Measuring MBSE Model Maturity – A Library-Supported Approach

Brett Morris and Derek Rogers
(formerly of) Shoal Group Pty Ltd

Abstract

This paper covers recent work undertaken at Shoal to investigate and develop maturity metrics to increase understanding of the maturity of MBSE models that are built on Shoal and Shoal client projects. Maturity metrics are an important engineering tool that can collectively serve as a leading indicator for project performance. If a project is at a lower level of maturity heading into a major milestone than required there is a good chance of cost and schedule over-run, or quality deficiency. The paper commences with a review of the open literature that identified different methods to track model maturity. The information uncovered during the literature review is then synthesized into an MBSE model maturity assessment method that includes the ability to select MBSE model maturity metrics from a library. The work undertaken to generate a metrics library is presented and the paper concludes with some observations and lessons learned, as well as plans for further work. The approach presented could be used by organisations undertaking MBSE modelling to build their own library of metrics to measure the maturity of their MBSE models as they are developed.

Keywords: Model-Based Systems Engineering, Measurement, Model Maturity.

Introduction

Surveys of Model-Based Systems Engineering (MBSE) practitioners by the INCOSE Model-Based Conceptual Design Working Group identified that “modelling without stopping criteria” was one of the key issues they encountered (Morris, Harvey, Robinson, & Cook, 2016). Furthermore, this issue was exacerbated by the use of MBSE on projects (Morris, Harvey, Robinson, & Cook, 2016), leading to a focus on model development rather than doing Systems Engineering (SE). The prevalence of this issue suggests that it is not uncommon for modelling efforts to commence without a clear statement of purpose, or goal of the model. Without a goal, tracking the progress or maturity of a model can be problematic.

Shoal regularly undertakes MBSE modelling to support client and internal projects and have typically used a combination of expert judgement and inbuilt model validation rules to assess the maturity and quality of these MBSE models. An improved ability to track and assess the maturity of MBSE models as they are developed can help ensure the needs of clients are met and would enable Shoal more accurately to plan and optimise the resources applied on projects that involve MBSE modelling. To increase the ability to track the progress of MBSE model development on projects, Shoal recently undertook a research project to investigate potential methods to measure MBSE model maturity. The research question that underpinned the research project was:

Can a method be developed to measure the maturity of an MBSE model?

This leads to the sub-questions:

- How can the maturity of an MBSE model be measured?
- What attributes of an MBSE model provide an indication of model maturity?

The measurement of an MBSE model’s maturity and progress of the modelling effort as the model is developed falls into the category of measurement processes. The INCOSE SE Handbook (INCOSE,

2023) highlights that the measurement process “will help define the types of information needed to support program management decisions and implement SE best practices to improve performance.” The ISO standard for systems and software engineering – measurement process, ISO/IEC/IEEE 15939 (ISO/IEC/IEEE, 2017) provides a useful Measurement Process Model for the presented work as shown in Figure 1.

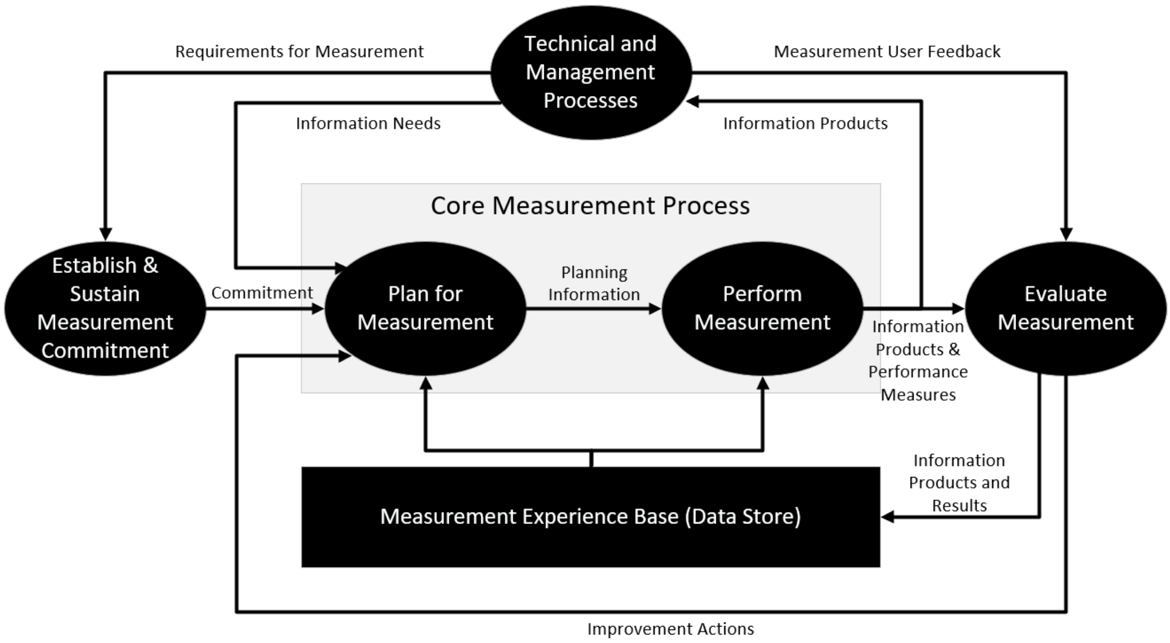

Figure 1: Measurement Process Model (ISO/IEC/IEEE, 2017).

The developers of the draft ISO/IEC/IEEE Model Based Systems and Software Engineering (MBSSE) standard, DIS 24641 (ISO/IEC/IEEE, 2021) appear to be cognizant of the issue of modelling without stopping criteria and have included “Establish MBSSE goals and measures” as an early process in the “define the scope and objectives of MBSSE” activity. However, the standard provides no guidance on how to establish the goals, or a library of generic measures that support the establishment of an MBSE model’s maturity or completeness. This is the objective of the work covered in this paper.

The following sections of this paper cover a literature review on the topic of measurement of MBSE models and other digital models used during system development. From the gaps identified in the literature review, a new method for measuring the maturity of MBSE models is proposed. The paper then focuses on work to develop a library of MBSE model maturity metrics that are consistent and integrated across the lifecycle of a project, and based on the Goal, Question, Metric (GQM) approach. The paper concludes with some lessons learned with areas for further development and follow-on work.

