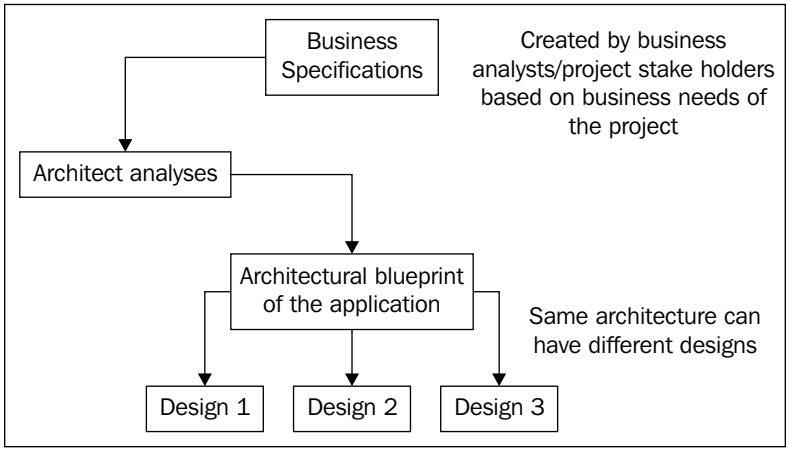
- How should the program perform error logging?
- How will the program be developed so that it is re-usable (if the architecture dictates it to be developed as a re-usable component)?

That's the part where design comes into the picture. The application architecture would define limits and boundaries within which the design would move around and improvise. So the architecture would neither go deep into the nitty-gritties of the design phase, nor would it dictate implementation guidelines and programming rules, as the architecture has no relation with programming at all. In fact, the architecture lays out specifications which are more aligned with business requirements, and makes sure that all business aspects are met and taken care of.

Coming back to our bulk email program, the term software design can be loosely translated into the process of designing the actual program, which involves using specific programming techniques (or design patterns, which we will study later) and laying out the basic solution framework. All coding would actually occur within that framework. We can have multiple design options for the same architectural specification, and it is up to the stakeholders to decide which one to go for, considering the overall efficiency and budget constraints.

Here is a simple diagram illustrating the basic process:



# Architectural Styles

With time, some of the famous and widely used approaches and techniques among the architects have been grouped together into architectural styles. A particular architectural style represents the interaction and behavior pattern between the system and its components, along with a particular layout and structure. Some famous architectural styles are:

- n-tier model
- Windows DNA
- Data-centric
- Service Oriented Architecture
- Plug-in system

There are many more styles, and each style can be customized to suit individual project needs. We will learn more about some of these styles in the coming chapters, along with some practical examples. It is very important to understand the concept, approach, and effective implementation of a style so that we can decide when to use which style in our own applications. One can even create a new style by combining any of the existing styles and customizing it to achieve greater efficiency and adaptability.

# Architecture and Design in ASP.NET

> *But, as we look to the horizon of a decade hence, we see no silver bullet. There is no single development, in either technology or in management technique that by itself promises even one order-of-magnitude improvement in productivity, in reliability, in simplicity.*

The above quote (taken from *No Silver Bullet – essence and accident in software Engineering*, Brooks, F. P.) aptly highlights the fact that technological improvements can only be stepping stones instead of being silver bullets to solve all architectural and design problems in one go. The ASP.NET platform has rapidly gained a foothold in the web development industry. One of the major factors in favor of ASP.NET, compared to JAVA or PHP, is the excellent integration of the Microsoft IDE, Visual Studio, with the framework. The VS IDE has evolved, complementing the framework itself, with time-saving features such as detailed intelligence support, debugging assistant, and code complete, to list a few. Also, Microsoft has been aggressively adding different tools and technologies, enhancing the overall developer experience. AJAX, LINQ, WCF, WWF and SilverLight have not only stirred up the development world but have also left many developers confused and wondering as to how good these new technologies are, and how they can maximize their productivity by using them.