# Benefits of Realist Ontologies to Systems Engineering[1]

Eric C. Merrell [1,2], Robert M. Kelly[2], David Kasmier[2], Barry Smith[1,2], Marc Brittain[3], Ronald Ankner[3], Evan Maki[3], Curtis W. Heisey[3], and Kevin Bush[4]

[1] *Department of Philosophy, University at Buffalo, 135 Park Hall, Buffalo, NY, USA*
[2] *National Center for Ontological Research Lab, University at Buffalo, 135 Park Hall, Buffalo, NY, USA*
[3] *Lincoln Lab, Massachusetts Institute of Technology, 244 Wood Street, Lexington, MA, USA*
[4] *PM UAS, Building 5681Wood Road, Redstone Arsenal, AL 35898*

**Abstract**

Applied ontologies have been used more and more frequently to enhance systems engineering. In this paper, we argue that adopting principles of ontological realism can increase the benefits that ontologies have already been shown to provide to the systems engineering process. Moreover, adopting Basic Formal Ontology (BFO), an ISO standard for top-level ontologies from which more domain specific ontologies are constructed, can lead to benefits in four distinct areas of systems engineering: (1) interoperability, (2) standardization, (3) testing, and (4) data exploitation. Reaping these benefits in a model-based systems engineering (MBSE) context requires utilizing an ontology's vocabulary when modeling systems and entities within those systems. If the chosen ontology abides by the principles of ontological realism, a semantic standard capable of uniting distinct domains, using BFO as a hub, can be leveraged to promote greater interoperability among systems. As interoperability and standardization increase, so does the ability to collect data during the testing and implementation of systems. These data can then be reasoned over by computational reasoners using the logical axioms within the ontology. This, in turn, generates new data that would have been impossible or too inefficient to generate without the aid of computational reasoners.

**Keywords**

BFO, Interoperability, MBSE, Ontological Realism, Ontology, Ontology Based Systems Engineering, Standardization

## 1. Introduction

Ontologies are controlled vocabularies comprised of terms and relations with formal definitions, and they have become increasingly widespread in data organization and management. The precise and controlled nature of the content of the ontology enables different organizations, people, and data to become systematically ordered, understood, and utilized, respectively. Moreover, the logical definitions within the ontology allow inferences over the data to be made computationally. Basic Formal Ontology (BFO) is one of the most successful top-level ontologies. Its widespread success in the biomedical domain is well-known [12] and, through sponsorship from the U.S. military, BFO recently became an

OntoCom 2021: 8th International Workshop on Ontologies and Conceptual Modeling, held at JOWO 2021: Episode VII The Bolzano Summer of Knowledge, September 11-18, 2021, Bolzano, Italy
EMAIL: ericmerr@buffalo.edu (E. Merrell); rkelly2@buffalo.edu (R. M. Kelly); heisey@ll.mit.edu (C. W. Heisey)
ORCID: 0000-0002-0193-3174 (A. 1); 0000-0002-6847-4726 (A. 2)


[1] DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

This material is based upon work supported by the Department of the Army under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of the Army.

ISO standard [13]. BFO's rigorous development and testing have made it an industry standard upon which interoperable ontologies can be built to efficiently discover, organize, and utilize data. These benefits are becoming increasingly vital to the success of developing autonomous AI into systems.

The U.S. Department of Defense (DoD) and Intelligence Community (IC), among many other organizations, are increasingly utilizing ontologies to organize, exploit, and enhance the sharing of their data[32]. At the same time, the DoD also heavily utilizes the systems modeling language (SysML) to construct systems models for a wide variety of military projects. Using ontologies alongside SysML models in such a way that the ontology enhances the SysML model is therefore crucial to the systems engineering process. Doing so would provide interoperability not only between the system models and the ontologies utilized, but also between systems in different domains linked through the ontology. In this paper, we hope to both gesture at the beginning of a process for how to use ontologies in model-based systems engineering (MBSE), as well as elucidate the various benefits of doing so.

In essence, we suggest beginning with an established upper- or mid-level realist ontology, such as BFO or one of the Common Core Ontologies (CCO), and then extending it in tandem with the construction of a SysML design. BFO and CCO are examples of *realist ontologies*, meaning that their asserted classes are intended to represent reality according to our best scientific understanding of it [15][16]. Abiding by realist principles is vital to maximizing interoperability. Following Delgatti's distinctions between a modelling language, a modelling method, and a modeling tool [14], we argue that incorporating realist ontologies into the systems engineering process adds a modeling method that will increase interoperability, scalability, and clarity within that process.

