28th annual INCOSE international symposium
Washington, DC, USA
July 7 - 12, 2018

Using Board Games as Subje
Developing Expertise in Mo -Based
Systems Engineering

Matthew Corrado
Georgia Institute of Technology
mattcorrado98@gmail.com

Kathryn Trase
NASA Glenn Research Center
216 ï433 ï8067
Kathryn.trase@nasa.gov

Abstract. As more organizations transition from traditional document-centric systems engineering to a model-based approach, many are challenged to train their staff in new languages, tools, and methodologies, while managing the expectations of stakeholders and their expected model outcomes. In particular, challenges associated with learning a new modeling language and developing skills in the óartô of modeling p
realizing this transition. This paper hypothesizes that systems engineers may more readily learn how to correctly model with SysML, and develop intuition about the art of modeling and using patterns, if their learning references a commonly and thoroughly-understood subject, such as a board game. This paper presents a case for the use of board games as subject matter for new modelers. It demonstrates the concept with a sample
Monopoly, and discusses the limitations of this approach and potential adaptations that may broaden the applicability of the learned skills to projects. Finally, results from a small feasibility assessment and concepts for more formal study to evaluate the hypothesis are presented.

Introduction

As organizations continue to transition their systems engineering tools and methodology to more model-based methods, they face challenges in training staff in new languages, cultivating intuition in the óartô of modeling, addressing stakehol
into current project management practices. In particular, challenges associated with learning a new modeling language and developing skills in the art of modeling present organizations with formidable obstacles to realizing this transition.

Training in the Systems Modeling Language (SysML) often references over-simplified sample models of subject matters, of which learners may only have a cursory understanding. For example, learners may know generally how an automobile (Friedenthal, Moore, & Steiner 2015) or spacecraft (Deligatti 2014) operates, but are not typically e
designs, or they may have different levels of understanding relative to their colleagues (whether fellow modelers or not). Pilot projects, such as that described by Vipavets, et al., often face challenges in proceeding to model at lower levels of detail than what was covered in initial training courses due to the limited depth of concept coverage.

SysML is the current de-facto language standard for a model-based approach, but has been identified as difficult to learn (Andersson et al. 2010) and difficult to specialize for a particular domain, especially by systems engineers who may not be trained in such topics as knowledge management and ontology development. Some (Voirin et al. 2015) have embarked on an approach to develop custom, domain-specific languages (DSLs) to get around the hurdle of having to learn new terminology and syntax, and make adoption more user-friendly. However, if SysML is to be utilized until the development of community-accepted DSLs, then a learning approach that focuses specifically on the language and the subtleti

While some modeling practitioners have cited the importance of modeling with a purpose and considering how the end stakeholder will use the information in the model (Hallqvist & Larsson 2016), there is a requisite learning period where new practitioners must simply be allowed to explore the language and associated techniques. Only after having grasped advanced aspects of the language and developing some confidence in successful modeling patterns can the new practitioner be able to think critically about a specific stakeholder and how to implement their concerns within the model content and structure.

Introductory aspects of SysML can be taught with top-level overviews of subject matters, but acquiring more advanced skill in applying the language and associated modeling techniques requires learners to have greater depth of understanding of the educational subject matter. By selecting a subject matter that learners understand thoroughly to the lowest level, training efforts can offer learning experiences that are more practically applicable to real project scenarios, including: how to model at different levels of abstraction while maintaining traceability between levels, or how to organize both the model and patterns to best achieve stakeholder objectives.

This paper makes the case for using board games as a commonly and thoroughly-understood subject matter for teaching the more subtle aspects of SysML and the art of modeling, followed by some example modeling of the game Monopoly by Hasbro, and finally, proposals for future investigations to validate the hypothesis of accelerated and deeper learning.

Premise: MBSE Learning Can Be Enhanced by Modeling Board Games

Board games may be suitable subject matter for new modeling practitioners to utilize when first learning to apply a modeling language and concepts. Most games are sufficiently multi-faceted in that several modeling language constructs must be utilized to capture the full nature of the game. These facets thus provide learners with opportunities to both apply several language constructs, as well as determine how to provide traceability between constructs (e.g., from requirements to behavior, or state machines to activities). While games may not necessarily be considered ñcomplexò by the systems engineering communi, because they do not consist of several parts that combine and transmit data in sophisticated ways (Ferreira 2001), one could consider games to be sufficiently complicated and thus warrant methodical description, and potentially analysis, via modeling.