Literature Review

The review of the open literature identified several approaches to measuring digital model maturity with potential to be applied to measuring MBSE model maturity were identified.

Measuring Model Maturity with Checklist Criteria

Gaskell & Harrison (Gaskell & Harrison, 2019) focus on metrics to determine whether a Descriptive System Model (i.e., MBSE model) is mature enough to enter a project Mandated System Review (MSR) milestone using the MSR entrance criteria. A typical project will have: an MSR at the end of an initial requirements phase, a System Requirements Review; up to three for system definition and design phases – System Definition, Preliminary Design and Critical Design; another for entry to test, a Test Readiness Review; and two final audit reviews Functional and Physical Configuration Audits; prior to a

System Acceptance Review. The names of reviews can vary within and between industries. Gaskell & Harrisson work backwards from the MSR entry criteria checklists to generate the percentage complete and adherence to measures related to checklist requirements for a model in three main areas:

1. Relationship Mapping
2. Metamodel
3. Methodology

Interestingly, the research covered in (Gaskell & Harrison, 2019) appears to have enabled model metrics to be calculated within an MBSE modelling tool. This is a useful approach as it allows modelers to be aware of model progress at any given point in time and avoids separate manual manipulation of data.

(Gaskell & Harrison, 2019) provide a process for developing an MBSE model's maturity metrics:

1. Define modelling maturity attributes – linked to the goal of model (tied to MSR entry criteria in this work, but the attributes could conceivably be linked to any goal or model purpose).
2. Construct metric suites using the maturity attributes – the authors do this in the model in the paper using tool specific elements. Each metric attribute is scored subjectively on a 0-100 scale, with 100 indicating fully mature.
3. Capture and assess metrics – The MBSE tool uses a "Metric Table" to do this.

(Gaskell & Harrison, 2019) conducted a pilot experiment of their proposed process for measuring MBSE model maturity on an acquisition project approaching the Critical Design Review MSR. The most critical limitation they identified in the pilot experiment was that "In order to have effective modelling maturity metrics a well-defined modelling environment is absolutely critical" (Gaskell & Harrison, 2019). This means any model that is going to be measured needs a well-defined modelling environment underpinned by a robust metamodel established before trying to undertake measurements. (Gaskell & Harrison, 2019) also provide some other useful limitations/considerations in the conclusions, which included the observation that maturity metrics don't assess gaps, or the quality of the modelled elements.

Overall, the work of (Gaskell & Harrison, 2019) provides a useful example of how MBSE model maturity can be measured if it is linked to the model's goal, with the measurements performed and maintained within the model. However, the metrics (Gaskell & Harrison, 2019) used are tied to the entrance criteria for an MSR. Not all MBSE models Shoal builds are used on programs governed by linear lifecycle models with defined milestones. This leads to the assertion that a more flexible MBSE model maturity assessment method is required so that it can be applied to the broad range of projects Shoal undertakes.

Measuring Model Maturity by Volatility

(O'Brien & Smith, 1995) focus on assessing design maturity in Concurrent Engineering (CE) development environments to optimise design time and maximise profits. They state that design is typically managed and monitored by:

1. Configuration management
2. Design reviews by individual managers
3. Design reviews by design committees
4. Project management and its adaptation of total time management

5. Checklists (in a similar vein to the work of (Gaskell & Harrison, 2019) who used the MSR entrance criteria checklist)

(O'Brien & Smith, 1995) argue that Electronic Data Management tools (e.g. design version number tracking) can be used to determine "…the current maturity of designs by quantifying the size, importance and frequency of design changes throughout the design process" (O'Brien & Smith, 1995). This infers that by tracking the number and frequency of design changes (later references refer to this as design volatility) as shown in Figure 2, managers should be able to track design maturity.

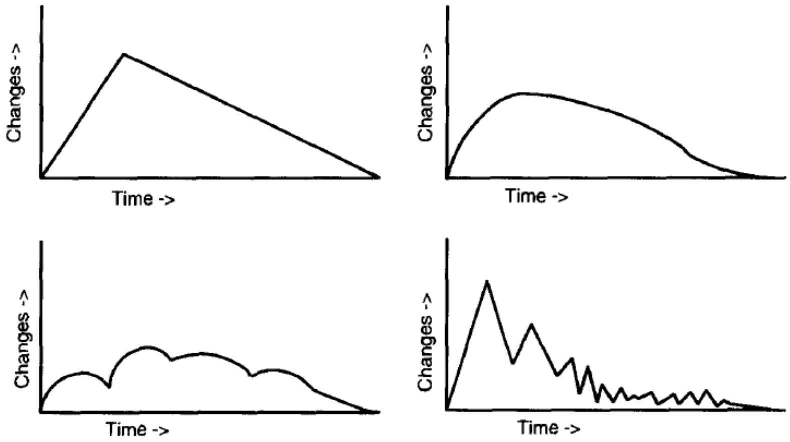

Figure 2: Design changes vs. time as an indicator of design maturity (O'Brien & Smith, 1995).

While the approach of tracking the number and frequency of design changes appears to be a useful concept, (O'Brien & Smith, 1995) provide little guidance on how or what changes to track.

To assess design maturity, (O'Brien & Smith, 1995) propose using:

- Change Frequency – track the number of version number changes in Electronic Data Management tools over time.

- Change Magnitude – the authors note that "determining the magnitudes of changes will not be so easy." This infers there would need to be a suitable data available to support an assessment of the magnitude of the design changes.

- Risk Evaluation – by making a comparison of the same Electronic Data Management files as used for change frequency to track change frequency before and after release. This can then be used to determine whether the design was actually mature before it was released.

(O'Brien & Smith, 1995) provide an indicative design change volatility profile, with possible explanatory notations for the peaks and troughs in the volatility curve as shown in Figure 3.

