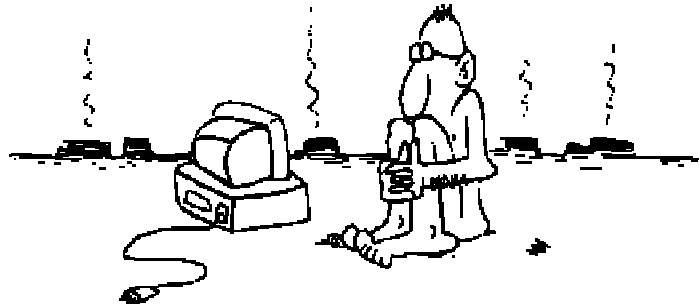


Is complexity a necessity or merely a complexity?

As I have just started documenting another technical architecture for a custom enterprise solution, I have once again managed to get my subconscious mind to resume an old worry thread of mine. The terms enterprise and architect have not found complete favour with me, or the fact that I am once again pondering over what complexity and three-lettered acronyms (TLAs) the pre-sales team and customer are expecting to find in the software development proposal.

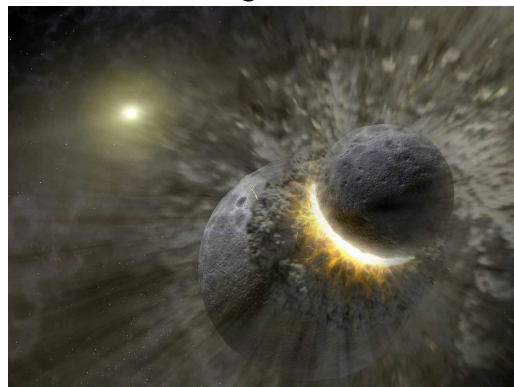


The world of Information Technology (IT) has a way of taking something very simple, like the terms enterprise and architect. While the Oxford dictionary claims that enterprise stands for “bold or difficult undertaking”, the IT industry has managed to create many types and understanding of an enterprise, an enterprise solution and even an enterprise architect. The Oxford dictionary also tells us that an architect is “someone who prepares plans for, and superintends the work of building”. In IT we once again have many types of architects, such as enterprise, solution, technical or infrastructure architects who admittedly are normally involved in the planning and designing of complex solutions. Unlike their counterparts in the civil engineering world, however, many architects in the IT enterprise world are not involved in either supervising or quality assuring solutions in progress, i.e. the “superintend” part of the Oxford dictionary.

The last paragraph has just resumed the next worry thread, namely what is the true definition, responsibility and skill/certification definition of an IT architect. As I am not a multi-processing individual, I will pause the thread again and defer the worry to the next saArchitect meeting. It is, however, important to realize that we as saArchitect must define what we as a community unanimously understand with the terms enterprise and architect.

Let us return to the original question, namely what TLAs and complexity the reader of the solution architecture document I am assembling is expecting at the end of the day. I have been involved in digital electronics and information technology since the early 80's and looking back I have come to the following conclusions many, many moons ago:

1. Core concepts hardly ever change; yet regular name changes have introduced complexity and confusion to IT, with a belief that solutions need to be redesigned.
2. Development tools and technologies are tools, not solutions. When architects delve straight into decision between Java, C# or C++ I often wonder what bridges would look like if their architects would worry about which sledgehammer to use, before drawing up the bridge blue prints.
3. Simplicity is superior over complexity and often simple concepts result in more maintainable and extensible solutions. As a developer I have often spent hours trying to understand the infamous C++ program in one statement code, or overly complex code, which once dissected turns into "oh this is what it does ... but why not just do it simpler?"
4. What we do today has been done before, so let's re-use existing building blocks, concepts and best practices.
5. Communication between all solution stakeholders is everything. In IT architects are often shielded from the rest of the organization and often placed on high pedestals, yet the business user often has the key knowledge of the enterprise and business. How often does it happen in the bridge building industry that an architect designs a bridge for pedestrians, only to realize on completion that the users are driving 100ton tanks over the construction? The comments "works as designed" or "scope creep", which could often be avoided with open communication.
6. The only rule of simplicity I break is when we assign a codename, usually space program related, to each project we initiate. An interesting codename stirs interest and communication amongst all stakeholders and often gives everyone a common entity to associate to. So if you see projects codenames Deep Space, New Horizon, Stardust or Cassini, then we are likely involved.



To cut a long story short. I have been and will continue the evangelism of simplicity, tool and technology agnostic architectures, best practices, patterns, open communication and sharing of ideas.

On this note I will defer all worry threads to the next saArchitects workshop and turn my attention back to the technical architecture documentation ... which will be straight forward and devoid of TLAs, which is probably not what they are expecting.