Model Assurance Levels (MALs) for Managing Model-based Engineering (MBE) Development Efforts

Julie S. Fant and Robert G. Pettit
The Aerospace Corporation, Chantilly, Virginia, U.S.A.

Keywords: Model-based Engineering, UML, Software Models, Managing MBE, Model Quality and Value.

Abstract: Model-based engineering (MBE) in industry is on the rise. However, improvements are still needed on the oversight and management of MBE efforts. Frequently, program managers and high-level decision makers do not have background in MBE to understand models, the value the models are providing, and if they are successfully achieving their MBE goals. To address these concerns, we developed a rating scale for models, called Model Assurance Levels (MALs). The purpose of the MALs is to be able to quickly and concisely express the assurance the model is providing to the program, as well as, risks associated with the model. Therefore, given a MAL level, program managers and decision makers will be able to quickly understand the model value and risks associated with the model. They can then make informed decisions about the future direction of MBE development effort.

1 INTRODUCTION

Model-based engineering (MBE) in industry is on the rise. It is a term that is often thrown around within industry and it is broadly used to describe any engineering effort that utilizes models. However, there is a wide range of activities that can be performed as part of a MBE effort (Chaudron, 2017) (Liebel et al., 2018) (Bencomo et al., 2019) (Pettit IV and Mezcciani, 2013) (Pettit et al., 2014). For example, some MBE efforts may focus on addressing just the functional requirements models at a high-level. While other MBE efforts will include addressing non-functional requirements and use models for simulation. Given this broad usage of the term in industry, it can be difficult for those without detailed knowledge of MBE to truly understand what the model is being used for and how much benefit and risk reduction it is providing. Therefore, a bridge is needed to quickly and concisely translate a complex model into terms program managers and high-level decision makes can understand.

Currently, there are no repeatable approaches that help program managers and high-level decision makers understand what type of information is in the model, understand if the model is maturing as expected, and determine if the model is providing enough value and risk reduction to the program. Current approaches to assess and evaluate MBE artifacts are either ad hoc or subjective based on the evaluators experience with MBE. This leads to inconsistent results and advice given to program managers and decision makers. Without being able to understand and manage MBE efforts, the benefits of the MBE may be lost, and/or the quality of the resulting system degraded.

To address these concerns, we developed a rating scale for models, called Model Assurance Levels (MALs). The purpose of the MALs to be able to quickly and concisely express the assurance the model is providing to the program. Additionally, each MAL level has a certain level of risk associated with it. Therefore, given a MAL level, program managers and decision makers will be able to quickly understand the risks associated with the model and make informed decisions about the development effort and the direction it is going. Additionally, MALs can be also be used by engineers building the model as a means to effectively communicate information in the model and set expectations for what should be in the model.

2 RELATED WORK

There is limited research in the area of model scales and levels. There are notable works on technology readiness levels, such as National Aeronautics and

Fant, J. and Pettit, R.
Model Assurance Levels (MALs) for Managing Model-based Engineering (MBE) Development Efforts.
DOI: 10.5220/0007697505420549
In *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2019)*, pages 542-549
ISBN: 978-989-758-358-2

Space Administration (NASA) Technology Readiness Levels (TRLs) (NASA,) (NASA, 2012) and the European Space Agency (ESA) Technology Readiness Levels (TRLs) (European Space Agency,) (European Space Agency, 2008). However, the TRLs are primarily focused on assessing the maturity of technology and they do not translate well to assessing models because they dont take into account what is included in the model.

Another area of related work is on maturity levels. Maturity levels have more an emphasis on assessing the processes and the role of models, rather than on the models themselves. First, Kleppe and Warmer (Kleppe et al., 2003) propose a set of Modeling Maturity Levels. These levels characterize the role of modeling in a software project. They do not address the content and risks on the models themselves. Additionally, Kleppe and Warmer have extensive levels prior to model use, but only one level for models, which is not detailed enough for managing model based efforts. Second, Rios et al (Rios et al., 2006) propose a Model-Driven Development (MDD) Maturity Model. This work however is roadmap for adoption of models. Finally, there is the Capability Maturity Model Integration (CMMI) appraisal program. CMMI was originally developed by Mellon University (CMU), but is now administer by the CMMI Institute (CMMI Institute, LLC, 2018) (Kneuper, 2018). CMMI assess the maturity of an organizations software processes and a CMMI level is given based on the maturity of an organization. Given a CMMI, one can quickly and concisely understand the maturity of an organizations software development processes. CMMI can be applied to organizations using MBE. However, CMMI levels assess processes rather than models themselves. Therefore, they are not as useful with trying to understand what type of information and how much information is being captured in a model during the actual development effort.