By using a game as the modeled subject matter, students could focus their learning only on modeling principles and techniques, without simultaneously trying to learn the subject (as may be the case when an automobile engineer attempts to learn SysML using a model of a spacecraft, for example). Peer tutoring and remote learning may also be more effective when familiar games are utilized as model subjects. Presuming two people have played the same game, it would be easier for the learners to intuit more subtle aspects of modeling rules and conventions by reviewing models created by someone else. Sample models of games, even if the learner has not generated the model him- or herself, may also prove more useful as an example or reference, given a reasonably similar level of understanding of the subject matter. Learners would more readily be able to identify when a description of the game (as modeled) does not align with their understanding of its rules or behavior, and either discern a language or pattern subtlety, or identify a modeling error or cheater.

In addition to simply learning modeling languages and constructs, games may also provide an avenue toward learning more advanced modeling and simulation techniques. Game play could be modeled in the system model, and more complex probabilistic or mathematical models developed in external tools could be integrated with the system model to enable a higher-fidelity simulation. The learner benefits from already having a sense of the expected simulation outcome (does the simulation output match their experience in playing the game?), and can focus attention on learning the tools and integrated simulation techniques.

Modeling games can also provide an opportunity to develop skills that would be utilized in applying modeling techniques within a project environment. For example, skills in developing and generating documents from the model could be practiced: in this analogy, the rules of the game are to the players as the system requirements document is to the project. The learner can again focus on learning the mechanics of preparing content for document generation or stakeholder review, rather than also trying to learn the subject matter.

With a greater confidence in applying the language correctly and some intuition about the appropriate way to model a topic, the learner may be better equipped to solicit input from stakeholders regarding both the scope of the model and intended use of the information stored within the model. However, this paper does not attempt to provide guidance on how the systems engineer should decide which design aspects are the appropriate topics to model, in order to address stakeholder needs.

Characteristics of Games Suitable for Modeling

The wide variety of games and their diversity of game play mechanisms provides plenty of opportunities to apply different pillars of SysML, or emphasize some over others, and test modeling patterns in different scenar-and-s. Whi moveò (found in Chutes and Ladders or Candyland, e.g.), may be too simple in concept to be instructive on the details of SysML or provid with several mechanisms quickly become sufficiently complex to be candidates for learning modeling techniques.

The gaming mechanisms themselves may be an interesting case study for learning how to develop patterns (in this case, the game mechanisms) and adapt known patterns to a specific instance (i.e., to a particular set of games that each use that pattern but perhaps in a variant form). Table 1 lists a few game mechanisms that may be extensible to real project scenarios and lists a few associated modeling constructs.

Other characteristics of games suitable for modeling include, as a subset or in combination with each other:

- Multiple players, game pieces or markers, and conditions for winning or losing
- Multiple types of interactions between players
- Variable player roles, levels of difficulty, rules, or initial game set-up or conditions

Table 1: Sample game mechanisms of interest for learning SysML and modeling techniques (Adapted from Board Game Geek (n.d.))

Mechanism	Description (adapted from Board Game Geek (n.d.))	Sample Modeling Constructs
Trading	Players interchange game pieces, tokens, currency, etc. between each other.	Interface definition and constraints, sequences
Set Collection	Players work to achieve a specified combination of items or conditions. Collecting or achieving a sets typically results in award of points or game resources, or wins the game.	Constraints, states, objective functions
Variable Player Powers	Different role capabilities or features are assigned to different players within the same game, such that each player might play to variations of the main rules.	Variants, constraints, requirements, states, use cases, allocation
Deck/Pool Building	Players start the game with a given deck or set of cards or pieces, and add and/or exchange those pieces over the course of the game, or multiple games. Some games allow additional items t subsequently added to the set of cards. These new items typically enhance the abilities of the player or rules of the game in future sessions.	Architecture, context or domain definition, variants, constraints, use cases, allocation
Chit-Pull System	A player draws a random counter to indicate which group of units may be moved during that turn. In some cases, constraints may be placed on which type of units, the quantity of units, or possible behaviors of those units, that may be executed.	Activities, sequences, states, allocation

Monopoly Example

A thorough discussion of the intricacies of a SysML model of Monopoly necessitates a discussion of the qualities of Monopoly that make it a good candidate to model. First, the popular and widely understood board game exhibits most, if not all, of the characteristics described above. In essence, Monopoly is highly complex *and* widely understood, both of which need to be true of a game that is to be used as a learning and teaching tool. In fact, Monopoly is perceived as so simple that Hasbro recommends it for anyone who is at least eight years old, and most people are typically exposed to it at that age (Jingles 2013). The gameôs ubiquity affords games: Monopoly only seems simple because a large number of people have a pre-existing familiarity with it, as it is arguably one of the most popular board games of all time (Kismet 2017). Without this familiarity, Monopoly would reveal itself as a complex system with moving parts, competing objectives, interfaces, and more. Given the right amount of time, one could create a SysML model that describes Monopoly using all of the pillars and diagrams of SysML. However, for the purposes of this demonstration, many, but not all, of the possible modeling constructs are used.