## 2. Background

MBSE already benefits greatly from modeling languages such as SysML. SysML is a useful tool for constructing dynamic models of systems and breaking down the functionality of each part of the system, how they relate to other parts, and specific requirements of each part of the system [14]. Ontologies have also been shown to be useful in a variety of ways to the systems engineering process, such as in establishing standards and facilitating common understanding [29]. Ontologies, which are often expressed in the Web Ontology Language (OWL), are a useful tool for modeling data [26]. As the data required to develop complex systems become available in increasingly massive amounts and complexity, ontologies are indispensable tools for organizing and exploiting that data. We contend that abiding by the principles of ontological realism maximizes the benefits of ontology when applied to MBSE. In this section, we give a brief appraisal of the capabilities of each of these tools before moving on to how they complement each other in the process of designing systems.

### 2.1. Model-Based Systems Engineering

SysML is an extension of the Unified Modeling Language (UML) and was developed in order to help systems engineers to model complex systems more effectively [3][14][22][23][29]. The main benefit of using modeling languages like SysML in the systems engineering process is that it centralizes the information about a system. From this centralized store of information, relevant specification documents can be generated and different views of the system model can be adopted, which cuts out the need to manually check to make sure the various specifications and designs of the system are being developed or changed uniformly. SysML and UML contain multiple views and artifacts that capture information about a system in a hierarchical fashion. Moreover, they provide traceability between information in the different views to further increase understanding [7]. SysML provides ways to model entities that permits describing those entities using a variety of attributes, parts, or other data. This provides an excellent way for systems engineers to model systems and then test the models even before prototypes of the system are constructed, since changes made in one part of the model will be reflected everywhere.

However, while tools like SysML and the MBSE approach in general have solved the problem of compiling system information together in a unified model, there is still the difficulty of unifying the models themselves. In the context of military operations, for example, a multitude of systems, designed and produced by different organizations, might need to work together. Consequently, as systems become more complex, and as the need for interoperability among these complex systems (sometimes developed by different organizations) arises, a pressing issue becomes how we can facilitate that interoperation. To facilitate this interoperation, we turn to ontology.

## 2.2. Ontology

Ontologies, recall, are controlled vocabularies, and their central usefulness is their ability to unify heterogenous data within and across domains. Data tagged with an ontology provides a common way to record and talk about the data, making it interoperable between different organizations and institutions that may have different (or incompatible) ways of managing, organizing, or referring to that data [8] [15]. Moreover, the ontology helps to codify a precise language, which increases clarity of communication between users of the ontology and thereby reduces miscommunication.

One kind of ontology is a realist ontology, that is, one that has been built according to the methodology of *ontological realism*. According to ontological realism, the terms in an ontology ought to represent the entities that exist in reality according to our best scientific understanding of the world. This can be contrasted with conceptualist approaches to ontology building, according to which terms in an ontology represent our concepts of the world rather than the world itself [16][31].

One of the major successes in the ontology world has been the top-level, realist ontology BFO [17]. The realist approach to BFO's development has allowed it to endure where other ontologies have become obsolete. This is mainly because *reality* is the standard for the development of realist ontologies, and reality is the same for everyone and it is free of contradictions. Alternatively, for conceptualist ontologies the concepts and models that we use can and often do change – sometimes with significant speed. Moreover, because reality is the same for everyone, BFO is able to be utilized and extended into any domain of interest. Where there are *disagreements* about how reality is – and there are many such disagreements – contentious classes are asserted only when needed and are revised if scientific consensus emerges or changes.

If principles of realism are not followed, reality is no longer the standard for asserting classes. As such, the common ground for constructing ontologies is lost, which leads to the "siloing" of the data organized by those ontologies. In turn, this prevents the easy sharing and understanding of those data and, somewhat ironically, undermines one of the main motivations for building and using ontologies at all. In the rest of this section, we give a brief overview of BFO and some of its important mid-level extensions before discussing the compatibility between ontology and the current military enterprise architectures in Section 3.