3 OVERVIEW OF MALs

Model Assurance Levels (MALs) are a measurement system for model value, content, and quality. MALs are based on a scale from one to three with three being the highest to reflect increasing value and risk reduction of the model. The MAL scale is conceptually depicted in Figure 1. Within each MAL level, there are also sub-levels to show incremental growth. When applying MALs, not every program needs to achieve the highest MAL level. Instead, a MAL level should be selected based on the amount of risk and cost a program is willing to accept. A higher MAL score will help to reduced risk since the increased modeling effort will find and correct faults earlier in the lifecycle. However, this will also cause the cost to increase since more time and effort is needed for the additional modeling effort. The higher modeling costs are recouped later in the lifecycle by reducing rework and testing. Additionally, the resulting system will be of higher quality. Thus, when setting a MAL goal, careful consideration should be taken to ensure the right balance of risk reduction and cost.

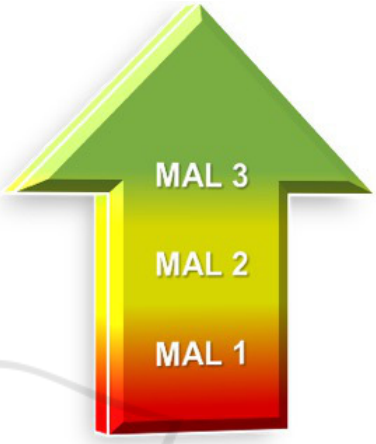

Figure 1: Conceptual MAL Scoring.

3.1 Benefits of MALs

Using MALs to understand and assess MBE efforts has several benefits. First, MAL levels are very easy to understand, even without details knowledge of MBE. MAL levels can be uses to consistently convey information about models.

Second, MALs are acquisition and development approach agnostic. Therefore, they can be determined and assessed regardless of acquisition structure or development lifecycle approach. This makes the approach flexible and broadly applicable across programs and customers.

Third, MAL scores dont have to be universal across a program or project. Different domain or subsystems models can have different desired MALs. For example, flight software models may seek to achieve higher MALs than ground processing software models. This is because flight software is often considered safety critical therefore reducing risk is more important on these types of programs (Ganesan et al., 2016).

Another benefit of MALs is they can be specified in acquisition language and proposals. This can help set MBE expectations between government and contractor during proposal and contract award stages, as well as help avoid confusion on what types and the

amount of content that will be developed. For example, on government acquisitions a contractor can state they plan to have their flight software model at certain MAL level by Preliminary Design Review (PDR) and a different and more mature MAL level by Critical Design Review (CDR).

4 MAL SCALE

Developing a scale to assess and define categories of software models is very difficult because software models are multifaceted. Models typically vary by breadth (i.e. how much information is modeled), by depth (i.e. how detailed is the information in the model), and by fidelity (i.e. how much validation and verification (V&V) is performed on the model. Therefore, we first developed a high-level MAL scale that broadly groups models into the different categories by breadth of information in the model, as captured in Table 1. We believe that all software models can fall into one of these three categories.

Table 1: MAL Scale.

MAL Level		Description
3	Advanced Models	An advanced model is defined as a model that addresses not only functional software requirements, but also the non-functional requirements such as software performance & reliability.
2	Basic Models	A basic model is defined as a model that addresses all the functional requirements, which include security requirements.
1	Sparse Models	A sparse model is defined as a model that only addresses a subset of functional requirements, which includes security requirements.

While these three MAL levels provide some distinction between different models, they are not sufficient for managing MBE efforts because they do not address depth or fidelity. For example, using just these three levels a high-level analysis model that addresses all functional requirements, but doesnt not contain detailed design information such as concurrency would receive a level 2. A detailed design model that address all the functional requirements and contains additional information detailed information such as concurrency, would also receive a score of 2. A detailed design model provides greater risk reduction than a higher level analysis model since additional implementation details have been thought out and included in the design. Therefore, sublevels are needed to add distinctions between these types of models.

