David Lambertson

**Waterfall by Royce Summary**

**Important Points from Article**

* “There are two essential steps common to all computer program developments, regardless of size or complexity. There is first an analysis step, followed second by a coding step as depicted in Figure 1. “ [328]
* “A more grandiose approach to software development is illustrated in Figure 2. The analysis and coding steps are still in the picture, but they are preceded by two levels of requirements analysis, are separated by a program design step, and followed by a testing step.” [328]
* “What we have is an effective fallback position that tends to maximize the extent of early work that is salvageable and preserved.” [328]
* “One cannot, of course, produce software without these steps, but generally these phases are managed with relative ease and have little impact on requirements, design, and testing.” [329]
* “As the analysis proceeds in the succeeding phase the program designer must impose on the analyst the storage, timing, and operational constraints in such a way that he senses the consequences. When he justifiably requires more of this kind of resource in order to implement his equations it must be simultaneously snatched from his analyst compatriots.” [330]
* “How is this procedure implemented? The following steps are required. 1) Begin the design process with program designers, not analysts or programmers. 2) Design, define and allocate the data processing modes even at the risk of being wrong… 3) Write an overview document that is understandable, informative and current.” [331]
* “The first rule of managing software development is ruthless enforcement of documentation requirements.” [332]
* “Until coding begins these three nouns (documentation, specification, design) denote a single thing. If the documentation is bad the design is bad. If the documentation does not yet exist there is as yet no design, only people thinking and talking about the design which is of some value, but not much.” [332]
* “A verbal record is too intangible to provide an adequate basis for an interface or management decision.” [332]
* “During the testing phase, with good documentation the manager can concentrate personnel on the mistakes in the program.” [332]
* “During the operational phase, with good documentation the manager can use operation-oriented personnel to operate the program and to do a better job, cheaper.” [332]
* Following initial operations, when system improvements are in order, good documentation permits effective redesign, updating, and retrofitting in the field.” [332]
* “With the simulation he can at least perform experimental tests of some key hypotheses and scope down what remains for human judgment, which in the area of computer program design (as in the estimation of takeoff gross weight, costs to complete, or the daily double) is invariably and seriously optimistic.” [334]
* “Without question the biggest user of project resources, whether it be manpower, computer time, or management judgment, is the test phase. It is the phase of greatest risk in terms of dollars and schedule. It occurs at the latest point in the schedule when backup alternatives are least available, if at all.” [335]

**Things I Didn't Agree With**

**“**If in the execution of their difficult and complex work the analysts have made a mistake, the correction is invariably implemented by a minor change in the code with no disruptive feedback into the other development bases” [329]

Royce wrote that “minor change in the code with no disruptive feedback…” I don’t agree with this statement at all. I think even with minor changes, there is always some sort of disruption in the development process. Even if it’s a single word in the code that needs changing, it requires the developer to stop what they are working on and focus on the fix they need to implement. It defocuses the developer from what they had been working on.

**Things I Did Not Understand**

I understood the article in its entirety.