### 2.2.1. Basic Formal Ontology (BFO)

The backbone of BFO is a taxonomic hierarchy of terms organized into type-subtype (or 'is_a', read as 'is a subtype of') relations. The definitions that produce such a hierarchy are called 'Aristotelian definitions', and they take the form of **A** *is_a* **B** *that Cs* (where '**A** is_a **B**' states the type-subtype relation and 'that Cs' captures the differentiating features that demarcate the subtype from both the parent class and any other of its subtypes). A toy example might be: **Person** *is_a* **Animal** *that is rational*. This tells us that **Person** is a subtype of **Animal**, and that being rational is what demarcates **Person** from other kinds (subtypes) of **Animal**. Because the definitions themselves logically imply a hierarchical structure, conflicts between the meaning and usage of the terms in the ontology can be avoided. Moreover, because the terms and definitions in BFO are determined by attempting to model reality on the basis of our best sciences [16], BFO can be extended to any domain that has an interest in collecting data about reality – virtually any domain. It was partly for this reason that BFO was sponsored to be an ISO standard for top-level ontologies [13].

One intended aim of BFO is to provide a framework that can all any entity to be represented. That is, anything that exists can find a place somewhere within the framework of BFO and its extension ontologies. This approach has allowed BFO to be useful in a variety of domains, including the biomedical, intelligence, and industrial domains [12][18][19].

To aide in this endeavor, mid-level ontologies extending BFO, such as those in CCO, have been developed in order to provide a more robust framework for developing domain-level ontologies [20][21]. BFO is high-level, and because of this it is very small. Yet, sometimes many more specific types of entities within a wide variety of domains need to be captured. Instead of reinventing the wheel every time an ontology is needed for a new domain, mid-level ontologies provide a more robust, in-depth framework from which to draw.

### 2.2.2. Ontological vs. Object-Oriented Class Hierarchies

The backbone of an ontology is a taxonomic class hierarchy, which can resemble an object-oriented (OO) class hierarchy in many respects. However, the class hierarchies developed in OO should not be confused with class hierarchies in ontologies. The former act as a template from which any number of instances can be (though need not be) created [8], whereas a class hierarchy in an ontology ought to represent the kinds of entities that exist in the world (usually in some specified domain) and how they relate to one another [15]. An example OO class hierarchy might be two derived classes "Airplane" and "Automobile" that inherit from a base class "Vehicle." Instances of these classes in the program can be created because the classes have been generated *prior to their instantiation.* This contrasts with realist ontologies, which assert ontological classes like **Airplane** or **Automobile** only if there are instances of those classes in reality. This difference is illustrated in Figure 1 below.