Next, we further decomposed the three MAL level into sublevels that address depth of a model. For depth of a model, we used the terms Analysis Model and Detailed Design Model. These terms are commonly used in MBE methods such as (Gomaa, 2016) (Gomaa, 2011). An analysis model captures the software classes and their interactions at a high-level. An analysis may define operations for classes, but they lack detailed implementation information such as parameters, types, and concurrency. Detailed design models contain detailed implementation information about the software, such as data types, concurrency, and data formats. Detailed design models should reduce the risk on a program since the lower level implementation issues are thought out and addressed prior to implementation. Therefore, on the MAL scale, an analysis model would have a lower score that a detailed design model.

Since MALs are intended to be used to help decision makers manage a MBE development effort, using just model depth and breadth is not enough. A final aspect that helps program managers and decision makers reduce risk on a program is through model V&V. The more V&V that is performed on the model, the more confidence we have the models are correct and the more risk reduction they provide. Therefore, we further decomposed the MAL scale by the amount of V&V that is performed on the model. We used the terms Sparsely V&V to indicate that only a subset of the software functionality in the model underwent V&V and Completely V&V to indicate that the majority of the software functionality underwent V&V.

The final MAL scale with sublevels that address model depth and amount of V&V performed on the software functionality model is captured in Table 2.

At level one, we only used sparsely V&V since because the model only addresses a subset of the functional requirements, therefore complete V&V of the requirements in not possible. Both MAL levels two and three contain sublevels for both sparsely V&V and completely V&V since they address all the functional requirements in the model.

The last addition we made to the sublevels was in MAL Level 3. Advances are being made every day in MBE and new techniques are being developed for other uses of models. Therefore, we added two more sublevels to address when models are used to drive implementation testing and when models are incorporated into operational systems. These types of models need to be highly trusted, therefore it is assumed these models would need to have a lot of breadth, a lot of depth, and need to be highly V&V, therefore they are included at the top of the MAL scale.

Table 2: MAL Scale with Sub Levels.

MAL Level		Sub Level	Description
3	Advanced Models	3.8	Advanced Detailed Design Model incorporated into operational system
		3.7	Advanced Detailed Design Model used to drive implementation testing
		3.6	Completely V&V Advanced Detailed Design Model
		3.5	Sparsely V&V Advanced Detailed Design Model
		3.4	Advanced Detailed Design Model
		3.3	Completely V&V Advanced Analysis Model
		3.2	Sparse V&V Advanced Analysis Model
		3.1	Advanced Analysis Model
2	Basic Models	2.6	Completely V&V Basic Detailed Design Model
		2.5	Sparsely V&V Basic Detailed Design Model
		2.4	Basic Detailed Design Model
		2.3	Completely V&V Basic Analysis Model
		2.2	Sparsely V&V Basic Analysis Model
		2.1	Basic Analysis Model
1	Sparse Models	1.4	Sparsely V&V Sparse Detailed Design Model
		1.3	Sparse Detailed Design Model
		1.2	Sparsely V&V Space Analysis Model
		1.1	Sparse Analysis Model

4.1 Associating Risk with MAL Levels

The MAL scale with sublevels is useful for understanding the type of information in a model. However, in order for decision makers to effectively understand and make informed decisions about the direction of the model, risks need to be associated with each level. That way, decision makers can understand the risk they are taking on.

To associate risk to the MAL scale, we need to determine the risk associated with the depth, breadth, and amount of V&V performed on the model. This is accomplished by determining the risk associated with what is not included in that type of model. For example, Table 3 show the risks associated with different model breadths. So, if a decision maker finds out that their programs model is a Basic Model, then they know there is risk associated with the unknown behavior on alternative paths and error conditions, as well as, the potential to not meet non-functional requirements. Now they are informed about the program risks and can effectively decide how they wish to address them.

Table 3: Risks associated with model breadth.

Breadth Category	Risks	Potential Mitigations
Sparse Model	Risk associated with the part of the software that is not modelled. In other words, you don't know what you don't know.	• Mature the software model to MAL Level 2 or 3 • Perform extra testing and V&V on areas of the model that are not addressed in the model
Basic Model and Sparse Model	Risk associated with alternative paths and error conditions	• Mature model to MAL level 3 • Perform extra testing & V&V on anomalous and error conditions
Basic Model and Sparse Model	Risk that non-functional requirements will not be met	• Mature model to MAL level 3 • Perform extra testing & V&V on non-functional requirements
Advanced Model	Low risk	• N/A

Similar risk tables were developed for model depth and fidelity. Using these risks, the overall risks for each MAL sublevel can be determine. By selecting the risks from the appropriate categories.

