Applying Model-Based Systems Engineering to the Development of a Test and Evaluation Tool for Unmanned Autonomous Systems

Solomon Gebreyohannes, Ali Karimoddini, and Abdollah Homaifar
shgebrey@aggies.ncat.edu, akarimod@ncat.edu, homaifar@ncat.edu
North Carolina A&T State University, Greensboro, NC 27411

***Abstract*—In this paper, we apply the Model-Based Systems Engineering (MBSE) concepts and approaches to the early phases of the development of a Test & Evaluation (T&E) tool for Unmanned Autonomous Systems (UASs). This helps meet the design requirements and maintain traceability (of design requirements and decisions for satisfying stakeholder's needs). UAS development is driving toward increasing levels of autonomy for unmanned systems. The dynamic, non-deterministic behavior of intelligent autonomous systems presents the testers with a significant challenge. The ability to predict the behavior and evaluate performance of increasingly intelligent systems, especially those that employ vision-based behaviors, is seen as a critical T&E shortfall. To address this challenge, we propose, in this paper, to use a high-fidelity simulation environment. This can significantly aid in the evaluation of UAS behaviors and their perception mechanisms. Such a high-fidelity simulator enables the testes to safely conduct a wide variety of mission scenarios to test an autonomous system by providing truth data to compare with the UAS's perceptions. A major challenge here is to manage the system modeling complexity and maintain traceability of design decisions made at each level of the development to meet stakeholder's needs. In this paper, we follow MBSE methodology and use Systems Modeling Language (SysML - a domain-specific modeling language for systems engineering used to specify, analyze, design, optimize, and verify systems) to establish a systematic framework for designing a T&E tool for UASs and to transform stakeholder's needs into design requirements to maintain traceability.**

***Index Terms*—Test & Evaluation, Model-Based Systems Engineering, Unmanned Autonomous Systems**

I. Introduction

As sensors and processors become more advanced, the algorithms for Unmanned Autonomous System (UAS) control will also become more autonomous. Treating the autonomy as a "black box" and evaluating its performance in simple pass/fail terms is insufficient to fully characterize the UAS's capabilities and limitations. Greater insight into how the UAS perceives its environment and makes decisions is necessary for testers and users to gain confidence and trust in using the UAS [1]. The ability to predict behaviors of increasingly intelligent systems of a UAS and evaluate their performance is seen as a critical Test & Evaluation (T&E) shortfall. Because testers seldom have access to the internal algorithms of a UAS, they need a tool that can provide them with insight into how a UAS perceives its world and what behaviors it will exhibit. Testing the UAS's perception mechanism using a high-fidelity simulation environment can significantly aid in the evaluation of UAS behaviors. Such high-fidelity simulators enable safely conducting a wide variety of mission scenarios to test a UAS by providing truth data to compare with the UAS's perceptions. However, it is challenging to manage the system modeling complexity and maintaining traceability of design decisions made at each level of the development to satisfy stakeholder's needs.

To manage the complexity of the design of a T&E tool for UASs, in this paper we propose to use Model-Based Systems Engineering (MBSE), which incorporates models (which are abstracted representations of reality) into the design process with a central and governing role in the specification, design, integration, validation, and operation of a system. MBSE is a paradigm shift from the traditional document-based and acquisition life cycle model approaches [2]. It can enable checking traceability (between design details and more abstract requirements) using simulation (due to the fact that it uses models) replacing trial-and-error revision in document-centric processes [3]. As a result, verification and validation become part of the design process, leading to a correct-by-design product, which is more reliable [4]. Systems Modeling Language (SysML) is a domain-specific modeling language for MBSE used to specify, analyze, design, optimize, and verify systems [5]. It is used to manage the complexity of systems modeling. These systems may include hardware, software, operations, messaging, facilities and other aspects of design [6]. MBSE has been applied to several applications, including small satellites [7], robots [8], railways [3], aircraft systems [9], [10], and industrial applications [11].

In this paper, we propose a systematic test and evaluation tool based on a high-fidelity simulation environment to address UAS T&E shortfall on testing highly autonomous systems. This will be achieved by using MBSE for a systematic and formal presentation of components of a simulation-based T&E tool, transforming stakeholder's needs into requirements to be incorporated in design, implementation, verification, and validation of systems under test. We use the MBSE methodology guidelines and SysML to formally capture processes of mission analysis, and definitions of needs, requirements, and architecture. This will enable the integration of design,

Corresponding author: A. Karimoddini. Address: 1601 East Market Street, Department of Electrical and Computer Engineering North Carolina A&T State University Greensboro, NC, US 27411. Email: akarimod@ncat.edu (Tel: +13362853313).

978-1-7281-5365-0/20/$31.00 ©2020 IEEE

development, validation, and verification processes.

The rest of this paper is organized as follows. Section II presents UAS T&E challenges and solutions. Section III introduces MBSE processes and SysML diagrams, and presents some MBSE applications. Section IV presents the proposed T&E framework and the approach for transformation of stakeholder's needs into formal requirements. Section V concludes this paper.