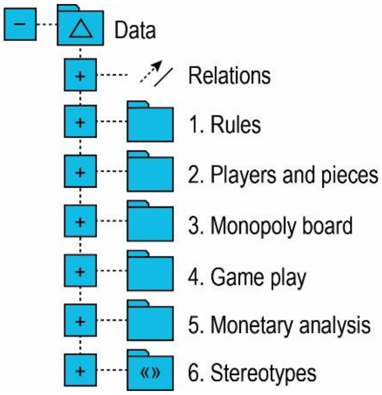

Figure 1. Monopoly model containment tree

The structure of the SysML model of Monopoly, is intended to analogize that of a model of a real system. In other words, each section, or package, of the model represents a different concept or pillar that one would likely represent in a model of an actual system. For instance, most SysML models capture requirements, block definition representing the structure or architecture of the system, activity diagrams to describe the behaviors of the system, and more. Refer to Figure 1, a view of the containment tree of the Monopoly model. The Monopoly model was developed so that different aspects of the game of Monopoly represent different pillars of SysML. These relationships are discussed in detail below. Samples of Requirements, Block Definition, Activity, and Parametric elements corresponding to concepts in the game are provided, and discussion of potential future work for Use Case, Internal Block, Sequence, and State Machine elements follows.

Requirements

In the case of a game, the requirements (for play, game set-up, winning or losing, etc.) are found in the rulebook. Rules can be modeled in the same fashion as technical requirements, as demonstrated in Figure 2, which is an example requirements diagram that describes how the initial money provided to each player shall be divided. In this case, Rules 1.1 through 1.7 are derived from Rule 1, as indicated via the <<deriveReqt>> stereotype on the relationships. It would be instructive to the learner, and typical of a real project discussion, to consider which SysML requirement relationships should be used in the project and under which circumstances. In this case, one may consider applying the <<Refine>> relationship as an alternative and compare the options with project stakeholders to understand why one relationship type is preferred over another. Even requirements numbering could be evaluated by the learner to assess whether one numbering approach works better than another, and the applicable circumstances to which each approach applies.

Figure 2 also illustrates how different cross-cutting modeling techniques can relate various SysML pillars to each other. Figure 2 relates the act of providing the starting money to each player (an activity) to the corresponding requirement being satisfied: a common modeling pattern used to demonstrate all requirements have a specified function, and all functions relate to a specified requirement. Similarly, Figure 2 describes a block that defines the architectural element that satisfies the specified requirement; another modeling pattern. In a real engineering model, every requirement should be satisfied by some other element in the model. In the Monopoly model, it is