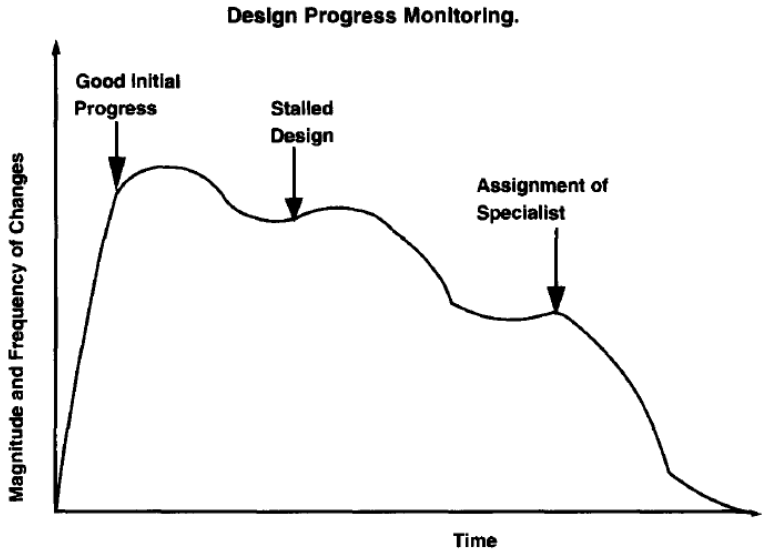

Figure 3: Design change profile vs. time with possible sources of peaks and troughs (O'Brien & Smith, 1995)

However, a maturing design may not be the only source for a reduction in design change volatility. Examples may include:

- A change in work priorities meaning that effort is expended outside of the engineering tools (e.g. a focus on documentation),
- Staff levels and availability can impact design effort (e.g. holiday periods, sickness or staff reassignment and prioritisation).
- Software access and IT issues.
- Changing user requirements that impact design effort.

Furthermore, MBSE model maturity is likely to be reflected in a range of metrics, of which model volatility in terms of the magnitude and frequency of changes is only one attribute.

Measuring Model Quality as a Surrogate for Maturity

(Giachetti, 2017) focuses on MBSE models (which he classes as “conceptual models”) and considers the question “what makes a good model?” He argues that understanding what makes a model good is important because:

1. “It is likely some models are better than others at representing a particular system aspect” (Giachetti, 2017) (pp.1).
2. “It is likely the models available to us influence how we think about systems” (Giachetti, 2017) (pp.1).
3. “Understanding the goodness of models can help tool developers to improve the tools we use in SE” (Giachetti, 2017) (pp.1).

(Giachetti, 2017) (pp.2) highlights the difficulty of measuring MBSE model goodness or completeness with:

“*Most work on assessing the goodness of models involves verification and validation (V&V) of simulation models. In this case, the criteria of what constitutes ‘good’ is clear because we want to use models to analyse and/or predict the performance of a system. The concept of model validation is not suited to the large number of systems architecture models forming part of MBSE.*”

(Giachetti, 2017) also makes an important point regarding the applicability of V&V to MBSE models that “… in these cases, we should be asking what is the purpose of the model and how well does the model serve that purpose? To answer the question, we need criteria.”

(Giachetti, 2017) highlights that some researchers focus on descriptive model validation, through logical tests of adherence to syntax and semantics. This is the approach of (Giammarco, 2010), which is covered in a later section. Giachetti highlights that: “The software engineering community has given much more attention to the quality of conceptual or descriptive models” and provides a table of descriptive model quality criteria from a range of sources as shown in Figure 4.

Paige	Friedenthal	ISO/IEC 9126
Simplicity	Purpose well-defined?	Functionality
Uniqueness	sufficient scope	Reliability
Consistency	model fidelity	Usability
Seamlessness	completeness relative to scope	Efficiency
Reversibility	well-formed	Maintainability
Scalability	internal consistency of the model	Portability
Supportability	understandability of the model	
Reliability	accurate or valid model of domain of interest	
Space economy		

Figure 4: Descriptive Model Quality Criteria from Various Sources (Giachetti, 2017)

A key challenge with the quality criteria given in Figure 4 is how to measure them in a quantitative manner with the attributes available in an MBSE model.

(Etzel, Hofhammer, & Bauer, 2020) focus on assessing the quality of system architectures (functional and logical) rather than the maturity of a model. This raises an interesting point regarding the maturity of MBSE models, since a model that measures as mature (e.g. most or all functions are allocated), may not necessarily be of good quality (e.g. doesn’t adhere to the syntax of the modelling language). Nonetheless, there appears to be some quantitative quality measures the authors identify from the software domain, that could support assessments of MBSE model maturity. Of the software quality metrics, the authors consider, the one that could potentially be useful to support the establishment of an MBSE model’s maturity is Coupling between object classes (CBO), defined as follows:

$$CBO = 1 - \frac{\#\ classes\ with\ associations}{\#\ associations} \qquad (1)$$

CBO could be useful for assessing Functional Analysis Architecture and Functional Design Architecture (Etzel, Hofhammer, & Bauer, 2020) since it would provide an indication of the progress from functional to physical design. This could be taken further to assert that the CBO (with appropriate adjustments to the equation terms) of an MBSE model can support an assessment of the maturity of the model’s functional behaviour and definition of interfaces. In a mature model it would be expected that there would be a similar number of functional flows between system elements and defined interfaces, and therefore a CBO approaching zero.

From the literature covered in this section, model quality has the potential to provide an indication of MBSE model maturity, however, the definition of “MBSE model quality” and how to measure it in an MBSE model remains a challenge.

A Model Maturity Index

(Garcia, Golparvar-Fard, de la Garza, & Fischer, 2021) investigated and developed a subjective Model Maturity Index (MMI) measurement framework for measuring the level of development in the digital models used in civil construction projects. The work provides a useful reference, but also highlights the difficulty in measuring progress in a model-driven development environment. The MMI reflects the approach of typical maturity models such as CMMI® and spans a continuum of model maturity starting from concept level to as-built facility management-enabled models (Garcia, Golparvar-Fard, de la Garza, & Fischer, 2021) (pp5). The approach is to assess the modelling disciplines separately (e.g. piping, layout, structural) and the authors write: “progress tracking is only meaningful at the levels of model discipline and/or Work Breakdown Structure (WBS) locations”. This may be overcomplicated for an MBSE model, where the modelling disciplines (e.g. functional, physical, operational, interfaces, etc.) are not particularly WBS element specific since they are all related.

If a similar maturity model approach was adopted to measure the maturity of MBSE models, then a maturity scale and descriptors suitable for a subjective assessment would need to be developed. Obtaining a consensus amongst MBSE practitioners on such a maturity scale may not be straightforward.

