Managing the Development of Software Systems

In August 1970, Dr. Winston W. Royce published his paper *Managing the Development of Software Systems*  to show what he believes is the ultimate way to organize a large scale software development program. He states, “I have become prejudiced by my experiences and I am going to relate some of these prejudices in this presentation” to establish his prior experiences with handling software development through most stages, from beginning to end. Thus, he wants to make sure that by writing this paper, he can convince other developmental teams to follow his method.

One method in this paper addresses the idea of documentation. Royce believes “management of software is simply impossible without a very high degree of documentation”. His emphasis on a “high degree” of documentation in contrast to just documentation means that this step is crucial to carry out the entire project efficiently. Royce believes that it is crucial because the documentation holds personnel accountable for completing tasks on time since it “forces the designer to provide tangible evidence of completion”. Otherwise, if there is no documentation it is easy for just the personnel to say he is “close” to finishing. Royce adds on to the importance of documentation by stating that it is more than just a paper, it is the specification and the design – thus if there is no documentation, there is nothing. He states this because without a documentation, the only thing that exists is the “idea” of the project, and thus is just a verbal exchange of the idea between people, which can only hold so much weight. Thus, with a document, there is tangible evidence of progress being made.

The Apollo computers article is an example of a heavily documented mission. The links in the article led to a 30-page instruction manual on just how to operate the AGC< which was supposed to be “more basic than modern toasters”. More examples like the AGC program was also viewable as well. Therefore, although this one article doesn’t fully prove that documentation works, a properly documented project / mission has shown great success in the past

Although Royce was correct on documentation, his ideas are not always in-line with commonly accepted practices. Such discrepancies exist in his inconsistent approach to the agile method. First, the agile approach is about members contributing together and consistently. However, Royce believes in making the document and then making sure everyone follows that document. There is no open interpretation of the document because it is supposed to be highly detailed – thus no contribution from members will occur as everyone is following the document. This also leads into the next agile idea of being flexible to changing environments because everything is already detailed in the document – any environmental changes or errors that come up already has been planned and accounted for, and if not, there is already an alternative solution. Thus, it is not as much “adapting” to the environment and more so of “accounting” for possible changes.

Another slight deviation Royce takes is in his approach regarding the waterfall model. When discussing the waterfall model, the main points are to make sure that it is a sequential design approach, with less iterative and flexible approaches – thus resulting in a one direction flow from “conception to construction to deployment”. In his final mode, the “one direction flow” is subtle but visible. If we break these steps down, we see that it goes from system requirements to program design and onwards to operations. However, between each of these steps exists iterative approaches such as running simulations and then going back to the analysis stage and many other small deviations along the way from conception to deployment. Thus, his article loosely follows the Waterfall model since there is a general one-direction path of progress, but with many deviations. Furthermore in his waterfall method, Royce addresses most steps, but does not cover the analysis stage at all and just refers to it but does give several hints as to what should be accomplished there. For instance, when he talks about the preliminary program design stage preceding the analysis stage, he states, “the preliminary program design may be substantially in error as compared to his design if he were to wait until the analysis was complete”. Here he considers the worries of having a preliminary design before an analysis, therefore the analysis must analyze factors such as storage, timing, operational constraints and other factors that contribute to the preliminary design in terms of supply, costs, and efficiency. Also, he talks about this “worry” being trivial because if all the analysts and program designers came together in the preliminary design stage, it would eventually “culminate in the proper allocation of execution time and storage resources”.

.