5 CASE STUDY: AUTOMATED VEHICLE GUIDANCE SYSTEM

MALs have been applied to an Aerospace Corporation customer. MALs received very positive feedback for the value it provides to decision makers and the program. Unfortunately, the details of these customer model cannot be openly published. Therefore, representative example will be used to illustrate the values MALs provide decision makers. The representative case study is an Automated Vehicle Guidance System with an academic model from (Gomaa, 2011). The AVGS software is responsible for moving automated vehicles around a factory in a counter clockwise direction where they stop/start at selected stations to collect parts. An external Supervisory system sends the commands to the vehicles. The overall context diagram for the AVGS is show in Figure 2.

In this case study, a government program manager is responsible for acquiring the AVGS. They would like get a MAL level during Preliminary Design Review (PDR) and Critical Design Review (CDR) to understand if the model is maturing as planned, to identify any risks, and to identify any areas for improve-

ment for the model.

5.1 First MAL Assessment

The first MAL assessment is performed at PDR. The contractors provided a copy of the model in IBM Rational Rhapsody to use in the assessment. First, to assess model depth, the level of detail provided on classes and interactions is examined. This is illustrated using Figure 3. It shows a portion of the AVGS class diagram and the properties associated with the Arm Boundary class. In this example, the Arm Boundary class has operations defined. However, the operations parameters and data types are not defined. This is consistent with model depth of an Analysis Model, which only contains high-level details. Since all the classes and dynamic views in the model are at this same level of detail, the model is categorized as an Analysis Model.

Next, the model breadth is examined. This is accomplished by examining how much of the software is captured in the model. This will be illustrated using the state chart for the Vehicle Control Class in Figure 4. This state chart clearly shows how the Vehicle Control Class is guiding the automated vehicle to meet its functional requirements, like moving to different locations in the factory and loading/unloading parts. However, it fails to show what happens if there is an error in this process. What happens if the vehicle gets stuck or cannot move anymore? What happens if it goes to unload a part and there is no part to grab because it accidentally fell out? This state chart and other dynamic and static views in the model support that the model does not address alternative paths and error conditions.

Additionally, the security requirements are not addressed by the model. For instance, we do not see any user authentication required to operate the vehicle or any verifying of vehicle commands to ensure that only valid locations within the factory are entered as a destination. Since the model does not address security, alternative paths, and error conditions, it is given a model depth of sparse model.

Finally, the amount of V&V performed on the model is assessed. The contractor does not have the requirements traced to model elements. However, there is an implied traceability since the sequence diagrams are traced to the use cases. Additionally, all classes are in at least one sequence diagram. Therefore, it can be assumed that they are all functionally required. The contractor also stated that they performed manual walk-throughs on the sequence diagrams. Since some V&V was performed on the model, but other types of V&V, such as simulation, were not performed, the model is categorized as Sparsely V&V.

In summary, the model is categorized as a Sparsely V&V Sparse Analysis Model, which is a MAL level 1.2.

The MAL score and risks associated with this score are presented to the government decision maker. The risks include:

- Risk associated with the part of the software that is not modeled in this case the security requirements.
- Risk that alternative paths and error conditions are unknown and wont be discovered until implementation phase
- Risk that non-functional requirements will not be met
- Implementation risk since lower level implementation details are not addressed in the model and any faults wont be discovered until implementation phase.

The program has a set budget for the MBE effort, so the government program manager must decide the best way to mature the model for CDR. Aerospace provided two courses of action: First, the model can be matured to a MAL 1.4, where the depth of the model can be increased, and the breadth of the model remain unchanged. This would reduce the implementation risk on a program since implementation details would be added to the model. There would still be outstanding risks including not addressing alternative paths, error conditions, security, and non-functional requirements. The second course of action is to mature the model to a MAL 2.2. This would entail keeping the current model depth but expanding the model breadth to address. alternative paths, error conditions, security, and non-functional requirements. The outstanding risk at level would be implementation risk, since additional implementation details about the model are not being added.

