**What to put in an SAD?**

There are a whole bunch of SAD templates on the internet, such as the template offered by RUP. However the following items seem to be necessary for each architecture document:

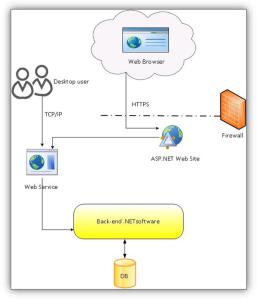
* Introduction. This can include Purpose, Glossary, Background of the project, Assumptions, References etc. I personally suggest that you explain that what kind of methodology you are following? This will avoid lots of debates, I promise!

It is very important to clear the scope of the document. Without a clear scope not only you will never know that when you are finished, you won’t be able to convince the stakeholder that the architecture is comprehensive enough and addresses all their needs.

* Architectural goals and constraints: This can include the goals, as well as your business and architectural visions. Also explain the constraints (e.g. if the business has decided to develop the software system with Microsoft .NET, it is a constraint). I would suggest that you mention the components (or modules) of the system when you mention your architectural vision. For example say that it will include Identity Management, Reporting etc. And explain what your strategy to address them is. As this section is intended to help the business people to understand your architecture, try to include clear and well-organised diagrams.

A very important item that you want to mention is the architectural principles that you are following. This is even more important when the client organization maintains a set of architectural principles.

* + Quality of service requirements: Quality of service requirements address the quality attributes of the system, such as performance, scalability, security etc. These items must not be mentioned in a technical language and must not contain any details (e.g. the use of Microsoft Enterprise Library 5).
  + Use Case View: Views basically come from 4+1 model so if you follow a different model you might not have it. However, it is very important that you detect key scenarios (or Use Cases) and mention them in a high-level. Again, diagrams, such as Use Case Diagram, help.
  + Logical View: Logical view demonstrates the logical decomposition of the system, such as packages the build it. It will help the business people and the designers to understand the system better.
  + Process View: Use activity diagrams as well as state diagrams (if necessary) to explain the key processes of the system (e.g. the process of approving a leave request).
  + Deployment View: Deployment view demonstrates that how the system will work in a real production environment. I suggest that you put 2 types of diagrams: one (normal) human understandable diagram, such a Visio Diagram that shows the network, firewall, application server, database, etc.  Also a UML deployment diagram that demonstrates the nodes and dependencies. This will again helps the business and technical people have same understanding of the physical structure of the system.
  + Implementation View: This part is the most interesting section of the techies. I like to include the implementation options (e.g. Java and .NET) and provide a list of pros and cons for each of them. Again, technical pros and cons don’t make much sense to business people. They are mostly interested in Cost of Ownership and availability of the resources and so on.  If you suggest a technology or if it has already been selected, list the products and services that are needed on a production environment (e.g. IIS 7, SQL Server 2008).  Also it’ll be good to include a very high-level diagram of the system.

[](https://aspguy.files.wordpress.com/2011/06/overall.jpg)

Also I like to explain the architectural patterns that I’m going to use. If you are including this section in the Implementation View, explain them enough so that a business person can quite understand what that pattern is for. For instance if you are using Lazy Loading patter, explain that what problem does it solve and why you are using it.

Needless to say that you have to also decide which kind of Architecture style you are suggesting, such as 3-Tier and N-Tier, Client-Server etc. Once you have declared that, explain the components of the system (Layers, Tiers and their relationships) by diagrams.

This part also must include your implementation strategy for addressing the Quality of Service Requirements, such as how will you address scaling out.

* Data View: If the application is data centric, explain the overall solution of data management (never put a database design in this part), your backup and restore strategy as well as disaster recovery strategy.

**Be iterative**

It is suggested that the architecture (and in result the Software Architecture Document) be developed through two or more iterations. It’s impossible to build a comprehensive architecture document in one iteration as not only Architecture has an impact on the requirements, but also architecture  begins in an early stage and many of the scenarios are likely to change.

**How to prove that?**

Now that after doing lots of endeavor you have prepared your SAD, how will you prove it to the stakeholders? I assume that many of business people do not have any idea about the content and structure of an SAD and the amount of information that you must include in it.

A good approach is to prepare a presentation about the mission of the system, scope, goals, visions and your approach. Invite the stakeholders to a meeting and present the architecture to them and explain that how the architecture covers their business needs. If they are not satisfied, your architecture is very likely to be incomplete.

**References**