Growing Great Software Designers

Saturn 2018 Position Paper
Randy Ynchausti
FamilySearch.org

Borrowing from Len Bass, Paul Clements, and Rick Kazman, *Software Architecture in Practice, Third Edition*, and incorporating my own liberties, "A software [designer] is not an artiste, whose creativity and freedom are paramount. [Software design] is about discipline, and discipline comes in part by restricting the vocabulary of alternatives to [provable] solutions." Notice that this statement is more about how talented software designers approach their work, rather than how to encourage and foster their development. It represents the result or end game of growing great software designers. Starting with the end in mind is always a good idea.

The statement in the previous paragraph is comforting in that it implies that anyone with a desire that motivates them to appropriate action can become a great software designer over time. The follow-on questions to that statement are, "What actions lead to becoming a high-caliber software designer?", and "What actions shorten the time it takes to become great at software design?"

The following sections outline components of the answers to these interesting questions by providing principles with which great software designers identify and rigorously apply and practice to create their designs.

Embrace the System's Functional Requirements and Quality Attributes

A great software designer must always understand the key functional requirements and quality attributes of a system to produce a great design. The author Lewis Carroll in the book *Alice in Wonderland* wrote a short dialog exchange between the Cat and Alice that illustrates the need to understand the goals and objectives one has.

"Alice: Which way should I go?

Cat: That depends on where you are going.

Alice: I don't know.

Cat: Then it doesn't matter which way you go."

The motivation to perform software design is to satisfy the functional requirements and quality attributes of the system. The greatness of a design is based on the degree to which the functional requirements and quality attributes are simultaneously satisfied based on their priority order.

Develop a Detailed Understanding of System Components' Cause and Effect Relationships

A great software designer produces an elegant software design by understanding the components of the solution and how they will interact, which can be described by their key cause and effect relationships. From that perspective, a great software designer sees the system as a machine whose operating parameters can be adjusted to ensure it functions efficiently and is fit for purpose.

Great software designers can describe and weigh the first-order, second-order, and even third-order consequences that result from the system components in their design. Being able to reason about a system at this level of detail distinguishes great software designers from the pack.

Designs Must Acknowledge and Deal with Reality

Great software designers are humble and acknowledge issues with their designs about which they know well, and even about some that they do not know. They are radically open-minded and transparent, which allows them to seek input and counsel from believable and knowledgeable sources regarding design aspects about which they have an understanding gap.

In addition, sharing and collaboration skills are extremely important when designing software for non-trivial systems since they are often created by more than one person. Even if the software designer is the only contributor, they demonstrate their driven desire to make sure their designs are fit based on the review and analysis from other knowledgeable people.

Understand and Leverage Design Patterns

Design patterns are reusable solutions to common or reoccurring software design problems. Software designers leverage design patterns as if they are parameterized solutions that can be customized and tailored to the problem.

Master Effective Behavioral and Communication Styles

Software designers do not generally learn about behavioral styles and communnication styles as part of their formal education. For those of whom these styles are not intuitive, they can take decades to learn about them and apply them in ways that optimize their behavior and communication about a software design. However, great software designers appreciate the impact that this has on producing a software design that others support and implement.

Use a Recipe for Designing and Documentation

Great software designers use a recipe or check list of the activities that guide them through creation of a high-quality design with the appropriate amount of documentation. They realize that it is easy to forget to deal with important aspects of the design. Having a recipe ensures that they will deal with details about all important aspects of the design they are developing.

They know that the design document contains a key set of artifacts that will be used through the implementation, operation, and extension and modification of the system over time.

Experience

Great software designers understand their greatest tool comes in the form of their experience. Through the act of designing software, great software designers gain valuable insight, knowledge and understanding about the consequences of their decisions. The accumulation of strong software design experience is a vital component to their effectiveness in their work.