II. Testing of UASs in a High-Fidelity Simulation Environment

Testing in a high-fidelity simulation environment can significantly aid in the evaluation of UAS behaviors. Such simulators allow for safely conducting a wide variety of mission scenarios to provide truth data to compare with the UAS's perceptions. Simulation tools have been matured to the point that they can readily provide an immersive environment with realistic sensor stimuli for triggering the behaviors of intelligent autonomous systems. In addition to current simulation capabilities, the testers need tools that offer insight into the UAS's decision-making process. When a UAS exhibits an unexpected response, the testers will ask, "Why did it do that?" or "What mode is it operating in now?" The testers need the ability to peer into the autonomy process to see where the UAS's decision-making diverged from what was expected. Further, by modeling the UAS's behavior, testers will better be able to predict such occurrences in other scenarios. One can address these needs through the development of a methodology to monitor the System Under Test (SUT) behaviors, infer internal states of reasoning, and predict system performance across a wide range of scenarios. The test tool created with this methodology will enhance current simulation-based testing techniques.

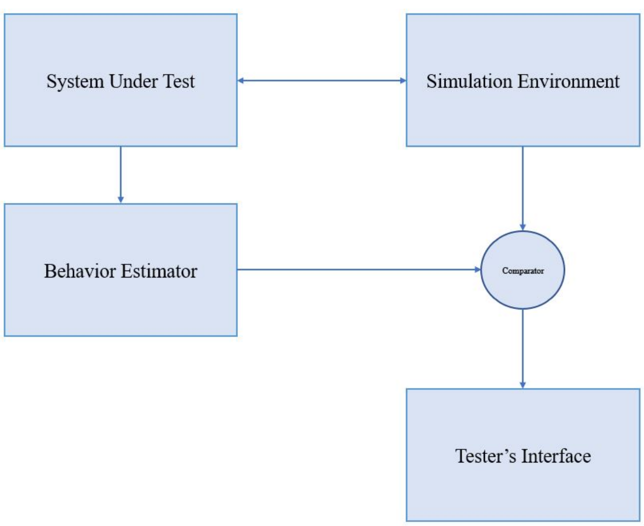

Fig. 1. Testing UAS in a high-fidelity simulation environment

The proposed system architecture for the UAS T&E tool is presented in Fig. 1. The inference engine (predictive technology in Fig. 1) taps into the connection between the SUT and the immersive simulation environment. The evaluation tool includes a graphical user interface for the testers to show comparisons between the UAS's perceptions and truth data in real-time.

III. Model-Based Systems Engineering

A. Overview

Systems engineering (SE) is a methodical, disciplined approach for the design, realization, technical management, operations, and retirement of a system [12]. SE is a technical and management process, which can help to effectively plan, design, analyze, and integrate complex systems, and manage their complexity [13], [14]. It seeks a safe and balanced design in the face of opposing interests and multiple, sometimes conflicting constraints [12].

A recent development in SE aims to represent the system components as models, which are abstracted representations of reality, which is defined as Model-Based Systems Engineering (MBSE). The MBSE methodology is about elevating models in the design process to a central and governing role in the specification, design, integration, validation, and operation of a system. This is different from traditional document-based approaches [2].

B. Systems Modeling Language

Systems Modeling Language (SysML) is a domain-specific modeling language for systems engineering that supports the analysis, design, and verification of complex systems in graphical notation [5]. These systems may include hardware, software, operations, messaging, facilities and other aspects of design [6]. It is, therefore, used to manage the complexity of systems modeling. The four main pillars of

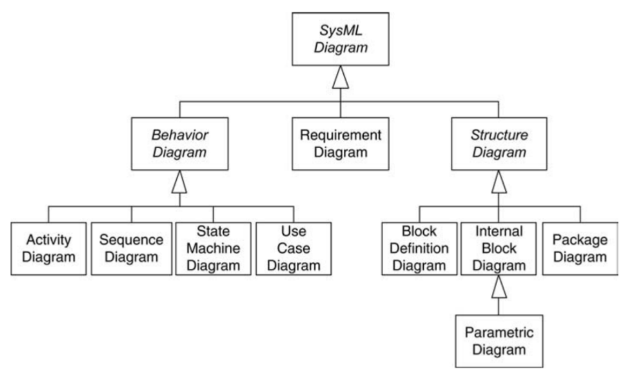

Fig. 2. SysML Taxonomy [6]

SysML are requirement, structure, behavior and parametric diagrams. Requirement diagrams are used to model text-based requirements and their relationships in the design. Structure diagrams are represented using blocks (definition, internal, and package blocks). Flow-based, message-based, and event-based behaviors are represented using activity, sequence and state machine diagrams, respectively [5], [6], [15]. Fig. 2 shows SysML taxonomy. There are different modeling tools for SysML, such as MagicDraw [16] and Papyrus [17], with the later being open source. The practical benefits and further applications of SysML are discussed in [18].