Model-Based Architecture Assessment

With this approach to model maturity assessment, the focus is on characterising the connectedness of an architecture model. Kristin Giammarco is a prominent author in this area. Giammarco (Giammarco, 2010) states that the objective of the research is to “mathematically express architectural elements, relationships, and some rules that can be used to develop customizable sets of criteria for some hard-to-quantify terms often used to characterize architecture models, such as ‘integrated’ and ‘complete’”. This relates to the point previously identified on the nature of a mature vs. quality MBSE model and the difficulty associated with quantifying model “goodness”.

Giammarco (Giammarco, 2010) argues that model quality and completeness requirements can be expressed in a logical expression to build checklists (“specifications”), or automated scripts for checking models. Several logical expressions could be used to build a means of measuring a subjective term like “model maturity”. Giammarco demonstrates this approach in (Giammarco K. , 2014), where a set of axioms are provided for a five-level maturity model as shown in Figure 5.

Phase 1 Model Maturity		
1.1	No action generates and receives the same input/output.	$(\forall a \in A)(\neg \exists n \in N)[generates(a,n) \wedge receives(a,n)]$
1.2	If any action generates an input/output, it also receives an input/output.	$(\forall a \in A)(\exists n_1 \in N)[generates(a,n_1) \rightarrow (\exists n_2 \in N) receives(a,n_2)]$
Phase 2 Model Maturity (Phase 1 axioms, plus the following are satisfied)		
2.1	Any input/output is generated by some action.	$(\forall n \in N)(\exists a \in A)[generatedby(n,a)]$
2.2	Any input/output is received by some action.	$(\forall n \in N)(\exists a \in A)[receivedby(n,a)]$
2.3	Any action is performed by some asset.	$(\forall a \in A)(\exists p \in P)[performedby(a,p)]$
2.4	Each action generates or receives at least one input/output.	$(\forall a \in A)(\exists n \in N)[generates(a,n) \vee receives(a,n)]$
2.5	Each asset performs at least one action.	$(\forall p \in P)(\exists a \in A)[performs(p,a)]$
Phase 3 Model Maturity (Phase 2 axioms, plus the following are satisfied)		
3.1	Each asset is connected by at least one conduit.	$(\forall p \in P)(\exists c \in C) connectedby(p,c)$
3.2	Any conduit connects to at least two disjoint assets.	$(\forall c \in C)(\exists p_1 \in P)(\exists p_2 \in P)[connectsto(c,p_1) \wedge connectsto(c,p_2) \wedge (p_1 \neq p_2)]$
3.3	Any conduit connects to no more than two assets.	$(\forall c \in C)(\neg \exists p_1 \in P)(\neg \exists p_2 \in P)(\neg \exists p_3 \in P)$ $[connectsto(c,p_1) \wedge connectsto(c,p_2) \wedge connectsto(c,p_3) \wedge$ $(p_1 \neq p_2) \wedge (p_2 \neq p_3) \wedge (p_1 \neq p_3)]$
Phase 4 Model Maturity (Phase 3 axioms, plus the following are satisfied)		
4.1	If any two assets exchange some input/output, those assets are connected to at least one common conduit.	$(\forall p_1 \in P)(\forall p_2 \in P)(\exists n \in N)$ $[exchanges(p_1,p_2,n) \rightarrow ((\exists c \in C) connectsto(c,p_1) \wedge connectsto(c,p_2))]$
4.2	Every exchanged input/output between any two assets is transferred by some conduit that connects to those assets.	$(\forall p_1 \in P)(\forall p_2 \in P)(\forall n \in N)$ $[exchanges(p_1,p_2,n) \rightarrow$ $((\exists c \in C) transferredby(n,c) \wedge connectsto(c,p_1) \wedge connectsto(c,p_2))]$
Phase 5 Model Maturity (Phase 4 axioms, plus the following are satisfied)		
5.1	Each asset generates an input/output to or receives an input/output from at least one other disjoint asset.	$(\forall p_1 \in P)(\exists p_2 \in P)(\exists n \in N)[(produces(p_1,n) \wedge consumes(p_2,n)) \vee$ $(consumes(p_1,n) \wedge produces(p_2,n)) \wedge p_1 \neq p_2]$

Figure 5: Example Model Maturity Axioms from (Giammarco K. , 2014).

Overall, Giammarco's work is a valuable contribution on the topic of measuring model maturity. The use of axioms that can be used to assess a model's elements and their linkages could be combined with some of the ideas on assessing design maturity through the number and frequency of changes to the design from (O'Brien & Smith, 1995) to form a more comprehensive set of metrics for MBSE model maturity.

A Digital Engineering Measurement Framework

While not directly aimed at providing an approach to measuring MBSE model maturity, the INCOSE Practical Software and Systems Measurement (PSM) Digital Engineering (DE) Measurement Framework (INCOSE, 2022) provides a set of measures and measurement process for DE maturity. The DE Measurement Framework was reviewed as there appeared to be value in identifying any measures in the DE framework that could be applicable to MBSE model maturity assessments.

The report provides some important word on the challenges with measurement, which were experienced by the authors during the work to define useable metrics covered in later sections of this paper:

"*A challenge with measures is both ensuring that they provide information needed to support decision making and that they are actually collected and used. A small set of measures should be tailored for each program and organization, focused on those needed for fact-based decision making. The measures should be regularly reviewed to ensure they are being used and that the decisions being made using those measures are producing the intended outcomes*" (INCOSE, 2022) (pp9).

While referring to the digital repository in which DE artifacts are stored, or the Authoritative Source of Truth (ASoT), the authors make a point relevant to MBSE models: "the number and type of model elements in the ASoT will be determined by the development process". It could be argued that there is also a link between the number/type of MBSE model elements based on what the goal of the modelling is within the development process.

Interestingly the authors note: "The DE measurement approach and associated measures should recognize a defined concept of a model element such that 1) the relative size of the DE effort can be measured and compared to other efforts or plans, and 2) the quality of the DE design decisions (correctness and completeness) can be measured." (INCOSE, 2022) (pp12). The authors also note: "In order to extract measurement information from the ASoT, the project must determine the type of model elements it will measure. These will be constrained by the tools selected" (INCOSE, 2022). A key point the authors make is: "Additional work is required to standardize on guidance for model elements that are most relevant to SE measurement". This point is related to the criticality of the metamodel in measuring MBSE model maturity that was highlighted by (Gaskell & Harrison, 2019) because the metamodel is the basis of the type of elements used in a MBSE model.

