Why do MBSE?

Models are created to deal with complexity. In doing so they allow us to understand an area of interest or concern and provide unambiguous communication amongst interested parties.

MBSE goals:
- Improved communications
  - With stakeholders
  - Within the engineering project teams
  - Across spoken language barriers
- Improved quality
  - Early identification of requirements issues
  - Enhanced system design integrity
  - Improved specification of allocated requirements to hardware and software
  - Fewer errors during integration and testing
  - More rigorous requirements traceability
  - Consistent documentation
- Increased productivity
  - Improved impact analysis of requirements changes
  - Improved interaction across a multi-discipline team
  - Reuse of existing models to support design and technology evolution
  - Auto-generation of documentation
- Reduced risk
  - Improved cost estimates
  - Early, and on-going, requirements validation and design verification

Dos and Don’ts for MBSE:

Do
- Define an approach to MBSE which applies to your particular project
- Plan and manage the modelling process
- Define the system of interest and keep the models as simple as possible
- Add extra content as required to solve a particular need
- Understand the assumptions made within the models – all models are incomplete by definition
- Verify your models as you develop them
- Question simulated results

Don’t
- Model in isolation from the rest of the design team
- Model for the sake of it
- Assume a standard MBSE methodology will simply give you answers – it still requires engineering know-how
- Simply believe what a tool vendor tells you – verify that the tool gives you what you need in your overall approach
- Model without understanding the inputs and outputs of the modelling exercise
- Use the same data to develop and test the model

…and why not?

MBSE is not the answer in isolation from other practices.
- It is hard to model non-functional requirements.
- The model can be a barrier to understanding for some stakeholders
- Effective MBSE requires a disciplined and well-trained project team and a mature process approach

Model Based systems engineering has the model, or models as the primary data source.
Model Driven Development uses the activities associated with modelling to drive the whole development process.
Defining and Implementing Effective Process

The use of standardised modelling languages, such as the Unified Modelling Language (UML) or Systems Modelling Language (SysML), is desirable as they are understood across a significant proportion of the SE community. However, since it is important to model to the level of the intended audience, it may sometimes be necessary to use an alternative notation.

Increasingly technology is moving into the virtual/conceptual world of dynamic simulation. These models can often be run in real time to give a virtual response close to the actual system, dynamically adapting to meet changes in tolerance or responses to events.

There are a multitude of modelling techniques and approaches that fall within MBSE. These include:
- Structured Analysis and Design
- Data Flow Diagramming
- State Transition Diagramming
- Behavioural Modelling
- Entity Relationship Modelling
- Finite Element Modelling
- Environment Virtualisation
- Computer Aided Design (CAD)
- Analytical Modelling
- Process Modelling

There is also a range of methodologies that support the MBSE approach. One methodology that has been supported by INCOSE specifically is Object-Oriented Systems Engineering Methodology (OOSEM). There are however many others, often produced by tool vendors; examples include the Rational Unified Process (RUP), Harmony SE, and Vitech's Model-Based Systems Engineering Methodology. It is often a good idea to start with a predefined methodology, or even a combination of methodologies, and then tailor this to suit the needs of a specific SE problem.

Modelling needs to be managed; without control or planning it is likely to result in rubbish in/rubbish out. It is important to record objectives and assumptions in a similar manner as when defining the SE approach. This is particularly true where the modelling and SE process become one and the same thing.

Fundamental Concepts and Enablers

Models can be either abstractions or representations of reality that facilitate the understanding of complexity. MBSE commonly uses multi-user repository-based modelling tools which provide an environment where a precise and unambiguous world view of the parts of the system and their behaviours and interactions can be defined and managed. This can be applied horizontally to support the SE lifecycle process and vertically from integration into the implementation disciplines.

A move to MBSE requires a cultural change and a different mindset. It is one where modelling is used to capture much of the required data.

With MBSE the system solution and design, as described in the modelling environment, are allowed to evolve, with increasing detail being added as required. Documents detailing particular design baselines are usually automatically generated.

This provides several advantages:
- Systems Engineers focus on the technicalities of the problem rather than document structure
- Diagrammatic descriptions are often less ambiguous than textual descriptions
- Greater consistency across related documents
- Dependencies are explicitly captured across stovepipes resulting in less duplication and inconsistency

Architecting processes capture elements, relationships, and attributes that are needed to describe a system architecture which forms a variety of viewpoints that address stakeholder concerns [see Z8 System Architecture]. It is the Systems Engineer's job to assimilate all these into one coherent representation, or model, of reality.

Models are commonly used to describe or capture the architecture. Equally the data contained within architectural descriptions feeds into models that aid the understanding of system structure or performance, such as simulation or a decomposition of functions.

A high level system model can be built, which allows the collaborating team to visualize the entire system and the surrounding environment. Following decomposition into sub-functions or components, models can then be used to map onto physical architectures. This allows validation of the resulting system to take place and for the customers/users to more clearly see their vision earlier than would otherwise be possible.

The models can be used as integration test benches to support Integration, Verification and Validation testing throughout the SE lifecycle.

The Modelling Language is just the language, and must be combined with a methodology to be useful.