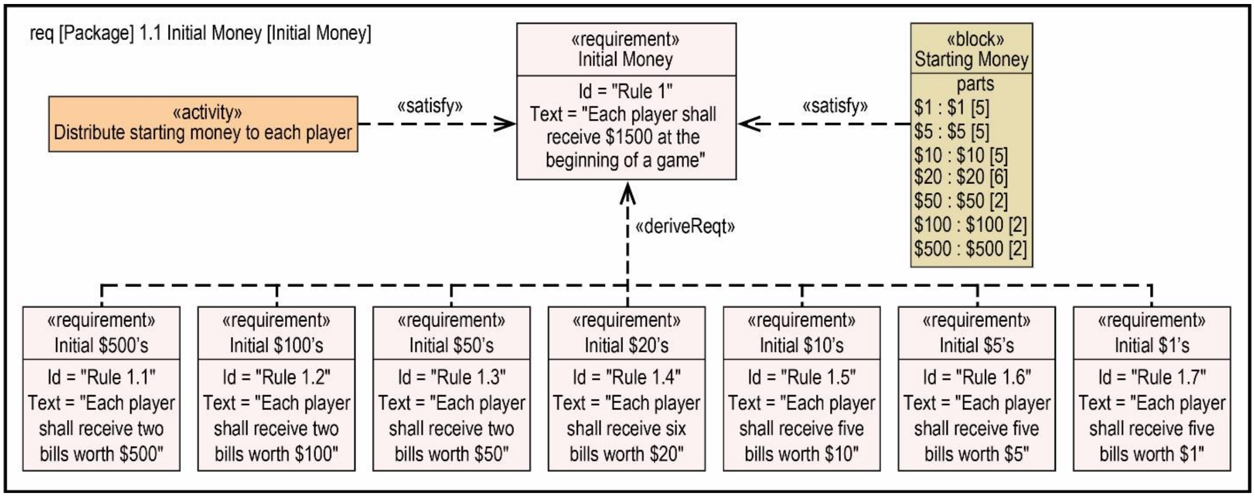
req [Package] 1.1 Initial Money [Initial Money]
«activity»
Distribute starting money to each player
«satisfy»
«requirement»
Initial Money
Id = "Rule 1"
Text = "Each player shall receive $1500 at the beginning of a game"
«satisfy»
«block»
Starting Money
parts
$1 : $1 [5]
$5 : $5 [5]
$10 : $10 [5]
$20 : $20 [6]
$50 : $50 [2]
$100 : $100 [2]
$500 : $500 [2]
«deriveReqt»
«requirement»
Initial $500's
Id = "Rule 1.1"
Text = "Each player shall receive two bills worth $500"
«requirement»
Initial $100's
Id = "Rule 1.2"
Text = "Each player shall receive two bills worth $100"
«requirement»
Initial $50's
Id = "Rule 1.3"
Text = "Each player shall receive two bills worth $50"
«requirement»
Initial $20's
Id = "Rule 1.4"
Text = "Each player shall receive six bills worth $20"
«requirement»
Initial $10's
Id = "Rule 1.5"
Text = "Each player shall receive five bills worth $10"
«requirement»
Initial $5's
Id = "Rule 1.6"
Text = "Each player shall receive five bills worth $5"
«requirement»
Initial $1's
Id = "Rule 1.7"
Text = "Each player shall receive five bills worth $1"

Figure 2. Sample requirements diagram describing rules of the game

not always possible to show, through some measurement or action, that a rule has been satisfied. However, there still are several cases in which this modeling technique can be practiced, as described above.

Requirements could also be developed to capture constraints of the game, just as a system must perform under a variety of operational constraints. For example, one rule in Monopoly could be that there shall be a certain finite quantity of money circulating throughout the game at a given time. This requirement could be satisfied by a constraint property representing the total amount of money in the game, which could be found by summing the total value that each player and the bank possesses.

Therefore, the rules of a game serve as an appropriate substitute for requirements in the scope of the game-system analogy. While the rules of games, Monopoly in particular, are not always extremely complex, there are a lot of opportunities to discuss subtleties, preferences, explore alternative modeling approaches, and create meaningful relationships between the rules of a game and other game characteristics. This process of modeling the rulebook of Monopoly then creates an excellent environment to practice modeling real requirements and refining modeling approaches to integrate them within the rest of the system model.

Block Definition

When modeling a game, the ñstructureò han one way and at multiple can be
levels of detail, just as a system may utilize numerous variations of ñstructu
during the lifecycle (physical, functional, logical, organizational, etc.). The physical board and the
illustrations or ñspacesò on it contribute to
ñtokensò should also be considered within the gam
Figure 3 defines each of these aspects as bei

There are multiple ways the context described in Figure 3 could be represented, and it would be instructive to the learner to explore these options and the rationale for choosing one approach over another. Ideally, the learner could develop an intuition to know whether they should model with one technique over another, according to their overall modeling objective - be it simulate a game

given the option of working in a modeling tool or simply drawing their activity diagrams on paper. After 15 minutes of development time, the participants submitted their drawings or sample model to the assessment organizer to review as a group.

The assessment organizer asked each group to review their model with the group and discuss why they took their selected modeling approach. Following the brief model overview, the organizer invited the other participants to ask questions or provide feedback to the presenting team. Each of the three models covered the steps to set up and play the game, but each team added details in different areas: the first group detailed the game set-up processes; the second group elaborated the VWHSV WKDW PDNH LQ D SOD\HU¶V WXUQ DQG WKH WKLUG JU counters for dice values, board position, and number of turns. This model variety facilitated a rich discussion between the participants, which included questions about the SysML notation, correct application of the language, and use of the modeling tool. At the conclusion of the discussion, the organizer asked the participants to again assess their confidence in modeling activities with different levels of abstraction, and evaluate how confident they were, with more time to do exercises similar to this, that their understanding of SysML would increase.