The DE Measurement Framework gives a very useful information model from the Practical Software Measurement group that highlights the links from measurement information needs through measurable concepts and constructs, into a specific attribute to be measured in a data repository or entity. The DE Measurement Framework also provides a table of Information Categories, Measurable concepts and (potential) Measures for DE, which are summarised to include those measures that may be applicable to measuring MBSE model maturity in Table 1.

Table 1: Summary of the DE Measurement Framework Information Categories, Measurable Concepts and Potential Measures that could be taken from an MBSE model (INCOSE, 2022).

Information Category	Measurable Concepts	Potential Measures
Schedule and Progress	Architectural Completeness	Architecture Completeness (from a specific viewpoint) and Volatility
Schedule and Progress	Model Coverage	Model Traceability Model Coverage (e.g., modelled elements)
Size and Stability	Functional Size and Stability	Product Size (e.g., Model Elements) Architecture Completeness and Volatility Functions Identified Functional Change Requests
Product Quality	Functional Correctness	DE (i.e., MBSE) Anomalies
Product Quality	Functional Correctness	Adaptability and rework Acceptance of Completed Work (e.g., Model Elements, Artifacts) Rework or rework defects

Product Quality	Functional Correctness	Model Traceability Traceability Anomalies
Process Performance	Process Effectiveness	Model Element DE (i.e., MBSE) Anomalies
Process Performance	Process Effectiveness	DE (i.e. MBSE) Anomalies Rework Effort Reworked Model Elements
Process Performance	Process Efficiency - Automation	Product Automation Cycle Time
Process Performance	Process Efficiency - Speed	Deployment Lead Time Cycle Time
Process Performance	Process Efficiency	Productivity Model Elements/Release Artifacts/Release
Technology Effectiveness	Technology Performance	Runtime Performance Elapsed Time

Finally, section 8 of the DE Measurement Framework (INCOSE, 2022) provides a set of measurement specifications, which give a robust set of items for each measure in the Framework that includes; a description of the measure, terminology, information need, base measures, derived measures, indicator description and example. Specification documents like these would be a very practical means of capturing the definition of each measure used to assess MBSE model maturity.

Key Findings from the Literature Review

The key findings from the literature review included that the approach of working backwards from the required model outputs is useful. Furthermore, MSRs are generally used to mitigate program risk, which again, will be useful for MBSE modelling activities. However, the defence MSR checklist used in (Gaskell & Harrison, 2019) will not always be applicable for MBSE modelling projects using different lifecycle models.

Another key finding was that a defined and well implemented metamodel with explicit model element classes is a necessary enabler of model maturity measurement. This is because the stereotypes, or data labels of the model elements and relationships will be the attributes within the model database that are measured and it helps ensure a quality model is developed.

Model change volatility appears to provide an indication of model maturity, however, there are many potential sources for model change volatility, which potentially makes it unreliable. To investigate this potential, a MBSE model from a previous Shoal project was analysed with the changes over a rolling four week period extracted for the project duration (84 weeks). This results of this analysis are shown in Figure 6.

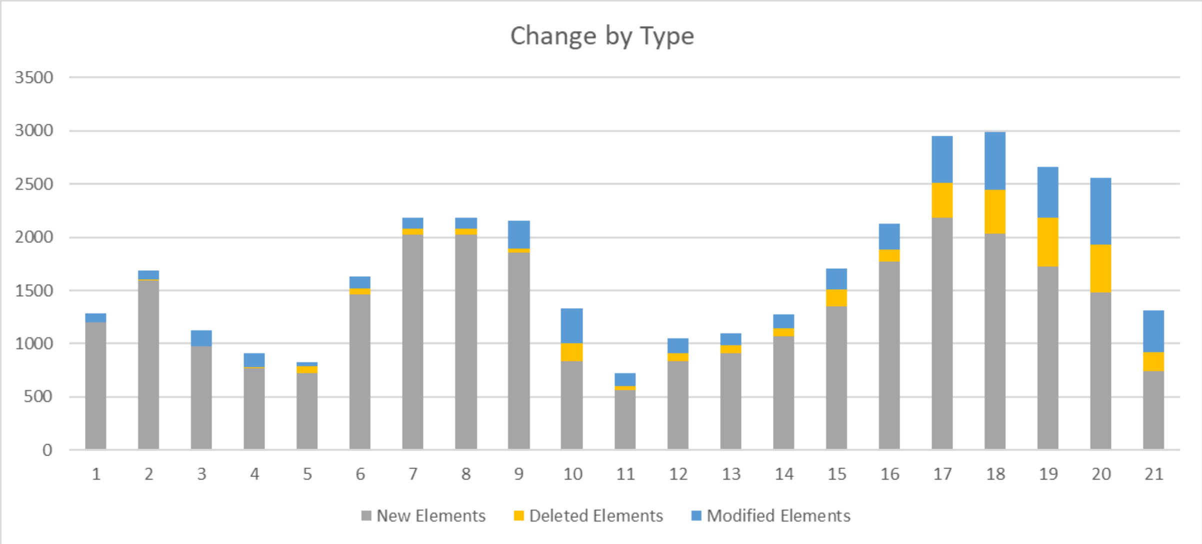

Figure 6: MBSE Model Changes over the Duration of a Project.

From Figure 6 it can be seen that the raw number of changes doesn't really follow a pattern, however, the number of modified elements increases as the total number of changes slows down, but there is no discernible link with the number of changes and model maturity. The validity of these statements is also limited due to the analysis only covering a single model, further work is ongoing, but client model access limitations and other constraints limit the range of models available for analysis. It would also be interesting to conduct an analysis to identify any second and third order effects associated with change volatility that may provide insight into model maturity.

The measurement process from the INCOSE Digital Engineering (DE) measurement framework (INCOSE, 2022) provides a useful basis from which to develop an MBSE model maturity assessment method. The measurement concepts and base measures described in the INCOSE DE measurement framework (INCOSE, 2022) could be sourced from a library of existing measures, be elicited from key stakeholders for the modelling program, or a combination of both.