Given this information about the model and potential risks, the government program manager decided to mature the model to a MAL 1.4. since security, alternative paths, and non-functional requirements are less important on the AVGS. The government decision maker would rather reduce the implementation risks.

5.2 Follow-up MAL Assessment

The purpose of the follow-up MAL assessment is to determine if the model matured to the desired MAL 1.4 level. Therefore, during CDR an updated copy of the model is provided to assess.

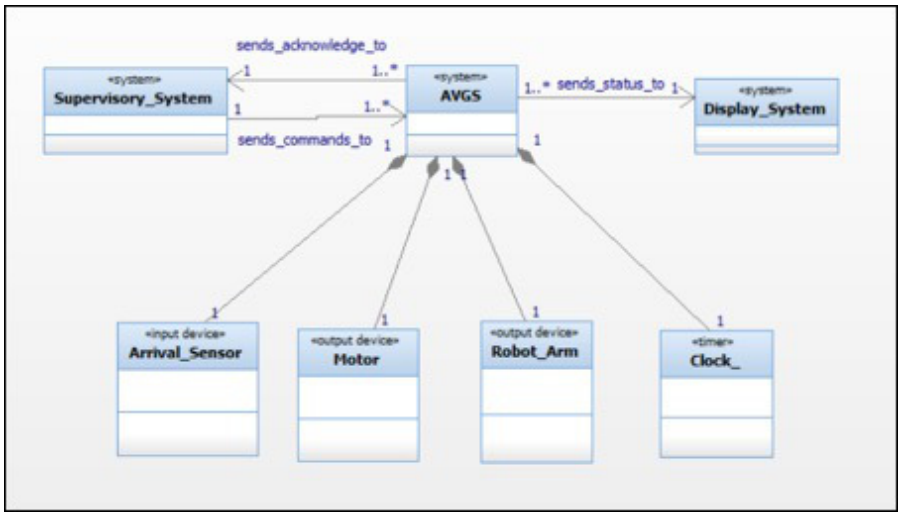

Figure 2: AVGS Context Diagram.

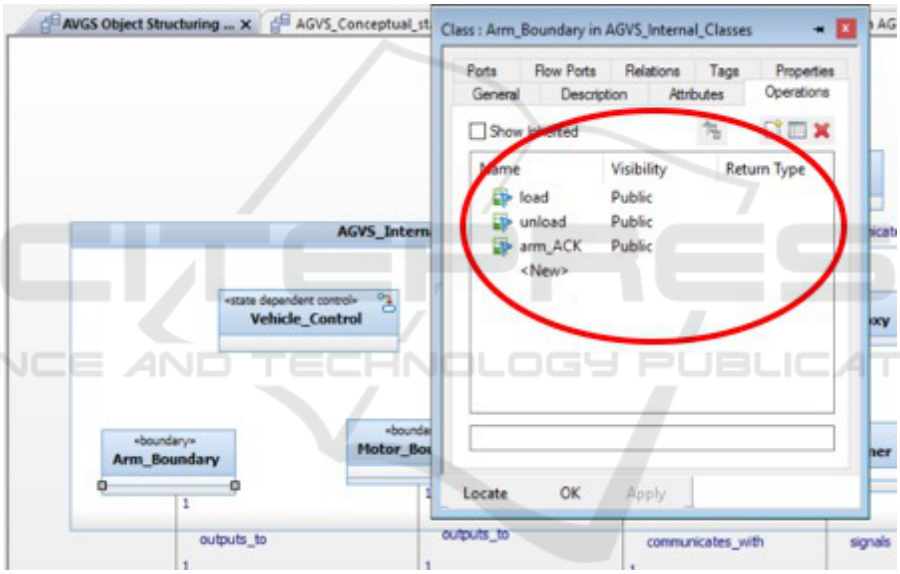

Figure 3: AVGS Class Diagram.

First, the level of detail provided on classes and interactions is examined to assess model depth. This is illustrated using Figure 5, which shows an updated portion of the AVGS class diagram and the properties associated with the Arm Boundary class. In this example, the Arm Boundary class has operations defined. Additionally, the class now contains the operations parameters and data types. Since all the classes and dynamic views in the model at this same level of detail, the model is categorized as a Detailed Design Model. This is consistent with the desired MAL level of 1.4.

