Managing the Development of Large Software Systems

* For a simple program, all you need is analysis and coding.
* For large programs, however, there are many more steps that requires more cost - which customer and development personnel would rather not deal with it.
* Figure 3 displays having iterative relationships between successive and preceding steps. This creates an effective fallback position. Thus it helps to maximize the amount of work that can be salvageable and preserved in the case of design difficulties.
  1. He believes in this concept, but he says it’s risky and is prone to errors.
     + Example: “The testing phase which occurs at the end of the development cycle is the first event where problems are experienced (in contrast to analyzed)..the development process has returned to the origin.”
* Nonetheless, he believes the approach is sound, and that he wants to add five additional features to eliminate further risks.
  1. Program Design Comes First
     + What?
       - Preliminary design stage: Program designer sets operational constraints to assure software won’t fail because of storage, timing, and data flux reasons.
       - If all analysts and program designers worked on this aspect, it will culminate in the proper allocation of execution time and storage resources.
       - If anything goes wrong, can go back to the requirements stage.
     + How?
       - Begin with program designers, allocate resources, write overview documents
  2. Document the Design
     + LOTS OF DOCMENTATION : “Management of software is simply impossible without a very high degree of documentation.
       - Holds designers accountable and provide tangible evidence of completion.
       - Document = specification = design in early phases.
         * It’s monetary comes out during these phases:
         * Testing: Concentrate personnel on the mistake, not just one person.
         * Operational phase: Hire personnel to operate effectively and cheaper.
         * After initial operations: If fix needs to be made, there will be effective redesign, updating, and retrofitting.
  3. Do it twice
     + Run the entire process “in miniature, to a time scale that is relatively small with respect to the overall effort.”
       - Allows questions of timing, storage, etc., which would otherwise be a matter of judgment – are now studied with precision.
  4. Plan, Control, and Monitor Testing
     + No matter how well we do the first 3 steps, there will always be unforeseen problems.
     + Interesting how he says to not use computers to test obvious errors.
       - Basically, use test specialists, manually check common errors, test logic path in program, and consider when to finally checkout the program.
  5. Involve the customer
     + Make sure customer is involved in a formal way and that the contractor and customer agree on the concepts.
     + Customer can provide insight, judgment, and commitment.

Apollo Moon Landing Computing Technology

Questions to Address

**When was the paper published?**

* August 1970

**Publication**

* Paper

**The paper is motivated by the author’s experience with what kind of systems?**

* “I have had various assignments..mostly concerned with the development of software packages… In these assignments, I have experienced different degrees of success with respect to arriving at an operational state, on time, and within costs.. I have become prejudiced by my experiences and I am going to relate some of these prejudices in this presentation”

**The author calls for extensive documentation for what reasons?**

1. Document the Design
   * + LOTS OF DOCMENTATION : “Management of software is simply impossible without a very high degree of documentation.
       - Holds designers accountable and provide tangible evidence of completion.
       - Document = specification = design in early phases.
         * It’s monetary comes out during these phases:
         * Testing: Concentrate personnel on the mistake, not just one person.
         * Operational phase: Hire personnel to operate effectively and cheaper.
         * After initial operations: If fix needs to be made, there will be effective redesign, updating, and retrofitting.

**Does the author advocate the waterfall model?**

* Waterfall Model:
  + “Sequential design approach”
  + “less iterative, and flexible approaches.”
  + “Flows in largely one direction through the phases from conception to construction to deployment”
* Royce uses a slightly modified version of the model because if problems persist, or occurs, a previous step has to be visited. However, for the most part, his model makes sure a preceding step is verified and completed before moving on to the next, never coming back unless an error occurs.

**What does the author mean by Analysis?**

* **“**[Preliminary program design] may be substantially in error as compared to his design if he were to wait until the analysis was complete”.
* **“**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”
* “In this way, all the analysts and all the program designers will contribute to a meaningful design process ( \* this refers to preliminary program design \* which will culminate in the proper allocation of execution time and storage resources)”

**Why does the author suggest Preliminary Design should precede Analysis?**

* If all analysts and program designers worked on this aspect, it will culminate in the proper allocation of execution time and storage resources.

**Is the author’s approach consistent or inconsistent with the agile approach? Why?**

**Was the Apollo Computers article helpful in understand the paper?**