At the end of the feasibility assessment, the organizer shared the premise of this paper with the participants, and invited discussion about the value of the exercise they had just completed with this context. The participants noted that the majority of their discussion had focused on comparing modeling approaches and model purpose, and there was very little discussion of the subject matter, Monopoly. The organizer asked the participants to describe what they did or did not like about the approach, and gave an opportunity to express any additional comments, questions, or concerns. Finally, the organizer asked participants how likely they would be to recommend this hypothesis IRU IXUWKHU VWXG\ RQ D VFDOH IURP ³QRW DW DOO OLNHO\´

Feasibility Assessment Results

Of the six participants, one had never modeled before, three rated themselves as novice practitioners, and two rated themselves as seasoned practitioners. The participants had taken an average of 2.3 SysML courses previously. Five of the six participants reported an increase in confidence in their ability to model activities with different levels of abstraction after conducting the exercise, while the sixth participant (self-designated a seasoned practitioner) reported they remained extremely confident both before and after the exercise. When asked to assess their confidence that their understanding of SysML would increase after further exercises similar to this IHDVLELOLW\ DVVHVVPHQW WZR SDUWLFLSDQWV LQGLFDW UDWLQJ WKUHH VDLG WKH\ ZHUH ³IDLUO\ FRQILGHQW´ W highest rating), and one said was ³H[WUHPHO\ FRQILGHQW´ WKH On average, the participants rated their likelihood to recommend the hypothesis for further study a 4.25, on a scale from 1 to 5.

Participants cited the ease of understanding the subject matter and its potential to be described in varying levels of detail as aspects of the approach they liked. Most comments from participants regarding what they did not like about the approach were related to the very short nature of the exercise and the fact that it did not include any instruction prior to the group exercise. One participant noted the difference between modeling a subject one understands very well, compared to the real world, where projects are working to define a system that has never been created before. The individual commented: ³Modeling a known system can help you learn a tool and SysML, but

LW ZRQ¶W ZRUN UHDOO\ KHOS \RX OHDUQ KRZ WR HQJLQHHU DQ /V GLIILFXOW WR WHDFK´

While this feasibility assessment was not conducted in a rigorous scientific manner, the authors believe the effort did establish that the local MBSE community at NASA GRC is supportive of further research and application of this approach.

Forward Work and Future Applications

In each of the existing Monopoly samples, additional detail could be added to further develop more advanced modeling skills. One could capture more attributes of the game or enhance traceability between elements. This increase in detail would aid modelers in developing skills in properly abstracting concepts and maintaining consistency and traceability between levels of abstraction. Additional value properties and constraints could be added to enable game simulation, and thus develop skills in setting up more complex analyses, or interfacing the system model with other mathematical modeling tools. Report templates or other communication tools could be written to enable the outcome of such simulations to be generated from the model and transformed into a more traditional document summary.

INCOSE could host tool vendor, chapter, and/or university demonstrations or competitions, where teams would showcase their approach to modeling a game and discuss challenges and benefits of specific modeling approaches or tool capabilities. Should multiple teams address the same game, alternative modeling approaches would more readily be assessed and the lessons learned applied to real world modeling problems.

Organizations could give different teams different objectives for building the model, and compare GLIIHUHQFHV LQ PRGHOLQJ DSSURDFKHV WR KHOS OHDUQHU a model that is built with the intent to simulate game play and the accumulation of wealth may devote more precision and effort to the behavioral language elements, such as state machines or activities, than a model built to describe the rules of the game.

Lastly, game designers could consider developing models of their games tied to relevant simulations, to test their selected arrangement of game mechanisms. Variants of the mechanisms and attributes could be simulated to more quickly arrive at the most engaging balance of complexity, with reduced reliance on human game testers and their subjective feedback.

For the game geeks among systems engineers, perhaps models may be a more instructive way to learn both the rules of the game and winning strategies than traditional paper instructions (documents!).

Acknowledgements

Monopoly subject matter used with permission. Hasbro does not sponsor or endorse the contents of this paper. The authors would like to recognize the NASA GRC MBSE Working Group for their technical review of this paper and willingness to participate in a feasibility assessment of this modeling approach. The authors also thank Kyle Wolff for applying his love for games to the review of this paper.