Next, the model breadth is examined. This is accomplished by examining how much of the software is captured in the model. The state chart for the Vehicle Control class is examined again. While it was updated to reflect the detailed messages. However, the state chart still only addresses how its functional requirements are being met. No additional behavior was added to the diagram. This is also true of other dynamic views in the model, which still do not address alternative paths, error conditions, and security requirements. Thus, the model remains a Sparse Model, which is consistent with the desired MAL 1.4 level.

Finally, the amount of V&V performed on the model is assessed. The contractor performed still similar V&V as before. Therefore, the model is categorized as Sparsely V&V Model.

In summary, the updated model is categorized as a

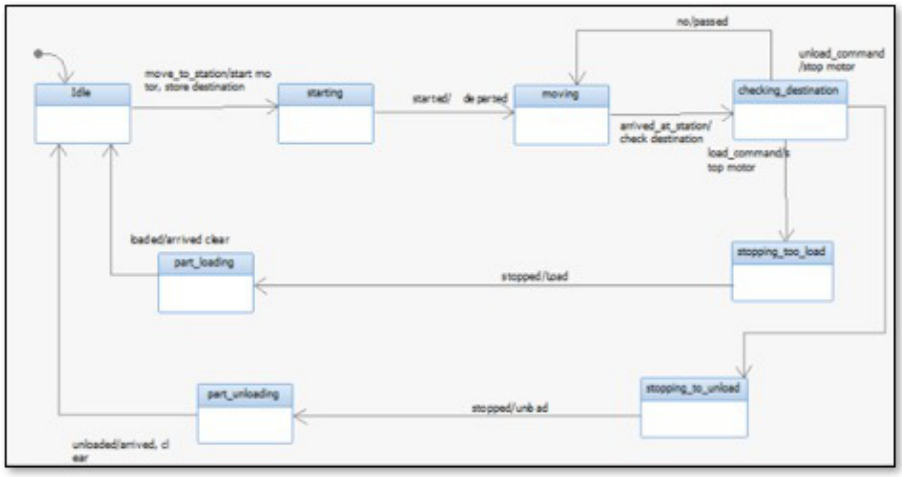

Figure 4: Vehicle Control State Chart.

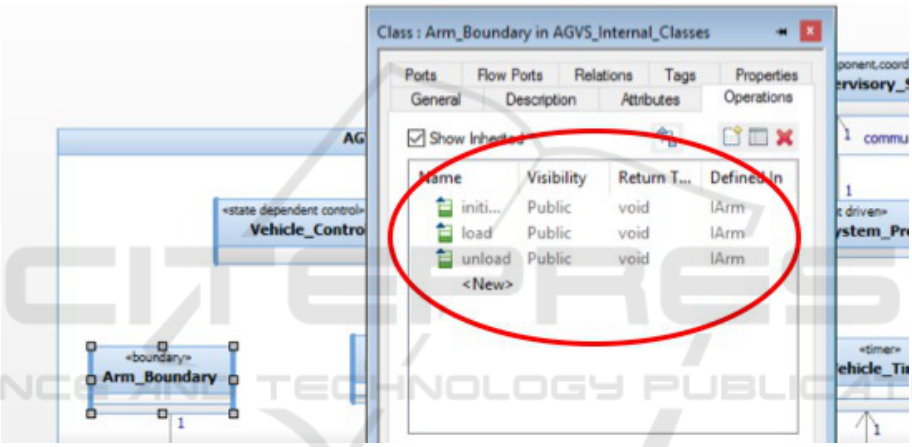

Figure 5: Updated AVGS Class Details.

Sparsely V&V Sparse Detailed Design Model, which is a MAL level 1.4. This is consistent with the desired MAL level the government program manager set as a goal for CDR. Now, as they move into the implementation phase, the government can focus their risk reduction resources on addressing the outstanding risk to address alternative paths, error conditions, security, and non-functional requirements.

5.3 Case Study Summary

In summary, the use of MALs helped the government program manager to understand what was being modeled, to shape the direction of the model, to understand the current risks, and to effectively uses limited resources. Using MALs, the program manager was able to quickly identify risks and was able to determine the best course of action for the program, given their limited budget and resources. Additionally, by using MALs the government program manager was able to understand the progress on the model, as well as, the type of progress made between two versions of the model. Finally, the case study illustrated how MALs can be used to set expectations and direction of MBE efforts between contractors and government.

