***Build a little, fix a little (the Integration Thread idea)***

*The world to which we apply computers is complex enough. Deriving a software solution to problems based in the complex real world is more complicated still. As software engineers, we must manage the complexity of the design process as well. Historically, we have relied mostly on the waterfall method: building large systems in a single large effort, throwing the switch, and hoping that all of the interactions, data boundaries, time slices, and pointers to memory work as planned. This "big bang" approach to design and implementation depends on knowing and building everything up front. Even with careful analysis and documentation, this is rarely how complex systems evolve... a design team's understanding of the problem naturally gets better through fielding and evaluation of better and better attempts. Otherwise, the concept of software versions might never be needed.*

*An alternative might be to derive a simple but representative subset of the system, and to build that in standalone form, to serve as a foundation to which the remainder of the system can be added. It reduces the complex problem to a small start. It allows a deployment and evaluation of the software as a tool for specification of system requirements without wasted or architecturally weak efforts, as is often done in rapid prototyping.*

*That subset must be stable and flexible. It must be representative of the end system, therefore it must include input, output, and calculation components (that is why it is often referred to as an "integration thread", because it is a single strand of reduced capability that extends through the entire system). It must be able to be augmented without weakening. Once stabilized, it must be added-to in a massively parallel fashion. The project team should be able to pick the add-ons and simultaneously begin working on as many of them as the schedule calls for. Over the course of a project schedule, the integration thread is always stable and working, with add-ons providing increased functionality. Errors are easy to find because they are likely due to the piece just added. The system evolves, but always works.*

Example (Integration Thread in **red**):