Another key finding from the literature was that model quality metrics were identified as a related measurable concept for model maturity, however, quality metrics can be difficult to measure in a quantitative manner. This meant that using quality metrics to measure maturity were deemed to be out of scope for this work.

There appears to be value in calculating MBSE model maturity metrics from within the model. The feasibility of doing this in different modelling tools needs to be ascertained, but (Gaskell & Harrison, 2019) demonstrated a capability to do this within a common MBSE modelling tool.

Combining these insights from the review of the open literature, the gap that needs to be addressed for measuring MBSE model maturity on Shoal projects is a flexible method that can adapt to the different program lifecycles and types of architectures (i.e. operational, functional, physical, etc.) being modelled.

The DE measurement framework provides a robust basis for such a method and ideally, the metrics would be calculated from within an MBSE model.

Proposed MBSE Model Maturity Assessment Method

Based on these key findings an MBSE model maturity assessment method is proposed in Figure 7 that extends the DE Measurement Framework measurement process.

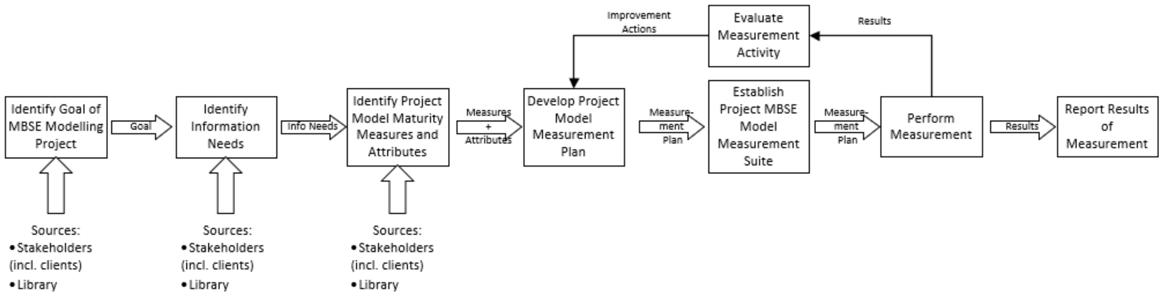

Figure 7: Proposed MBSE Model Maturity Assessment Method for use on Projects.

From Figure 7 the first three activities in the method could use a library of MBSE model maturity measures as a source for measures on a project that involves building an MBSE model. It is also worth noting the “Establish Project MBSE Model Measurement Suite” could be done in an MBSE model by leveraging inbuilt model Metric Suite features, in a similar manner to (Gaskell & Harrison, 2019). The following sections cover work undertaken at Shoal to investigate and establish a library of MBSE model maturity measures.

Building a Library of MBSE Maturity Measures

The approach adopted to build a library of MBSE model maturity measures was to emulate the approach of (Gaskell & Harrison, 2019) and work backwards from the goals of previous and predicted future MBSE modelling projects. A workshop of Shoal modelling Subject Matter Experts (SME) was convened to elicit and subsequently group the goals. Initially the intent was to elicit the previous goals of Shoal MBSE modelling projects and group them into modelling projects that were either above, or below the line (i.e., client, or delivery side; left, or right of contract; acquirer, or supplier). However, through the workshop discussions, it was evident that the goals could be grouped by the architecture level that was being modelled. This led to the process shown in Figure 8 being used to develop the MBSE maturity measure library. Note the SIPOC (Suppliers, Inputs, Process, Outputs, Consumers) approach used in step three will be explained in a subsequent subsection.

Figure 8: MBSE Maturity Model Library Development Process.

Modelling Framework

The first step in the process was to define the modelling framework. The modelling framework that emerged from a discussion of Shoal SME focused on the architecture levels that were identified. The levels were the Operational Architecture, Functional Architecture, Logical Architecture, and Physical Architecture levels as shown in Figure 9.

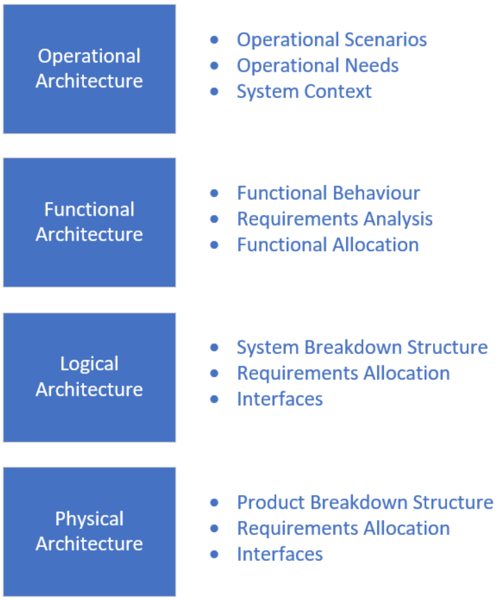

Figure 9: Architecture Levels for Grouping Modelling Goals and example modelling activities.

These architecture levels are consistent with those in defense architecture frameworks such as the Unified Architecture Framework® (UAF®), Department of Defense Architecture Framework (DoDAF), and the NATO Architecture Framework (NAF).

Previous Modelling Goals

During the workshop of Shoal SME, a suite of previous MBSE modelling goals was established. The full suite of modelling goals is not reproduced here, but to provide an indication, the modelling goals at each of the architecture levels have been included:

- Operational level – to generate a complete set of Measures of Effectiveness (MOEs).
- Functional level – to generate a complete set of Measures of Performance (MOPs).
- Logical level – to develop a System Breakdown Structure (SBS).
- Physical level- to develop a Product Breakdown Structure (PBS).

SIPOC Diagrams

SIPOC (Suppliers, Inputs, Process, Outputs, Customers) diagrams have been used in the Six Sigma approach to business process improvement to assist with the scoping of the process to be improved (Marques & Requeijo, 2009). These diagrams were identified as having potential to assist with the identification of predicted modelling goals at each level of the modelling framework shown in Figure 9 due to the ability to treat MBSE model development as a process, then identify the modelling outputs at each level. These predicted modelling outputs can be viewed as the goals of the MBSE modelling activity or process (analogous to the MSR checklist criteria used by (Gaskell & Harrison, 2019). As described above, these predicted goals can then be used to work backwards and track progress towards their completion. A SIPOC diagram generally takes the form shown in Figure 10.

