Core Concepts: Abstraction

Mark Bools

May 30, 2021

Last Modified: 2022-02-01

Abstract

Examining the core concept of abstraction

Abstraction is, simply, extracting the interesting parts.

Okay, it's a bit more involved than that. In fact choosing good abstractions is part art, part science. Ask two developers what the best abstraction is in a particular situation and you are likely get three answers.

Models are perhaps the commonest form of abstraction. By their nature models are wrong. Any model is wrong unless you are using the model for its intended use. Consider a building's architectural model. These are intended for customers to judge a solution's aesthetics, for this purpose they are well suited. It would be stupid to use these models to assess a buildings structural properties because the abstraction is inappropriate.

You will encounter abstraction in many situations, perhaps most commonly in interfaces. Interfaces are abstractions, they hide a multitude of details. For example, a graphics library might offer a `circle(centre,radius)` function for drawing a circle onto a screen. Such an interface hides details about calculating the specific pixels to be drawn, how to deal with different hardware, how to maintain the image, indeed it says nothing about the display at all and may also support drawing to paper. The point being, as a client of this function I don't need to know these details, all I care about is that a circle is drawn somewhere (other parts of the system will be tasked with dealing with these details).

Well chosen abstractions make using and maintaining systems simpler. Poorly chosen abstractions make systems difficult to use and a nightmare to maintain.