C. Applications of MBSE

Model-Based Systems Engineering has been applied to several applications. In [7], MBSE and SysML are used to model a standard CubeSat and applied for an actual mission - the aurora explorer mission. The authors extended their initial MBSE framework and demonstrated the ability to model different behaviors and scenarios in [19]. Then, they presented an integrated and executable MBSE representation of the application in [20]. MBSE (using SysML) is further used to define behaviors of CubeSats as an activity hierarchy (from mission requirements to functional architecture) [21]. In [8], MBSE is used for the development of robotic systems. The methodology is demonstrated using the collaborative robot applications. MBSE can also provide verification that considerably increases the robustness of monitoring systems and efficiency in their development as demonstrated by using a railway case study in [3]. The paper also shows the analysis of requirements and traceability within the model. In [9], MBSE is applied to an industrial test case to perform the functional design of control maintenance systems to be integrated with the aircraft fuel system. The research in [11] presents the use of MBSE for product engineering management concepts for industrial applications. Improved model-based engineering practices for software-reliant aircraft systems are presented in [10]. Further development such as merging reliability and maintenance activities with MBSE and its applications are presented in [22].

IV. Development of a UAS Test & Evaluation Tool using Model-Based Systems Engineering

System engineering processes enable systems engineers to coordinate interactions among different parties such as stakeholders, engineers, and operators [23]. These processes include the transformation of stakeholder's needs into requirements to be included in design, implementation, and operations of systems. This transformation leads to the creation of a sufficient set of requirements and reliable system solutions that address the desired capabilities and can be traced back to the stakeholder's needs - satisfying SE principle: "design right" and "right design". The risk of project failure would be unacceptably high if technical processes are not followed in systems design [23]. Therefore, we follow the INCOSE SE procedures for the design and implementation of a UAS T&E tool.

Fig. 3 shows the activities we will follow for designing a T&E tool and to systematically transform the stakeholders' needs to design requirements, which enables the integration of design, development, verification, and validation.

A. Test & Evaluation Mission Analysis Process

This section presents the test & evaluation mission analysis. The purpose of mission analysis process in general is

> "...To define the business or mission problem or opportunity, characterize the solution space, and determine potential solution class(es) that could address a problem or take advantage of an opportunity."
> (ISO/IEC/IEEE 15288, as restated in INCOSE handbook, p. 49 [23])

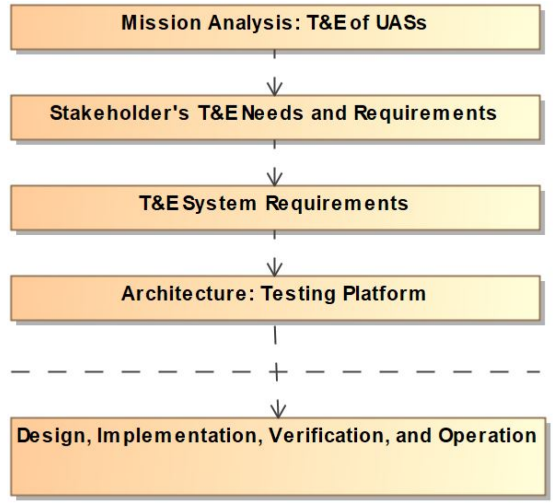

Fig. 3. Technical processes for T&E of UASs

The activity diagram shown in Fig. 4 outlines the major activities of T&E mission analysis. The mission analysis processes include preparation for mission analysis, problem definition, solution space, evaluation, and management of the mission analysis:

1) Strategy Preparation: The first step in mission analysis is to investigate the preparations needed to establish a strategy. This includes the needs for requirements of any enabling systems, products, or services [23]. Testing in a high-fidelity simulation environment can address T&E shortfall and be used by commercial companies or military organizations.

2) T&E Problem definition: This activity helps to identify and analyze the gaps with respect to the desired goals and objectives of an organization, and concept of operations (ConOps). Fig. 5 shows the problem definition using a requirement diagram.

The main goal is to test and evaluate an intelligent autonomous system from a cognitive perspective and correlate its internal processes with observed behaviors. It should satisfy the stakeholder's needs at different levels, which consists of a simulation environment and interface requirements.

3) Solution space: In this section, a possible solution and test operation of the solution from the user or operator's view (i.e., operational concept - OpsCon) are presented with the system context and interfaces.

As discussed in Section II, testing in a high-fidelity simulation environment is a solution for T&E of UASs. The solution holds valid under given assumptions as presented in Table I.

Table II shows the scope of the solution. It lists some limitations of the T&E tool.

The tester is the ultimate user of the T&E tool - selects the SUT, sets the environment parameters, and then start the test or simulation. A usecase diagram showing the tester's involvement in test operation is shown in Fig. 7.