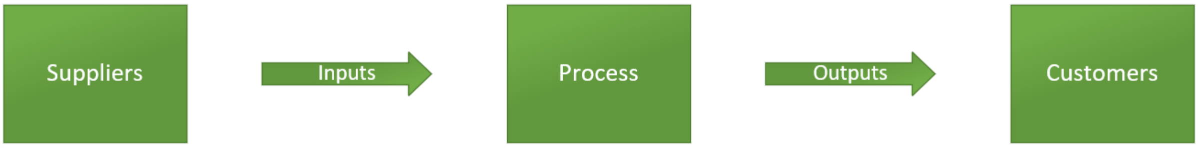

Figure 10: SIPOC Diagram General Format

SIPOC diagrams were developed for each of the four architecture levels shown in Figure 9, which were subsequently reviewed by Shoal modelers so that additional modelling goals that may arise in future MBSE modelling projects could be identified. The modelling goals also allow the associated MBSE model maturity measures to be identified and included in a MBSE model maturity measure library. As an example, the SIPOC diagram developed for the physical architecture modelling level is shown in Figure 11.

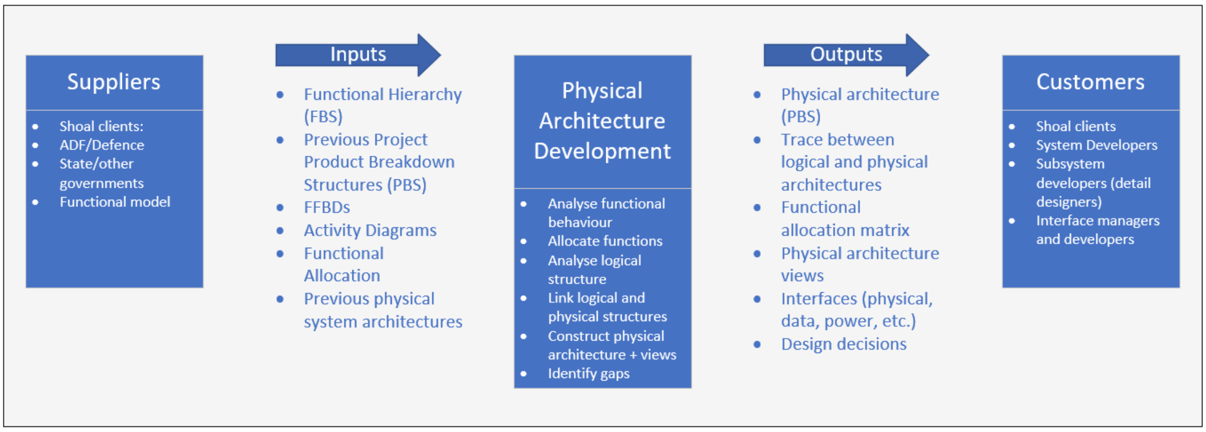

Figure 11: Physical architecture level MBSE modelling SIPOC diagram.

Grouping Modelling Goals

Following development of the SIPOC diagrams for each of the architecture levels in Figure 9, the predicted outputs, or modelling goals in the SIPOC diagrams were compared to the modelling goals elicited during the workshop with Shoal modelers. Where there was alignment with the predicted goals from the SIPOC diagrams and workshop modelling goals, they were grouped together. In the case where SIPOC modelling outputs couldn't be grouped with the elicited modelling goals, they were kept in an "other" group. The full suite of MBSE modelling goals identified are not presented here, but as an example, the groupings of the modelling goals from the workshop and SIPOC diagram for the physical architecture level are shown in Table 2.

Table 2: Physical Architecture Level Modelling Goals from Workshop and SIPOC Diagrams.

Modelling Level	Modelling Goals (Workshop)	Modelling Goals (SIPOC Outputs)
Physical	Develop Breakdown Structures	Product Breakdown Structure (PBS)
	Assist in Requirements Definition	Requirements allocated to systems
		Requirements gap analysis
	Other	Develop physical architecture views
		Identify physical interfaces

From Table 2, for the physical architecture level, the SIPOC outputs were mostly aligned with the workshop modelling goals, except that the SIPOC analysis identified two additional goals. These additional, or "other" goals were for developing views of the physical architecture, such as those in a typical architecture framework, as well as identifying the physical interfaces of the system.

Goal Question Metric Analysis

The Goal Question Metric (GQM) approach has its roots in software development that provides a measurement framework for defining measurable goals in an organization (Basili, Caldiera, & Rombach, 1994). A GQM process commences with the identification of a set of quality and/or productivity goals (Basili, Caldiera, & Rombach, 1994), which was described in the previous section using the SIPOC diagrams at each architecture level. The next step is to develop questions that define how the goals are to be met (Basili, Caldiera, & Rombach, 1994). Finally, measures are specified that can answer each of the questions (Basili, Caldiera, & Rombach, 1994). It is worth noting that there are no specific rules on whether there needs to be a one-to-one, or many-to-one, or one-to-many mapping between the Goals, Questions and Metrics. The GQM approach was used to develop metrics for all four architecture levels shown in Figure 9. To provide an example of the MBSE model maturity measures that were generated, the physical architecture level modelling goals is shown in Table 3.

Table 3: Outputs from a GQM analysis for the physical architecture level modelling goals.

Modelling Level	**Modelling Goals (SIPOC Outputs)**	**Question?**	**Model Maturity Metrics**
Physical	Product Breakdown Structure (PBS)	How many levels are complete in the PBS?	Levels completed
		How complete is the system's PBS?	% lowest level PBS elements allocated to Configuration Items
	Requirements allocated to systems	How complete is the allocation of requirements to the phys. arch?	% unallocated requirements
	Requirements gap analysis	How many requirements/system elements are unallocated?	% Systems without trace to requirements
	Develop physical architecture views	How many of the required physical arch. views are complete?	% required physical arch. views complete
	Identify physical interfaces	Have all physical interfaces been identified?	% physical interfaces identified
	Trace between logical and physical arch.	How complete is the trace from logical to physical architectures?	% physical elements traced from logical elements

When the physical architecture level MBSE model maturity metrics shown in Table 3 are combined with the MBSE model maturity measures identified for the other architecture modelling levels (i.e. operational, functional and logical), a library of maturity measures is created that provides a starting point for an MBSE modelling project. As a final check, the MBSE model maturity measures identified through the SIPOC diagrams were peer reviewed by the modelers that generated the initial set of modelling goals based on previous Shoal projects. This peer review resulted in some minor adjustments to the set of measures as well as the development of some guidance on how to apply the measures on MBSE projects.