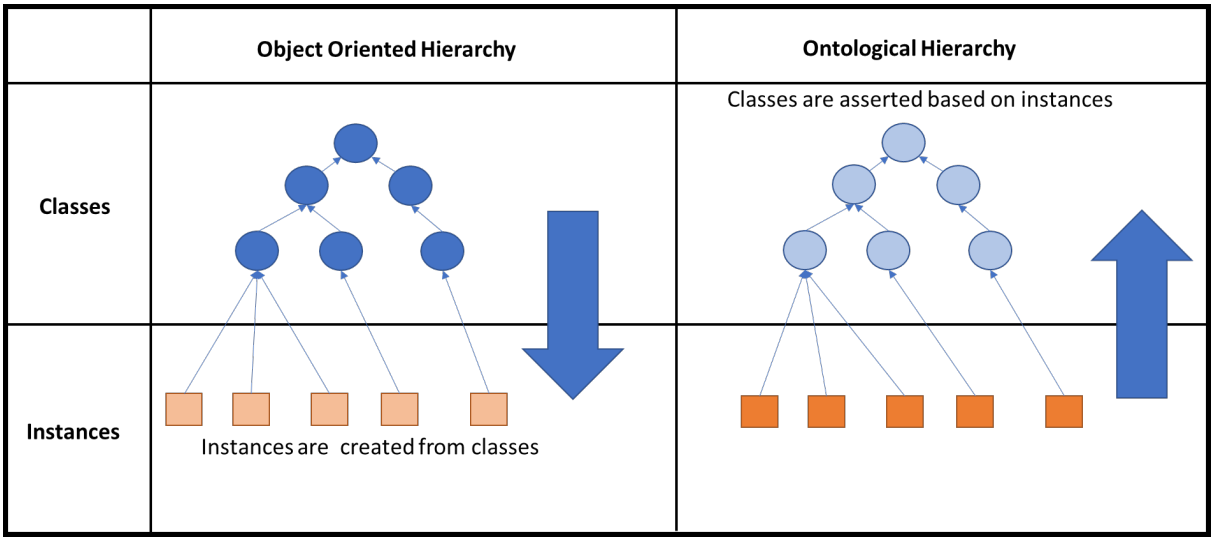


**Figure 1:** Relationship between classes and instances in OO and ontological hierarchies

### 2.2.3. Ontological Axioms and Reasoning

The classes in an ontology are arranged hierarchically using axioms, and the relations in an ontology can be asserted either as axioms between classes or as relations between multiple instances, or between instances and classes [8]. In order to be most useful, however, the ontology has to be machine readable. That is, computers must be able to understand and effectively reason over the data that is tagged with the ontology.

In addition to the benefit of providing a common set of terminology for humans, the computational aspect of ontologies makes them vital in an age where the amount of data that is being generated is increasing exponentially, including in the intelligence and military domains: "The Department of Defense (DoD) and the intelligence community at large (IC) rely on large, and ever-growing amounts of distributed data to perform intelligence-related analysis" [5]. To help with this, the DoD works with several architecture frameworks, such as the DoD Architecture Framework (DoDAF), the Unified Architecture Framework (UAF), and the NATO Architecture Framework (NAF), to help organize the data generated to plan and conduct large scale operations[1][2][4][28]. By using realist ontologies like BFO and CCO, which are compatible with these frameworks, a standard for building and organizing data about models can be established. Going forward, this can help to solve the problem of making systems and system components developed by different organizations easily and effectively interoperable.

## 3. Applying Ontology to Systems Engineering

We propose a "meet-in-the-middle" approach to incorporate ontology into systems engineering. On the one hand, established upper-, mid-, and domain-level ontologies can be mined for their terms (and corresponding definitions), which can then be utilized in SysML models. Everything from what is being modeled in SysML to the parts of the system being designed, the relations to other entities outside the model, and even the kind of model itself can be represented using a consistent terminology provided and maintained by a BFO-conformant ontology. This approach consists of four steps. However, depending on the goals and desires of the project, not all steps need to be taken in order to reap the benefits of ontology. That said, the greater number of steps completed, the better the benefits.

The first step in using an ontology to enhance systems engineering is to identify an ontology or suite of ontologies that capture the kinds of entities that need to be incorporated into the system. As noted, BFO and its mid-level extensions in CCO can be applied to any domain of interest. Accordingly, there are a variety of domain-level ontologies built for use in the military and industrial domains, among others, that can be utilized. If the specific entities that the system deals with are not captured in any ontology, existing ontologies can be extended by adding the needed classes that capture those entities.

After the ontology has been constructed, the second step is to model the system in SysML using the vocabulary from the ontology to designate the entities in the model. This includes the physical components of the system as well as the attributes, capabilities, roles, and functions that those physical components have. Maintaining this semantic consistency from the ontology will ensure that other systems models will be using terms in a consistent way, which will make it easier for engineers to understand each other's models and designs.

The third step is to then use the ontology to tag data collected about the system. Because the same terms that are in the ontology are used to construct the system model, it becomes an easy task to use that ontology to collect and share data about the system.

The third step enables the fourth, which is to use the axioms in the ontology to reason over the data. Tagging the data with the ontology involves structuring that data in some language that is machine readable. One popular means of doing so is to express the ontology and tagged data using the Web Ontology Language 2 (OWL 2), which is based on the Resource Description Framework (RDF). While other languages exist that can be used for such a purpose, the important requirement is that the language be capable of expressing decidable description logic [27]. OWL-RL and OWL-DL (profiles of OWL) enable this kind of reasoning [30]. Thus, modeling systems using the vocabulary from a widely used ontology will do two things. First, it increases the understanding of individuals working on the system. Second, it ensures that data about the systems that is tagged with that ontology will be interoperable with other data that has been tagged with the same or interoperable ontologies.