The test operation is shown using a sequence diagram in Fig. 6.

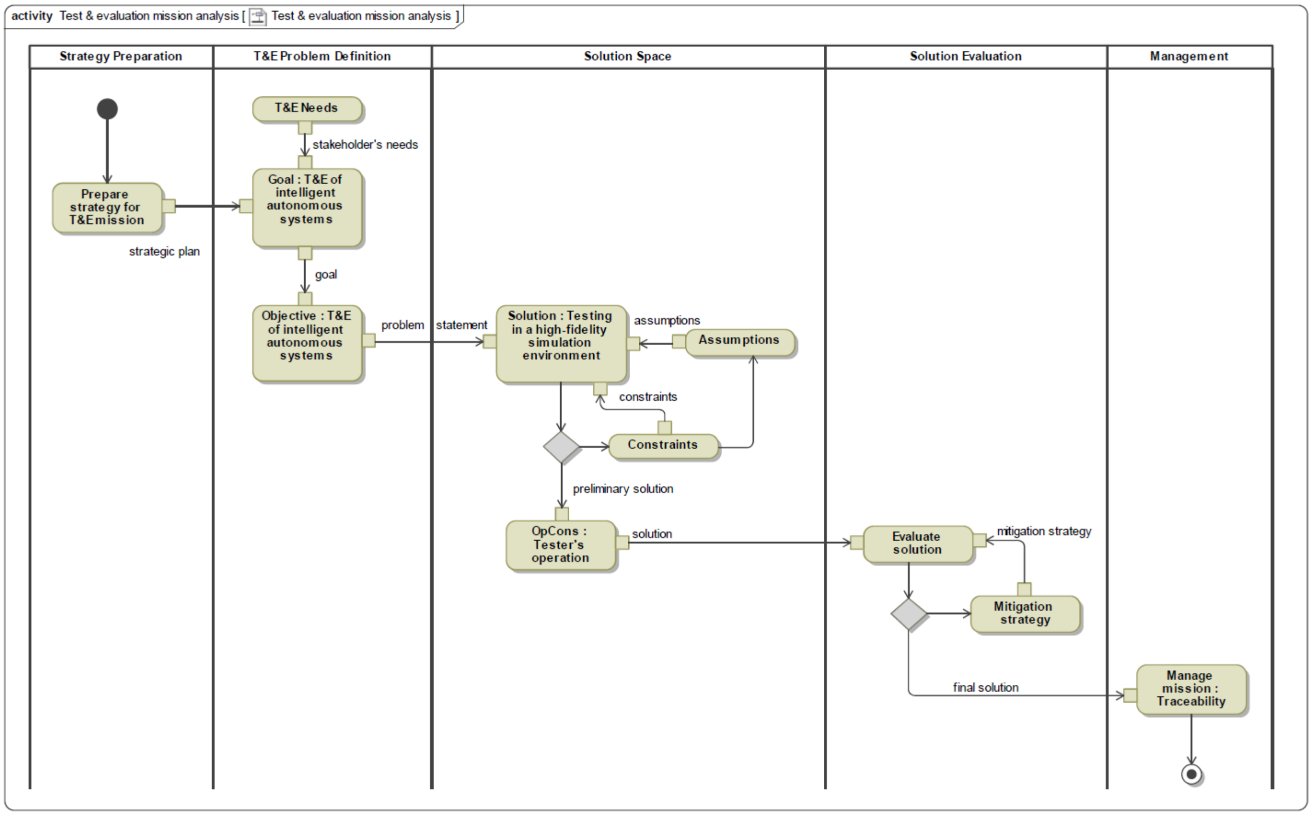

Fig. 4. Activity diagram for test & evaluation mission analysis

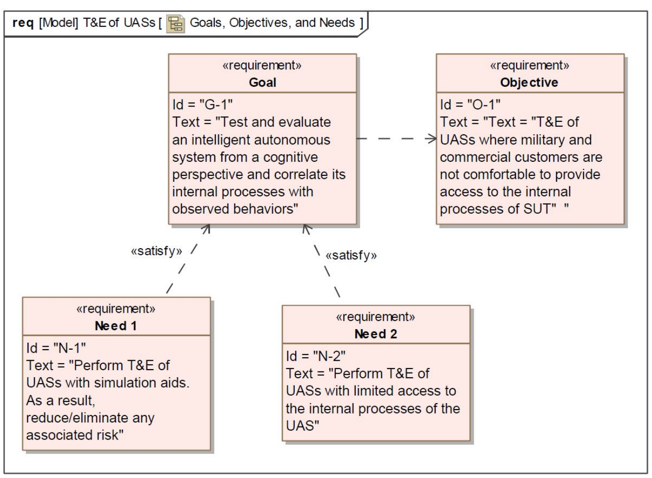

Fig. 5. Goals, Objectives, and Needs

TABLE I
ASSUMPTIONS

Id	Assumptions
A-01	SUT characteristics fed to the predictive technology are accurate or not largely altered due to noise.
A-02	The simulation tool can provide an immersive environment with realistic sensor stimuli for triggering the behaviors of intelligent autonomous systems.
A-03	Estimation technology needs are achievable.

