a grave disservice from a process perspective because the two activities are dissimilar in many ways including:

1. The processes are aligned to different audiences with support focusing on application users and maintenance focusing on application sponsors
2. Support needs to scale with the user base where maintenance scales with enhancement requests
3. The optimal process is different with support generally adopting a Kanban and issue ticketing process where maintenance following a software development process like Waterfall or Agile
4. Support is optimized by supporting all products associated with a user base where maintenance is more single product focused
5. The cadence of support is minutes and hours where maintenance is days, weeks, and months which can make it challenging to balance the immediate vs the important daily activities

Software Maintenance

For our discussion we will attempt to distinguish between Software Maintenance & Software Support. Software Maintenance includes:

- Software enhancements including planning, estimating, and prioritizing
- Re-engineering or refactoring
- Testing
- Re-deployment

Software Support

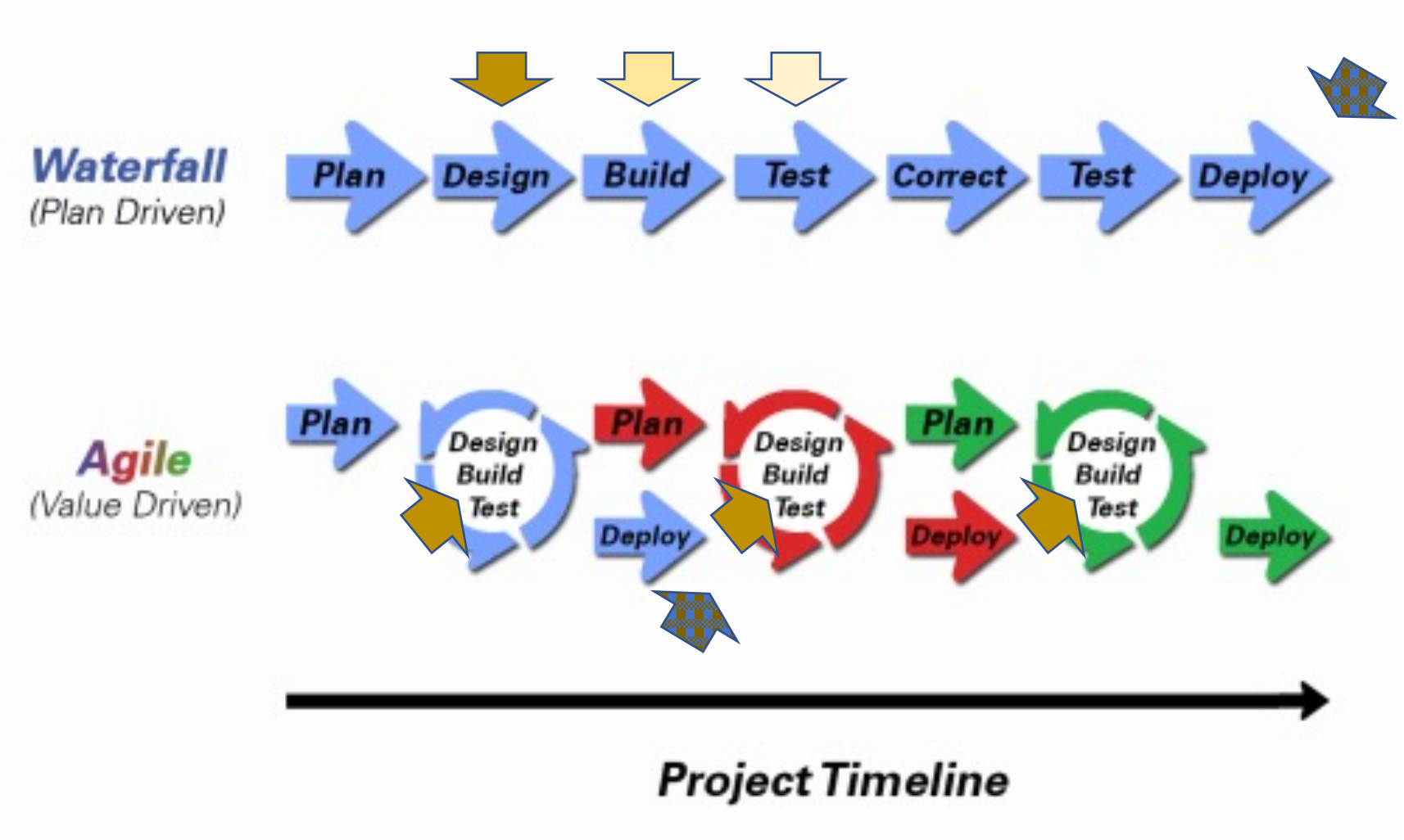
Software Support includes:

- Responding to end-user questions and issues
- Coordinating responses across applications
- Tracking & escalating issues

Waterfall vs Iterative vs Agile Maintenance & Support

	Waterfall	Iterative	Agile
References	United States Department of Defense: DOD-STD-2167A (1985)	Rational Unified Process (RUP) Open Unified Process	Scrum Kanban Scaled Agile Framework (SAFe)
Priorities	Planning and predictability	Architecture, modeling, and efficiency through early detection & fixing of issues (verification)	Responsiveness to feedback, efficiency through engineering practices, early detection & fixing of issues, and validation
Principles	Execute phases sequentially: 1. Requirements 2. Analysis 3. Design 4. Coding 5. Testing 6. and Operations Define and commit to Scope, Cost, and Timeline "early" Implement strict Change Control	Develop and test iteratively Manage requirements Use components Model visually Verify quality Control changes	Develop, test, deploy, and release iteratively Capture lightweight near term requirements Empower teams Allow requirements to evolve but maintain fixed timelines Apply engineering practices and systems thinking (e.g. TDD) Integrate early user feedback into remaining plan Maintain a collaborative approach between all stakeholders

Maintenance & Support in Waterfall and Agile



Maintenance & Support – Waterfall

When working in a Waterfall SDLC Maintenance generally:

- Occurs only at the end of the project when it is deployed to production
- Is managed, staffed, and funded by a separate project, organization, and process
- Separated into multiple tiers often including customer support (tier-1), escalated support (tier-2), and application coding support (tier-3)
- Significant ongoing enhancements are often managed, staffed, and funded separately

Maintenance & Support – Iterative (RUP)

When working in an Iterative (RUP) SDLC Maintenance & Support are generally very similar to Waterfall.

Maintenance & Support – Agile

When working in a Agile SDLC Maintenance generally:

- Occurs after the first release
- Is initially managed, staffed, and funded by the same project, organization, and process
- Separated into multiple tiers if/when Support becomes too large for the team
- In full production often includes customer support (tier-1), escalated support (tier-2), and application coding support (tier-3)
- The size of the Agile team is scaled to handle support (tier-1), escalated support (tier-2), and ongoing enhancements are often managed, staffed, and funded separately

Waterfall vs Iterative vs Agile Maintenance & Support

	Waterfall	Iterative	Agile
References	United States Department of Defense: DOD-STD-2167A (1985)	Rational Unified Process (RUP) Open Unified Process	Scrum Kanban Scaled Agile Framework (SAFe)
Priorities	Planning and predictability	Architecture, modeling, and efficiency through early detection & fixing of issues (verification)	Responsiveness to feedback, efficiency through engineering practices, early detection & fixing of issues, and validation
Principles	Execute phases sequentially: 1. Requirements 2. Analysis 3. Design 4. Coding 5. Testing 6. and Operations Define and commit to Scope, Cost, and Timeline "early" Implement strict Change Control	Develop and test iteratively Manage requirements Use components Model visually Verify quality Control changes	Develop, test, deploy, and release iteratively Capture lightweight near term requirements Empower teams Allow requirements to evolve but maintain fixed timelines Apply engineering practices and systems thinking (e.g. TDD) Integrate early user feedback into remaining plan Maintain a collaborative approach between all stakeholders

Maintenance & Support in Waterfall and Agile