## 4. Discussion

So far, we have seen two sets of tools: SysML and systems models, on the one hand, and OWL and ontologies, on the other. The question we now turn to is: How can OWL and ontologies be utilized in

the systems engineering process? Ontologies have already been used to great effect in systems engineering. Indeed, the term 'ontology-based systems engineering' (OBSE) has already been coined in the field [24], and further work to integrate SysML and OWL is underway [6][9][10]. However, while the benefits of OBSE have already been observed, we argue that there are four main areas where ontologies that conform to the principles of ontological realism (such as BFO and its extensions in CCO) can enhance the systems engineering process beyond what has already been accomplished. These four areas are:

1. Interoperability: links developing systems models to existing knowledge frameworks;
2. Standardization: standardizes vocabulary that enhances cross-domain understanding;
3. Testing: enhances outcomes-based research and testing; and,
4. Data Exploitation: unifies data that agents in a multi-agent system (MAS) can exploit.

We discuss these four benefits in two sections. Interoperability and standardization, while distinct, are closely related and mutually reinforce each other. Similarly, while testing and data exploitation are each distinct benefits of ontologies, each enhances the other in an organic way. Thus, the two subsections below discuss these pairs in turn. We emphasize that throughout the discussion we talk exclusively of the benefits of *realist* ontologies (though for the sake of redundancy we often omit the qualifier 'realist') because realist ontologies are the ones that have been shown to unify data most effectively [12][13][18].

## 4.1. Standardization and Interoperability

By 'interoperability', we refer to two different phenomena: (1) the ability of systems to work with each other, and (2) the ability to share data in an easily understandable and accurate way. We argue that the standardization provided by an ontology will facilitate both kinds of interoperability because increasing shared understanding (the data interoperability) will make it easier to design systems that can work with each other, even if those systems are designed by different organizations.

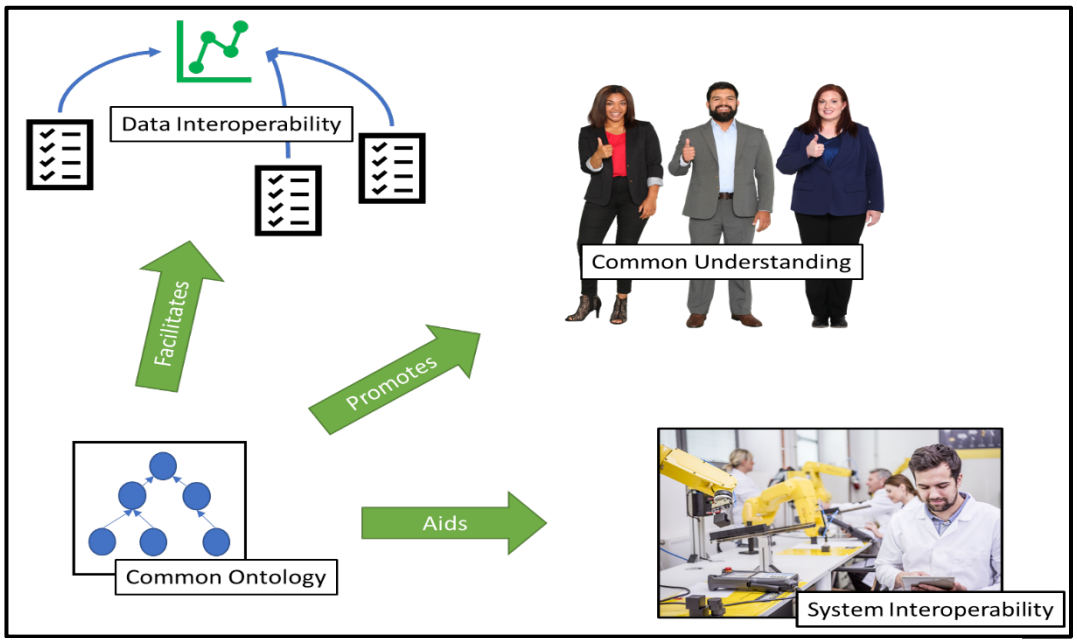


**Figure 2:** Benefits of Realist Ontology to MBSE

Closely related to interoperability is standardization. Standards provide a way for isolated groups and organizations to align their various work, products, goods, and services. In the context of systems building, effective standardization enables systems designed by one organization for a particular purpose to align with systems designed by other organizations for other purposes. Thus, standardization can be seen as a necessary component for large-scale interoperability and long-term scalability. We do not argue here that creating and using standards guarantees interoperability, only that standardization seems to be a necessary condition for achieving widespread system interoperability. However, other requirements beyond standardization will, of course, be necessary to foster this kind interoperability.