TABLE II
CONSTRAINTS

Id	Constraints
C-01	The predictive technology needs to be trained before use.
C-02	The SUT needs to be one of the types of UAS that predictive technology is trained on (defined on maximum weight, nominal operating altitude, and speed parameters space)

4) Evaluate the solution: This simulation approach is feasible as it mainly is a programming task. Of course, one possible risk is how to simulate an immersive environment with realistic sensor stimuli. This is mitigated with a fact that most simulation tools have been matured to the point that they can readily provide such an environment.

5) Manage mission analysis: This activity is used to establish and maintain traceability. Traceability is the ability to verify the history or location of an information and its context. Maintaining traceability reduces or avoids a gap often exists between stakeholder's (vision and) needs and requirements representation.

B. Stakeholder Needs and Requirements Definition Process

As stated in ISO/IEC/IEEE 15288, the purpose of the stakeholder needs and requirements definition process is to define requirements for a system that can provide the capa-

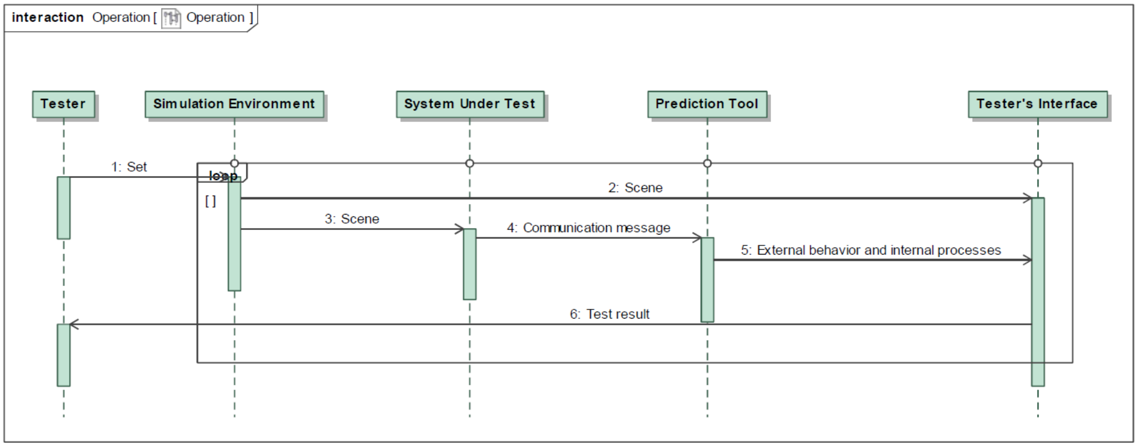

Fig. 6. Test operation using sequence diagram

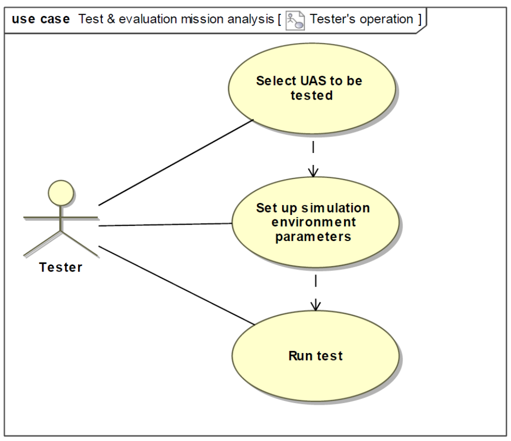

Fig. 7. Usecase diagram

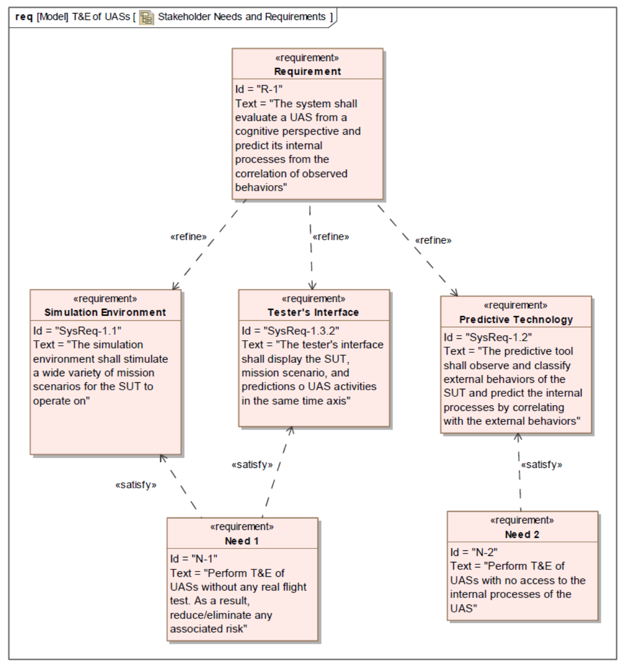

Fig. 8. Stakeholder's needs and requirements

bilities needed by users in a defined environment to meet the stakeholder's needs.