6 NEXT STEPS

The concept of software MAL scale and scores provides value to program managers and decision makers. However, MAL scores must be determined and applied in a consistent manner in order for the scores to be meaningful. For example, what happens is 60% of the model is at the analysis level and 40% is at the detailed design level. Should it be a level 2.1 or a level 2.4? Opinion on this matter may vary depending on who is performing the assessment. Therefore, next steps include developing a repeatable and quan-

tifiable assessment method so that the interpretation of the MAL scale is consistent regardless of who is performing the assessment. Additionally, tool support to automate the MAL assessment process will also be pursued. Automated tool support will make the approach more practical and it will provide faster results to decision makers.

Another area of future work is to increase the applicability to agile methodology, where models may not be being developed. Research is needed to determine if MALs can be effectively applied on models reverse engineered from code. This is useful on Agile efforts where only code is being developed.

Finally, the last area of future work is to expand MALs into the systems engineering space and create a separate scale and assessment method for enterprise models.

7 CONCLUSIONS

In conclusion, there exists a need in industry to help program managers and high-level decisions makers understand and manage MBE efforts. This paper describes the first step in establishing a repeatable and concise way to describe models to decision makers using Model Assurance Levels (MALs). MALs provide the means to express a lot of information about a software model into a single score. Each MAL score has as level of associated risk, which further helps program managers and decisions makers make informed decision about the direction and expectations of the model. MALs are development and acquisition approach agnostic, therefore they have broad applicability across industry.

REFERENCES

Bencomo, N., Götz, S., and Song, H. (2019). Models@ run. time: a guided tour of the state of the art and research challenges. *Software & Systems Modeling*, pages 1–34.

Chaudron, M. R. (2017). Empirical studies into UML in practice: pitfalls and prospects. In *Modelling in Software Engineering (MiSE), 2017 IEEE/ACM 9th International Workshop on*, pages 3–4. IEEE.

CMMI Institute, LLC (2018). CMMI Institute. https://cmmiinstitute.com.

European Space Agency. ESA Technology Readiness Levels.

European Space Agency (2008). Technology readiness levels handbook for space applications.

Ganesan, D., Lindvall, M., Hafsteinsson, S., Cleaveland, R., Strege, S. L., and Moleski, W. (2016). Experience report: Model-based test automation of a concurrent flight software bus. In *Software Reliability Engineering (ISSRE), 2016 IEEE 27th International Symposium on*, pages 445–454. IEEE.

Gomaa, H. (2011). *Software modeling and design: UML, use cases, patterns, and software architectures*. Cambridge University Press.

Gomaa, H. (2016). *Real-Time Software Design for Embedded Systems*. Cambridge University Press.

Kleppe, A. G., Warmer, J., Warmer, J. B., and Bast, W. (2003). *MDA explained: the model driven architecture: practice and promise*. Addison-Wesley Professional.

Kneuper, R. (2018). Capability Maturity Model Integration for Development (CMMI-DEV).

Liebel, G., Marko, N., Tichy, M., Leitner, A., and Hansson, J. (2018). Model-based engineering in the embedded systems domain: an industrial survey on the state-of-practice. *Software & Systems Modeling*, 17(1):91–113.

NASA. TRL Definitions. https://www.nasa.gov/pdf/458490main_TRL_Definitions.pdf.

NASA (2012). Technology readiness levels. https://www.nasa.gov/directorates/heo/scan/engineering/technology/txt_accordion1.html.

Pettit, R. G., Mezcciani, N., and Fant, J. (2014). On the needs and challenges of model-based engineering for spaceflight software systems. In *Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), 2014 IEEE 17th International Symposium on*, pages 25–31. IEEE.

Pettit IV, R. G. and Mezcciani, N. (2013). Highlighting the challenges of model-based engineering for spaceflight software systems. In *Proceedings of the 5th International Workshop on Modeling in Software Engineering*, pages 51–54. IEEE Press.

Rios, E., Bozheva, T., Bediaga, A., and Guilloreau, N. (2006). MDD maturity model: A roadmap for introducing model-driven development. In *European Conference on Model Driven Architecture-Foundations and Applications*, pages 78–89. Springer.