*Semantic* standardization, in contrast to system interoperability, can be achieved by having a common, controlled vocabulary that prevents misunderstandings and miscommunications when data is exchanged and interpreted. For example, if there is ambiguity in what is meant by terms like 'multi-agent system' (MAS) or 'unmanned aerial systems' (UAS), then data collected about these entities will not necessarily be uniform; nor, then, could it be easily integrated. If one organization uses 'UAS' to refer to one kind of system and a second organization uses 'UAS' to refer to a slightly different kind of system, and these organizations also share their data, then the shared data will give rise to bad inferences since there are really two (albeit very similar) systems that are falsely thought to be one kind of thing. Incorporating an ontology into this process eliminates this problem by carefully tracking what is meant by each term and organizing the data about the entities which correspond with those terms accurately. In other words, standardization is one of the necessary means by which interoperability is achieved. These standards ultimately foster greater interoperability between systems.

Interoperability ought to be a paramount feature of systems because the more interoperable a system is, the more easily it can be adapted for use within other contexts or systems. Interoperable ontologies are ones that are compatible with other ontologies in terms of their definitions, relations, and class hierarchies. One benefit of incorporating BFO-conformant ontologies in the systems engineering process is a terminology that is consistent across domains. BFO-conformant ontologies already allow data to be collected and synthesized across military, manufacturing, and biomedical domains, as BFO acts as the hub for the extension ontologies in each of these domains [15]. Similarly, if the data about systems models are interoperable, it is easier to see whether two systems are interoperable or not. (indeed computers could use the ontological axioms to infer this) For example, UASs designed for a military application could potentially be utilized in other contexts and systems – such as biomedical ones – as long as the systems models were interoperable enough to allow such a transition.

Suppose that a UAS is designed to be able to dynamically transport cargo to where they are most needed in a battlespace. This UAS would be equipped with various instruments and fixtures in order to allow it to efficiently carry, load, and unload the cargo. If, however, there was suddenly a new need for medical transport of injured troops, one might desire to modify or outfit these cargo-transport UASs to be able to transport injured warfighters. Doing so would require installing new equipment such as heart monitors, stretchers, and other medical equipment. This requires that fixtures, screws, the dimensions of these equipment, and so forth could all be suitably installed into the UAS.

The extent to which the above modifications are possible is dependent upon the extent to which these various pieces of equipment have been designed for interoperability and modularity in mind. To design a piece of equipment with a high level of interoperability, one is required to consider a wide range of other devices and pieces of equipment with which the new piece of equipment can compatibly operate. Having an ontology can aid in this design process by standardizing the vocabulary that is used to talk about the models themselves. The semantic standard, then, makes it easier both to determine which existing equipment the model will be interoperable with. This is because the vocabulary used to talk about, say, the functions, capabilities, and information associated with that equipment would be standard across models and even across modeling frameworks.

Additionally, the ontologies used to develop the model also serve to organize and axiomatize the data about the models that are produced. This is advantageous for systems designers since computers can use the ontology to reason over the data about large sets of models to determine which equipment is or is not interoperable with another piece of equipment. The ontology enables this by using the logical axioms associated with its classes. We emphasize that even with a semantic standard in place, not every piece of equipment will be (or even should be) interoperable with every other piece of equipment, machinery, or system. However, the semantic standard the ontology provides makes it easier to create

interoperable systems, equipment, devices, or software, and allows computers to efficiently determine the extent to which various systems and system components are interoperable.

## 4.2. Testing and Data Exploitation

Once models have been created, the systems that have been designed need to be tested. This testing could take place using simulations, physical prototypes, or both. However, it is accomplished, testing requires data, and data must be organized and processed in order to be useful. Without an ontology, one must devise a way to collect and process the generated data after the model has been made. Without an ontology, the data that is generated will also only be usable by organizations that have sufficiently similar methods of collecting and organizing that data, which is unlikely without a strong standard. With an ontology, however, the way to organize data collected from the testing of the systems is inherent already within the ontology used to build the system. Moreover, even if an ontology was incorporated after the design of the system during data collection, the ontology would enable any organization with access to that ontology to use that data and generate insights and improvements.