Fig. 8 summarizes UAS T&E requirements and stakeholder's needs.

C. System Requirements Definition Process

The purpose of the system requirements definition process is to "transform the stakeholder, user-oriented view of desired capabilities into a technical view of a solution that meets the operational needs of the user" (ISO/IEC/IEEE 15288, as restated in INCOSE handbook, p. 57 [23]).

Fig. 9 shows the system requirements using a requirement diagram.

SysReq 1: The system shall evaluate a UAS from a cognitive perspective and predict its internal processes from the correlation of observed behaviors.

- SysReq 1.1 Simulation environment: The simulation environment shall simulate a wide variety of mission scenarios for the system under test to operate on.
  - Rationale: In order to conclude that a UAV passes or fails a test, it should be tested under different mission scenarios.
- SysReq 1.2 Prediction tool: The prediction tool shall observe and classify external behaviors of the SUT and predict the internal processes by correlating with the external behaviors.
  - SysReq 1.2.1 Technology 1 (State Estimator) shall observe and classify external behaviors of a UAS. These include the state (search mode, survey mode,...) and triggers that initiate the transition between states.
  - SysReq 1.2.2 Technology 2 (Perception Estimator) shall predict internal processes of UAS; e.g., predict its image processing algorithm.
    * SysReq 1.2.2.1 Data requirement: Technology 2 needs to have access to outputs of Technology 1.
  - Data requirement: The prediction tool (both technologies) shall tap into the communication between