It is noted that there are challenges in calculating several of these measures shown in the right-hand column of Table 3 as they are percentages, which require an estimate of a final number of elements or relationships that will be created in the MBSE model. These estimates will improve as the project progresses and it is anticipated that as the measures are implemented on different MBSE modelling projects, knowledge upon which estimates can be made will be increased.

Conclusions

This paper covered a body of work undertaken to develop an approach to measure MBSE model maturity that leveraged a library of measures to track the progress of Shoal projects that involve MBSE modelling. From the literature review, it appears that several efforts have been undertaken to create methods to measure model maturity in different domains, however, none of these were directly suitable for Shoal's MBSE modelling projects.

A research program was established to construct a library supported approach to measure the maturity of an MBSE model as it is developed on a project. An initial library of MBSE model maturity measures was developed by firstly setting a modelling framework consisting of four architecture levels: operational; functional; logical; and physical. Then, using an approach of working backwards from the goals of modelling at each layer for previous projects, as well as a SIPOC analysis, a suite of MBSE model maturity measures was established. This approach to generating a library of model maturity measures for use on projects appears to be feasible for most organisations undertaking MBSE modelling.

The proposed MBSE model maturity assessment method shown in Figure 7, highlights that each modelling project an organization undertakes could use a combination of consultation with clients and other stakeholders, as well as an MBSE model maturity measures library, to establish and monitor progress over the project's duration. Furthermore, the MBSE model maturity measures library can be refined further and their accuracy improved as more projects use the approach and experience with using the measures is gained. The accuracy improvements for the metrics that include "percentage complete" type scores, which necessitate an initial estimate size of the PBS for example, would be significant as experience applying the metrics is gained.

A final key conclusion is that this work reinforced the criticality of defining a modelling goal before commencing any modelling project.

Further Work

Further work is required to fully develop and implement both the model maturity assessment method shown in Figure 7, as well as to refine the measures library. Work is underway within Shoal to run a pilot test of the MBSE model maturity assessment method on an MBSE modelling project. This pilot test is intended to investigate whether the method and library are usable and identify their key value in practice.

Following the pilot study, the MBSE model maturity assessment method and library can be improved based on the pilot study findings. If the pilot study demonstrates potential, then the method and library can be deployed onto modelling projects. A final step is to include model maturity metric suites within the MBSE modelling tools used on projects for continuous progress tracking.

Acknowledgements

The authors would like to acknowledge the input and review provided by colleagues during the work that is covered in this paper. Franck Poubeau, Ashok Samalan and Tim Carter all participated in workshops to identify previous modelling project goals and the maturity measures that were subsequently developed. Tommie Liddy of Turen analysed previous project MBSE models and provided the analysis shown in Figure 6.

References

Basili, V. R., Caldiera, G., & Rombach, H. D. (1994). The Goal Question Metric Approach. *Encyclopedia of Software Engineering*, 528-532.

Etzel, C., Hofhammer, F., & Bauer, B. (2020). Towards Metrics for Analyzing System Architectures Modeled with EAST-ADL. *Proceedings of the 8th International Conference on Model-Driven Engineering and Software Development - MODELSWARD*, (pp. pages 441-448.). doi:10.5220/0009165704410448

Garcia, G., Golparvar-Fard, M., de la Garza, J., & Fischer, M. (2021). Measuring Progress and Productivity in Model-Driven Engineering for Capital Project Delivery. *Journal of Construction Engineering and Management.*

Gaskell, J. D., & Harrison, C. N. (2019). Improved System Engineering Technical Review's Entrance/Exit Criteria with Model Maturity Metrics. *ISSE 2019 – 5th IEEE International Symposium on Systems Engineering, Proceedings.*

Giachetti, R. (2017). Making the case for quality metrics for conceptual models in systems engineering. *12th Systems of Systems Engineering Conference.* doi:10.1109/SYSOSE.2017.7994940

Giammarco. (2010). Formal methods for architecture model assessment in systems engineering. *8th Conference on Systems Engineering Research (CSER2010)*, (pp. 522 - 531). Hoboken, NJ.

Giammarco, K. (2014). A formal method for assessing architecture model and design maturity using domain-independent patterns. *Conference on Systems Engineering Research (CSER 2014)*, (pp. 555 - 564). Redondo Beach, CA.

INCOSE. (2015). *Systems Engineering Handbook* (Fourth ed.). INCOSE.

INCOSE. (2022). *Practical Software and Systems Measurement (PSM) Digital Engineering Measurement Framework.* INCOSE-TP-2022-002-01.

INCOSE. (2023). *Systems Engineering Handbook* (Fifth ed.). San Diego, CA: INCOSE.

ISO/IEC/IEEE. (2017). *ISO/IEC/IEEE 15939 Systems and Software Engineering - Measurement Process.* New York: ISO/IEC/IEEE.

ISO/IEC/IEEE. (2021). Draft DIS 24641 - Systems and Software Engineering - Methods and Tools for Model-Based Systems and Software Engineering. ISO/IEC/IEEE.

Marques, P. A., & Requeijo, J. G. (2009). SIPOC: A Six Sigma Tool Helping on ISO 9000 Quality Management. *XIII Congreso de Ingenieria de Organizacion*, (pp. 1229-1239).

Morris, B. A., Harvey, D., Robinson, K. P., & Cook, S. C. (2016). Issues in Conceptual Design and MBSE Successes: Insights from the Model-Based Conceptual Design Surveys. *26th Annual INCOSE International Symposium (IS 2016).* Edinburgh, Scotland: INCOSE.

O'Brien, C., & Smith, S. (1995). Design maturity assessment for concurrent engineering co-ordination. *International Journal of Production Economics, Volume 41*(Issues 1–3), Pages 311-320.

Rodano, M., & Giammarco, K. (2013). A formal method for evaluation of a modeled system architecture. *Procedia Computer Science, 20*, pp.210-215.