**What are the differences between SSADM & UML**

It may be easier to approach this question by considering both the similarities and differences. At the intentional level, the main similarity must be that both are intended to help project teams prepare a quality system specification. However, because of the different philosophies followed by each to achieve this end, the differences at the detailed level are endless.

The following notes on similarities and differences between SSADM and UML are provided as a guide to the reader who wishes to carry out a comparison exercise in detail:

1. **SSADM is a method while UML is a modelling language.**

   As a method, SSADM provides a standard set of activities which cover the analysis and outline design stages of a system development project. The activities generate products, some of which are system models. For each type of model, SSADM defines the rules and notation to be followed.

   UML, as a modelling language, provides the rules and notation for its system models but it does not provide a method framework within which to use the models.

   It is important for an organisation to recognise that if they intend using UML they will also need to select a complementary development method, or define their own.

2. **SSADM and UML use different types of system models.**

   The intention of any model is to provide a simplified view of reality and thus enable important features to be viewed with clarity. As it is not possible to cover all essential details of a system in a single model, all serious development approaches require a set of complementary models to be produced. Each model within the set will focus on a different aspect of the system but will also contain sufficient additional detail to enable cross referencing to other models.

   SSADM and UML contain roughly equivalent models with roughly equivalent intentions but there may be significant differences in their content. The following table identifies the main correlations and identifies the significant differences for the two models most commonly used in SSADM and UML:

<table>
<thead>
<tr>
<th>SSADM Model</th>
<th>UML Model</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical Data Model</td>
<td>Class Diagram</td>
<td>These models are central to SSADM and UML. The main system aspects described by each are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the types of things managed by the system</td>
</tr>
</tbody>
</table>
the attributes that describe each type of thing
the relationships between types of thing

The significant differences between the models are:

The scope of the UML Class Diagram is much broader than the scope of the SSADM Logical Data Model. In the SSADM model the types of thing being represented are Third Normal Form relations of business information; in the UML model, the types of thing being represented are Classes which include business information (although not as TNF relations) but also include things to do with the implementation of the system (e.g. toolbars, frames, etc). The SSADM model only covers detail about data; the UML model additionally associates operations and responsibilities with classes.

Function Definition
Use Case

These models are used to describe system behaviour from an external perspective. The main system aspects described by each are:

where inputs come from and outputs go to
significant conditions that affect the outcome of the processing

Although the primary means of definition for both models is a text narrative, each is supported by a set of diagrams that can be used to further define aspects of the processing.

3. Conversion of the system specification to an internal design presents different problems.

As SSADM places great emphasis on data analysis, the translation of the appropriate SSADM specification products (primarily the Logical Data Model) into a database schema (relational or network) is straightforward. The translation of processing products (primarily ECDs, EAPs and Function Definitions) into program specifications is, however, much more tricky.

With UML, if an Object-Oriented language such as Java is the target, because Classes and Operations have direct equivalents within the programming language, procedural translation is straightforward. Database design may, however, be more problematic because no formal data analysis is carried out in UML.
**What are the benefits of using SSADM?**

The main benefits of using any structured systems development method (as opposed to none) are:

- the likelihood of aspects of development being overlooked are reduced because a standard set of activities are defined,
- development staff are more interchangeable because a standard set of deliverables are defined,
- process improvement becomes possible because the method constitutes a repeatable, measurable process.

The main benefits of using SSADM (as opposed to producing your own method) are:

- the cost of developing your own method is prohibitive,
- the time taken to develop your own method would delay its implementation by months, if not years,
- many organisations do not have the necessary in-house skills to develop their own method,
- a pool of development staff already exists with experience of SSADM; if you develop your own method, you must train staff to understand it.

The benefits of using SSADM over other proprietary methods will vary depending on the attributes of the other method.

**How appropriate is SSADM to small and medium sized companies?**

It is the attributes of the development rather than the company that indicate the appropriateness of SSADM. SSADM is primarily suited to developing systems that will manage the operational processing associated with several high volume business events. As small companies are less likely to develop this type of system, they have less need for a method such as SSADM.

The reason for adopting a method such as SSADM is to reduce risk of:

- misunderstanding the requirements,
- implementing a system which cannot be maintained,
- implementing a system which does not deliver the intended benefits,
- implementing a system which is technically unsound,
- implementing a system which exceeds budget/timescales.

At the end of the day, it is not a matter of whether SSADM is appropriate for small/medium businesses, but whether it is appropriate for the system being developed.

Many large organisations have produced 'cut-down' versions of SSADM to cater for developments that require say, less than 40 man days effort. However, even on smaller projects it is important that the requirements of the system are understood and a good design is produced. Products which we would consider to be essential for SSADM developments are:

- Logical Data Model
- Requirements Catalogue
- Function Definitions

Other products can be added to this list depending on the complexity/risk of the
What are the drawbacks of using SSADM?

SSADM is prescriptive and following all the steps it may take some time before users actually see a concrete (non-paper based) deliverable. The users may therefore only pay 'lip service' to the paper-based deliverables and only realise much later that the solution offered isn't satisfactory.

There is a danger of 'paralysis by analysis' - so much time is spent over-analysing the problem that the project fails to make satisfactory headway. (several large Government projects experienced this problem).

Following a method (SSADM, UML, DSDM) doesn't make people into good analysts, all this produces is consistent rubbish, you still need competent people.

What are the benefits of using techniques like DFDs, ELH, ER diagrams, Rich Pictures, and Root Definitions?

Graphical representations are often more easily understood than long passages of textual narrative.

Key aspects of a system are brought together using these techniques and analysed in different ways to ensure consistency.

The drawback of these techniques in SSADM is that they analyse data and process separately, more recent Object Oriented approaches treat processes and data together as objects.

Any picture-based techniques must have good supporting narrative and definitions. A picture alone is dangerous, it provides little substance but gives the impression of progress.