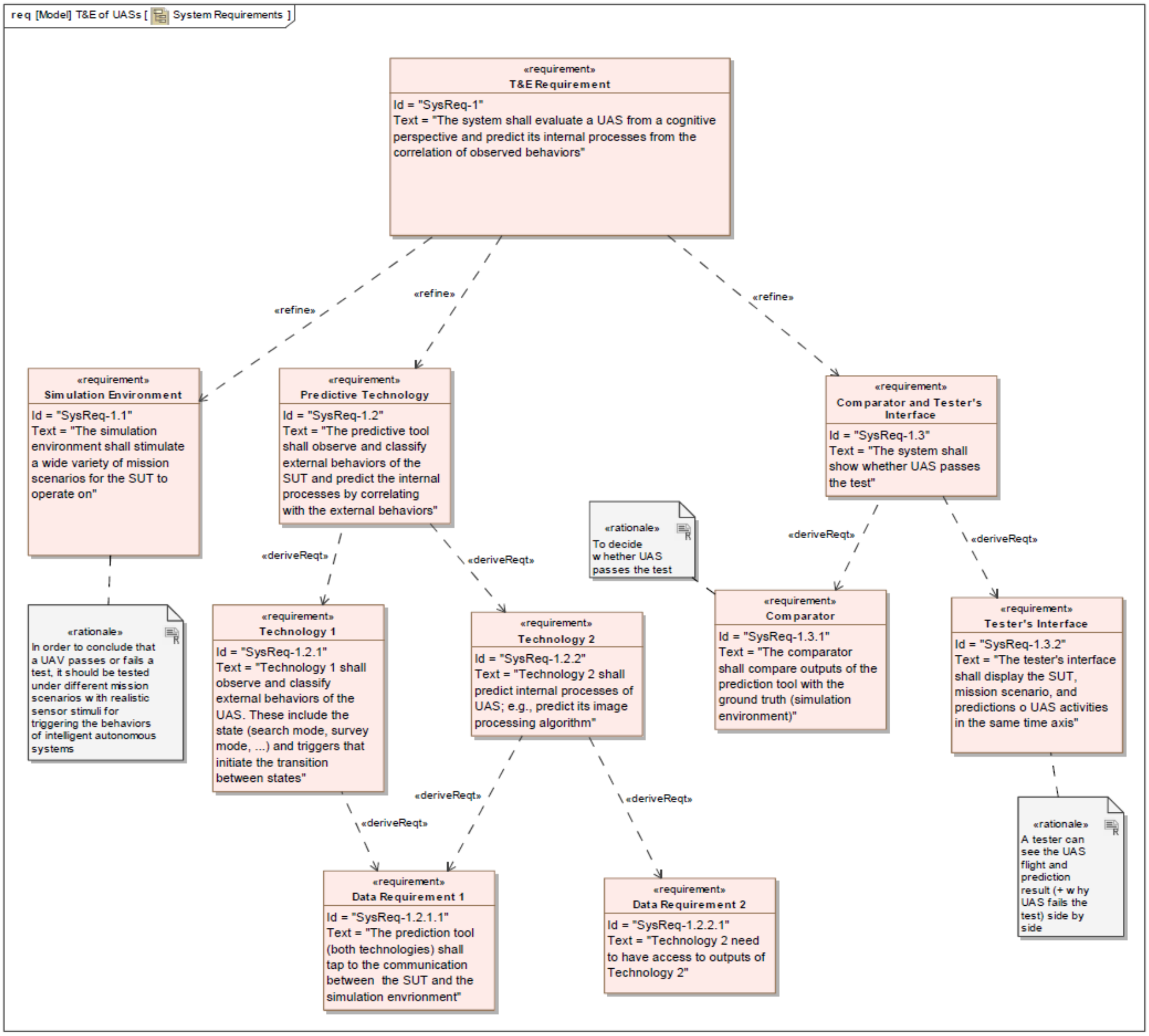

Fig. 9. System requirements

the SUT and the simulation environment.

- SysReq 1.3 Comparator and Tester's interface: The system shall show whether UAS passes the test.
  - SysReq 1.3.1 The comparator shall compare outputs of the prediction tool with the ground truth (simulation environment)
    - ∗ Rationale: To decide whether UAS passes the test.
  - SysReq 1.3.2 The tester's interface shall display the system under test, mission scenario, and predictions of UAS activities in the same time axis.
  - Data requirement: The comparator and tester's interface need to have access to the output of both technologies and the simulation environment.
  - Comment: A tester can see the UAS flight and prediction result (+ why UAS fails the test) side by side.

D. Architecture Definition Process

The purpose of the architecture definition process is to generate system architecture alternatives that meet system requirements [23]. This section presents the T&E system architecture alternative.

Fig. 10 presents the structure of T&E system architecture and relation between the components using block definition diagram (bdd). The system consists of a system under test, simulation environment, and prediction tool. The prediction tool comprises predictive technologies (1 and 2) and tester's interface.

Fig. 11 shows the architectural entities, i.e., input/output flows, system elements, and physical interfaces using internal block diagram (ibd).

The simulation environment provides a `scn: scene` for the SUT to operate on and for the tester's display as a ground truth. Technology 1 predicts the external behaviors of the UAS, such as the state it is operating, from communication, `msg: message` in Fig. 11, between the SUT and ground operator.

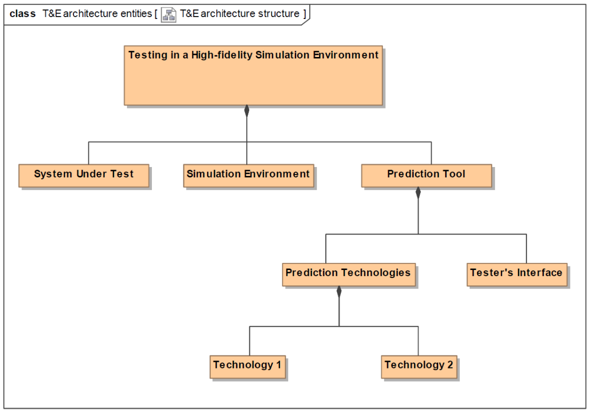

Fig. 10. System structure - Block definition diagram

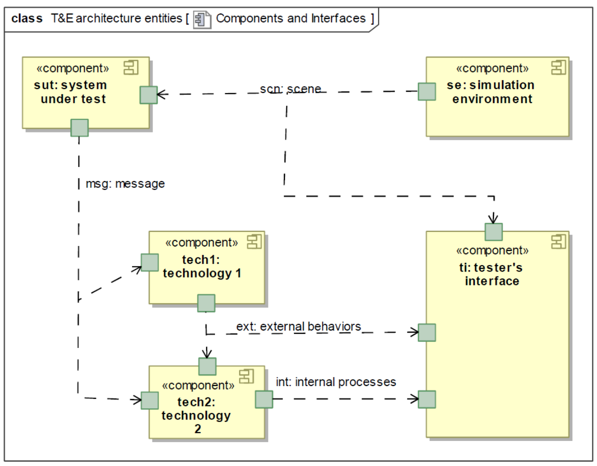

Fig. 11. Architectural entities - Internal block diagram

Finally, Technology 2 inputs all information and predicts the internal processing of the UAS to hint the tester as to why UAS performs a particular action. Both, `ext: external` and `int: internal` predictions, along with the `scn: scene` will be displayed on the tester's interface.

V. Conclusion

In this paper, we proposed UAS testing in a high-fidelity simulation environment to address UAS T&E shortfall and managed the development of UAS T&E tool using MBSE. In particular, we transformed stakeholder's needs into requirements. This covers processes of mission analysis, and definitions of needs, requirements, and architecture.

Acknowledgment

The authors would like to acknowledge the support from the National Science Foundation under the award number 1832110, Air Force Research Laboratory and OSD under agreement number FA8750-15-2-0116. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, AFRL, OSD, or the U.S. Government.