The most prominent example of this is seen in outcomes-based research. Outcomes-based research began in the healthcare domain, and it attempts to understand the outcomes and results of particular practices and interventions within that domain [11]. This is done by combining datasets that look for the relationships between variables and desired outcomes, and then performing metanalyses on the data to determine which parameters produce the desired outcomes most effectively. As Susan Michie and Marie Johnston put it:

> Implementing research findings into healthcare practice and policy is a complex process occurring in diverse contexts… Questions asked with the aim of improving implementation are multifarious variants of "What works, compared with what, how well, with what exposure, with what behaviors (for how long), for whom, in what setting and why?" [[25], p. 1]

In the context of systems engineering, outcomes-based research opens the door to testing which kinds of systems and which models for systems are most effective at achieving particular outcomes. This could even go so far as to reveal which parts of systems are contributing the most to which outcomes, thereby allowing the best pieces taken from various systems to be combined into a more effective one.

For example, suppose one is testing a prototype design for semi-autonomous rovers that can detect mines in a minefield and transmit their location back to a commander. Getting the rovers to sweep a minefield with maximal efficiency, determining which environments the rovers work best in, comparing different rover designs, and testing which kinds of sensors or sets of sensors are best for detecting the mines, and in which environments, requires an enormous amount of data collection over a huge number of tests. If the models for these rovers were built using an ontology, data generated by these tests can be seamlessly organized by the ontology since each system component being tested will already have a place in the ontology. Once the data is organized by the ontology, determining the answers to questions like, "which sensors work best in which environments", become far easier. This stems from the ability of computer reasoners to infer a myriad of useful data using the axioms present in the ontology, which can then be queried as needed.

Thus, ontology enhances the power of data by allowing it to be more seamlessly aggregated from different sources and using that aggregated data to computationally generate further data using the axioms built into the ontology.

## 5. Conclusion

In this paper, we proposed that incorporating BFO-conformant ontologies into the systems engineering process will facilitate the unification of the data that is collected by different groups and organizations, as well as facilitating the development of interoperable systems. BFO-conformant

ontologies abide by principles of ontological realism, according to which ontological classes represent reality according to our best scientific understanding of the world. By using reality as a standard, BFO and its extension ontologies provide a way to unify heterogeneous domains under a single framework. The ontology provides a semantic standard that eliminates ambiguities in the data collected, and it also contributes axioms that allow for additional data to be generated. This is especially helpful in comparing and testing the effectiveness of systems using an outcomes-based research method.

## 6. References


[1] Unified Architecture Framework (UAF) Domain Metamodel Version 1.1., 2020. URL: https://www.omg.org/spec/UAF/1.1/DMM/PDF.

[2] DoDAF: DoD Architecture Framework Version 2.02, 2010. URL: https://dodcio.defense.gov/library/dod-architecture-framework/.

[3] SysML Open Source Project, 2021. URL: https://sysml.org/.

[4] G.M. Miranda, J.P.A. Almeida, C.L.B. Azevedo, G. Guizzardi, An ontological analysis of capability modeling in defense enterprise architecture frameworks. Proceedings of the 8th Brazilian Symposium on Ontology Research (ONTOBRAS 2016) (2016) 11–22.

[5] G.M. Miranda, J.P.A. Almeida, G. Guizzardi, & C.L.B. Azevedo, Foundational Choices in Enterprise Architecture: The Case of Capability in Defense Frameworks. IEEE 23rd International Enterprise Distributed Object Computing Conference (EDOC). (2019). 31-40.

[6] H. Graves, Integrating Reasoning with SysML, INCOSE International Symposium. 22 (2012). 2228-2242. doi: 10.1002/j.2334-5837.2012.tb01470.x.

[7] D. Bell, An Introduction to the Unified Modeling Language, 2003. URL: https://developer.ibm.com/articles/an-introduction-to-uml/.

[8] D. Allemang, J. Hendler, The Semantic Web for the Working Ontologist, 2nd Edition. Morgan Kaufmann Publishers, Elsevier. MA, 2011.

[9] T. Hagedorn, M. Bone, B. Kruse, I. Grosse, M. Blackburn, Knowledge Representation with Ontologies and Semantic Web Technologies to Promote Augmented and Artificial Intelligence in Systems Engineering. *INCOSE.* (2020).

