

Lifecycle Models

CS183 – Hypermedia and the Web

Pure Waterfall

Document driven process

Review held at the end of each phase to determine if you progress to the following phase

Works well for product cycles where there is:

- stable product definition
- working with well-understood methodologies

Con: slow, can be hard to fully define requirements up front

variant: Sashimi Model (Waterfall with overlapping phases – allows project work to proceed before entire prior phase is complete)

Pro: more efficient

Con: milestones are more ambiguous, harder to track progress accurately

Code and Fix

Informal model: start with general idea of what to do, then start coding

Advantage: low overhead, shows signs of progress immediately

Disadvantage: doesn't scale to large projects, hard to ensure you develop code that meets requirements

Evolutionary Prototyping

Develop a prototype, and then iterate on it until customer and developer agree it's "good enough."

Pro: good when requirements are changing, or not well understood, or if the optimal architecture or design is not well understood

Con: impossible to predict how long the project will take up front (but mitigated by customers being able to see steady signs of progress)

Design to Schedule

Staged release, but with a fixed period of time, so the final stages may never be achieved.

Advantage: ensures that by a ship date, you've implemented the most important desired features