References

[1] A. Homaifar, A. Karimoddini, B. A. Erol, M. A. Khan, E. Tunstel, R. L. Roberts, R. F. Young, K. Snyder, R. S. Swanson, M. Jamshidi, Y. Seong, and E. A. Doucette, "Operationalizing autonomy: A transition from the innovation space to real-world operations," *IEEE Systems, Man, and Cybernetics Magazine*, vol. 5, no. 4, pp. 23–32, Oct 2019.

[2] J. A. Estefan *et al.*, "Survey of model-based systems engineering (mbse) methodologies," *Incose MBSE Focus Group*, vol. 25, no. 8, 2007.

[3] M. Blumenfeld, E. Stewart, and E. Larios, "Using model based systems engineering to increase the robustness of monitoring systems in railways," 2016.

[4] S. Gebreyohannes, A. Karimoddini, A. Homaifar, and A. Esterline, "Formal verification of a fuzzy rule-based classifier using the prototype verification system," in *Fuzzy Information Processing*, G. A. Barreto and R. Coelho, Eds. Cham: Springer International Publishing, 2018, pp. 1–12.

[5] S. Friedenthal, A. Moore, and R. Steiner, *A practical guide to SysML: the systems modeling language*. Morgan Kaufmann, 2009.

[6] L. Delligatti, *SysML distilled: A brief guide to the systems modeling language*. Pearson Education, Inc, 2014.

[7] S. C. Spangelo, D. Kaslow, C. Delp, B. Cole, L. Anderson, E. Fosse, B. S. Gilbert, L. Hartman, T. Kahn, and J. Cutler, "Applying model based systems engineering (mbse) to a standard cubesat," in *Aerospace Conference, 2012 IEEE*. IEEE, 2012, pp. 1–20.

[8] C. Hernandez and J. L. Fernandez-Sanchez, "Model-based systems engineering to design collaborative robotics applications," in *Systems Engineering Symposium (ISSE), 2017 IEEE International*. IEEE, 2017, pp. 1–6.

[9] C. Pessa, M. Cifaldi, E. Brusa, D. Ferretto, K. M. Malgieri, and N. Viola, "Integration of different mbse approaches within the design of a control maintenance system applied to the aircraft fuel system," in *Systems Engineering (ISSE), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 1–8.

[10] S. M. Simi, S. P. Mulholland, and W. G. Tanner, "Tes-savi awesum^{TM} model-based systems engineering (mbse) for facetm applications," in *Aerospace Conference, 2014 IEEE*. IEEE, 2014, pp. 1–17.

[11] C. Tschirner, L. Bretz, R. Dumitrescu, and J. Gausemeier, "Applying model-based systems engineering for product engineering management concepts for industrial application," in *Systems Engineering (ISSE), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 42–49.

[12] , "Nasa systems engineering handbook, nasa/sp-2007-6105," December 2007.

[13] A. P. Sage, "Systems engineering education," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 2, pp. 164–174, 2000.

[14] O. A. Asbjornsen and R. J. Hamann, "Toward a unified systems engineering education," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 2, pp. 175–182, 2000.

[15] H. Crisp, "Systems engineering vision 2020," *Seattle, Washington*, 2007.

[16] MagicDraw, in *url:http://www.nomagic.com*.

[17] Papyrus, in *url: http://www.topcased.org*.

[18] D. Bank, F. Blumrich, P. Kress, and C. Stöferle, "A systems engineering approach for a dynamic co-simulation of a sysml tool and matlab," in *Systems Conference (SysCon), 2016 Annual IEEE*. IEEE, 2016, pp. 1–6.

[19] S. C. Spangelo, J. Cutler, L. Anderson, E. Fosse, L. Cheng, R. Yntema, M. Bajaj, C. Delp, B. Cole, G. Soremekum *et al.*, "Model based systems engineering (mbse) applied to radio aurora explorer (rax) cubesat mission operational scenarios," in *Aerospace Conference, 2013 IEEE*. IEEE, 2013, pp. 1–18.

[20] D. Kaslow, G. Soremekun, H. Kim, and S. Spangelo, "Integrated model-based systems engineering (mbse) applied to the simulation of a cubesat mission," in *Aerospace Conference, 2014 IEEE*. IEEE, 2014, pp. 1–14.

[21] D. Kaslow, B. Ayres, P. T. Cahill, L. Hart, and R. Yntema, "A model-based systems engineering (mbse) approach for defining the behaviors of cubesats," in *Aerospace Conference, 2017 IEEE*. IEEE, 2017, pp. 1–14.

[22] M. Izygon, H. Wagner, S. Okon, L. Wang, M. Sargusingh, and J. Evans, "Facilitating r&m in spaceflight systems with mbse," in *Reliability and Maintainability Symposium (RAMS), 2016 Annual*. IEEE, 2016, pp. 1–6.

[23] D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin, and T. M. Shortell, *Systems engineering handbook: A guide for system life cycle processes and activities*. John Wiley & Sons, 2015.