[10] C.W. Heisey, M.W. Brittain, D.E. Maki, & K. Bush, "Multi-Agent Systems Collaborative Teaming (MASCOT) Definition Process to Create Specifications for Multi-Agent System (MAS) Development, 25th International Command and Control Symposium (2020).

[11] Jefford, M., Stockler, M., & Tattersall, M. "Outcomes research: What is it and why does it matter?" Internal medicine journal. 33 (2003). doi: 110-8. 10.1046/j.1445-5994.2003.00302.x.

[12] B. Smith, M. Ashburner, C. Rosse, et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration, Nat Biotechnol 25 (2007) 1251–1255 doi: https://doi.org/10.1038/nbt1346

[13] ISO/IEC PRF 21838-2.2 Information technology — Top-level ontologies (TLO) — Part 2: Basic Formal Ontology (BFO). URL: https://www.iso.org/standard/74572.html.

[14] L. Delligatti, SysML Distilled: A Brief Guide to the Systems Modeling Language (1st. ed.)". Addison-Wesley Professional. 2013.

[15] R. Arp, Robert, B. Smith, & A.D. Spear, Building Ontologies with Basic Formal Ontology, Cambridge, MA: MIT Press. 2015.

[16] B. Smith, W. Ceusters, Ontological Realism: A Methodology for coordinated evolution of scientific ontologies, Appl Ontol. 5.3-4 (2010) 139-188. doi: 10.3233/AO-2010-0079.

[17] Basic Formal Ontology 2.0 (BFO). URL: https://github.com/BFO-ontology/BFO2/blob/master/ontology/src/bfo.owl

[18] B. Kulvatunyou, E. Wallace, D. Kiritsis, B. Smith, C. Will, (2018). The Industrial Ontologies Foundry Proof-of-Concept Project, Springer Nature Switzerland AG. (2018). doi: https://doi.org/10.1007/978-3-319-99707-0_50.
[19] D. Limbaugh, D. Kasmier, R. Rudnicki, J. Llinas, & B. Smith, Ontology and Cognitive Outcomes, ArXiv, abs/2005.08078. (2020).
[20] Common Core Ontologies, URL: https://github.com/CommonCoreOntology/CommonCoreOntologies
[21] R. Rudnicki, Overview of Common Core Ontologies Prepared by: CUBRC, Inc. 4455 Genesee St., Buffalo, NY 14225. (2019).
[22] SysML v2, URL: https://github.com/Systems-Modeling/SysML-v2-Release
[23] OMG Unified Modeling Language (UML) version 2.5.1. URL: https://www.omg.org/spec/UML/2.5.1/PDF
[24] L. Yang, K. Cormican, M. Yu. Ontology-based systems engineering: A state-of-the-art review, Computers in Industry. 111 (2019) 148-171. doi:10.1016/j.compind.2019.05.003.
[25] S. Michie, M. Johnston. Optimising the value of the evidence generated in implementation science: The use of ontologies to address the challenges, Implementation Science. 12.131 (2017) 1-4. doi: https:doi.org/10.1186/s13012-017-0660-2.
[26] OWL 2 Web Ontology Language Document Overview (Second Edition). URL: http://www.w3.org/TR/2012/REC-owl2-overview-20121211/.
[27] RDF 1.1 Concepts and Abstract Syntax. URL: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/.
[28] NATO Architecture Framework Version 4, 2018, URL: https://www.nato.int/nato_static_fl2014/assets/pdf/pdf_2018_08/20180801_180801-ac322-d_2018_0002_naf_final.pdf.
[29] A.M. Madni, M. Sievers, Model-based systems engineering: Motivation, current status and research opportunities, Systems Engineering, 21 (2018) 172– 190. https://doi.org/10.1002/sys.21438
[30] OWL 2 Web Ontology Language Profiles (Second Edition). URL: https://www.w3.org/TR/2012/REC-owl2-profiles-20121211/.
[31] B. Smith, Barry, Beyond Concepts: Ontology as Reality Representation, Proceedings of FOIS, (2004).
[32] S. Riley. “Object-Based Production & Activity-Based Intelligence for Cybersecurity,” Linkedin.com (2015). https://www.linkedin.com/pulse/object-based-production-activity-based-intelligence-shawn-riley (Accessed February 21, 2021)