SDEVEN Software Development & Engineering Methodology

Version: 7.0.5
Release date: 230626

Design approaches and their review and change (SDEVEN.65-DEREV)

Table of Content

Preliminaries

This SDEVEN section presents some common design approaches (also used as review objective) practiced in company. The existing software engineering theory behind these approach is supposed to be known and understood by team members with role in software engineering (`sen` role explained in "Administrative policies (SDEVEN.10-ADM)" section "Development and research").

> ⚠ **Review process and design approach**
>
> The review process is strongly correlated with design approach followed by project. This allows allows *compliance with standards* to be verified and to make "good, reliable and usable" recommendations after a code review.

Audience

Targeted audience is:

- designers, architects - to practice the ideas and guidelines of here
- software engineers, technical leaders, product managers - to understand the design issues and impact on development

Introductory and Approach Models

In SDEVEN there are 2 (two) kind of classic and traditional paradigms used for software development:

- *waterfall paradigm* (aka sequencing) - design is a step that must be finished before development
- *incremental paradigm* (aka evolutionary) - design and development are made continuously, in small steps with returns from one to the other step

SDEVEN approach

Both paradigms are "good" and have their "pros and cons" for each project depending of its context. For that reason, SDEVEN methodology combine both approaches and apply *Agile* principles seeking to obtain **best results** by doing that.

Basics of SDEVEN approach

As already said, in SDEVEN both paradigms are used and combined, and when make ***a mix of them*** inherently a predominant one will result. This predominant one is mainly dependent of nature of product and intended features to be developed.

The **Product Manager** is responsible to decide what approach to be used in a particular situation. Here are listed the basic and *minimum* recommendations:

- if a waterfall approach is decided, then a `major` **and** `minor` version numbers should be considered **ALWAYS** and NOT only a patch version or just a build
- an incremental approach can cause version numbers to be updated in an "out-of-order way" but this thing must be managed accordingly to remain *relevant and consistent*

Agile methods

As *Agile* approach, the following methods are recommended to be used:

- SCRUM
- Extreme Programming (XP)
- Dynamic Software Development Method (DSDM)
- Feature Driven Development (FDD)

See also this Agile reference.

Type of design changes in SDEVEN approach

The following type of **design changes** must be considered related to **Design Review process**:

- **MA** *design change* - this is considered a MAJOR change and it is happening when a **completely new feature** needs to be implemented, by "*completely new*" meaning there is no background relative to that feature. A *full analysis* should be made and *some related supplementary research* could be needed. And all of these can generate collateral unexpected problems, even a design takes place.
- **CR** *design changes* - these changes appear as usually customers want some features but "staying in product scope / universe" (attn, in product scope does not necessarily mean in contract scope)
- **WIS** *design changes* - these are kind of changes, out of project scope and they could present some important features regardless they comes as customer request or an internal idea; the experience to approach these changes exists but should be placed on product ROADMAP

> ⚠ **Applying customer changes**
>
> Customer changes are normally changes that does not update the product version, but are STRICT LOCALLY for that implementation. The decision to generalize and apply them TO PUBLIC PRODUCT RELEASE is at ***product management level***.

Last update: July 18, 2023