Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems

Chapter · August 2017

DOI: 10.1007/978-3-319-38756-7_4

2 authors:

Michael Grieves
Digital Twin Institute
42 PUBLICATIONS **7,605** CITATIONS
SEE PROFILE

John Vickers
Johnson Space Center
1 PUBLICATION **2,757** CITATIONS
SEE PROFILE

Franz-Josef Kahlen
Shannon Flumerfelt
Anabela Alves *Editors*

Transdisciplinary Perspectives on Complex Systems

New Findings and Approaches

Springer

Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems

Michael Grieves and John Vickers

1 Introduction

Up until fairly recently, the only way to have extensive knowledge of the physical object was to be in close proximity to that object. The information about any physical object was relatively inseparable from the physical object itself. We could have superficial descriptions of that object, but at best they were limited in both extensiveness and fidelity.

Such basic information as dimensions height, width, and length, only became available in the mid-1800s with the creation of the standard inch and a way to consistently measure that inch. Prior to that period of time, everyone had his or her own version of measurement definitions that meant that interchangeable manufacturing was impossible.

It was then only in the last half of the twentieth century, that we could strip the information from a physical object and create what we are calling a Digital Twin. This Digital Twin started off relatively sparse as a CAD description and is becoming more and more rich and robust over the years. While at first, this Digital Twin was merely descriptive, in recent years it is becoming actionable.

What actionable means is that the CAD object is no longer simply a three dimensional object hanging in empty space, time independent. We can now simulate physical forces on this object over time in order to determine its behavior. Where CAD models were static representations of form, simulations are dynamic representations, not only of form but also of behavior.

M. Grieves (✉)
Florida Institute of Technology, Melbourne, FL, USA
e-mail: mgrieves@fit.edu

J. Vickers
NASA MSFC, Huntsville, AL, USA

F.-J. Kahlen et al. (eds.), *Transdisciplinary Perspectives on Complex Systems*,
DOI 10.1007/978-3-319-38756-7_4

One of the previous issues with only being able to work with a physical object is that the range of investigation into its behavior was both expensive and time-consuming. We first had to physically create the object, a one-off proposition. We then had to create a physical environment in which the object was impacted by actual forces. This meant that we were limited to investigating forces and their associated levels that we thought were of concern. Often, the forces would result in destruction of the object, dramatically increasing the expense.

This meant that the first time we actually saw a condition not covered by a physical test would be when the physical object was in actual use. This meant that there were going to be many unforeseen conditions or emergent behaviors that resulted in failures that could result in harm and even death to its users.

Aggregating these objects into systems compounded the problem. Systems are much more sophisticated than simple objects. We need to have better ways to understand these increasingly sophisticated systems.

The idea of the Digital Twin is to be able to design, test, manufacture, and use the virtual version of the systems. We need to understand whether our designs are actually manufacturable. We need to determine the modes of failure when the system is in use. We need all of this information before the physical system is actually produced. This will reduce failures of the physical system when it is deployed and in use, reducing expenses, time, and most importantly harm to its users.

2 Conventional Approaches and Current Issues

The issue with complex systems is not that they are complex. A complex system that performed its tasks flawlessly, always produced the desired results, and gave forewarning that potential failures were likely in the future so that these potential failures could be corrected before they occurred would be a desirable system to have.

But, unfortunately, that is not usually the case with complex systems. The issue is that the systems do not always either perform flawlessly or do not produce the desired results. More importantly, they often fail without warning and can fail dramatically and catastrophically, with a minor issue cascading into a major failure.

It is this aspect of complex systems that is the thorny problem that needs major mitigation and/or resolution.

2.1 Defining Complex Systems

The first task is to define what a system and a complex system are. The definition of a system is taken to be the following [1][1]:

[1]Modified to add that the results could not be obtained from the components individually. While there are much more detailed descriptions of systems and their characteristics (see Ackoff, R. L. [2]) this definition suffices for the points we wish to make here.

> A system is two or more components that combine together to produce from one or more inputs one or more results that could not be obtained from the components individually

Systems can be categorized into three types: simple, complicated, and complex.

Simple systems are just that. The outside observer has no problem in discerning the operation of the system. The system is completely predictable. The inputs are highly visible. The actions performed on those inputs are obvious and transparent. The outputs are easily predictable.

Complicated systems are also completely predictable. The system follows well-defined patterns [3]. The difference between simple systems and complicated systems is the component count. Complicated systems have many more components. Complicated systems are often described as intricate. However, the inputs are well known, as are the resulting outputs. The connection between components is linear and straightforward. A mechanical watch would be a representative example of a complicated system.

Complex systems are in a different class of systems entirely. There is not very good agreement as to how to even describe a complex system. In fact, there is little agreement on the term "complexity" [4].

Complex systems have been characterized as being a large network of components, many-to-many communication channels, and sophisticated information processing that makes prediction of system states difficult [5].

We would contend that complex systems have a major element of surprise, as in "I didn't see that coming." That surprise is generally, although not always, an unwelcome one.

2.2 Complex Systems and Associated Problems

While man-made complex systems can be considered relatively new phenomena, the issue of complex systems as it relates to serious and catastrophic problems goes back decades. The first major discussion of this issue is often considered to be Perrow's seminal work on the inherent dangers of complex systems, *Normal Accidents* [6]. Perrow defines the difference between complex and complicated systems.

Perrow defines complexity in terms of interactions (p. 78). He defines linear interactions as "those in expected and familiar production or maintenance sequence, and those that are quite visible even if unplanned". He defines complex interactions as, "those of unfamiliar sequences, or unplanned and unexpected sequences, and are either not visible or not immediately comprehensible." Perrow also uses "tightly-coupled" to describe complex systems and "loosely-coupled" to describe complicated systems.

In other terminology, Perrow's linear interactions and loosely coupled connections would be characteristic of a complicated system, whereas his complex

interactions and tightly coupled connections would be the characteristic of a complex system. Perrow's claim, which he supports with numerous examples, is that complex systems lead quite naturally to the idea of "normal accidents".

Perrow's laundry list of normal accidents, better described as disasters, span the land, air, and sea. Examples include the Three-Mile Island nuclear reactor meltdown, the 1979 DC-10 crash in Chicago, and maritime disasters.

The common thread to Perrow's examples is the human element in interacting with complex systems, which makes these systems sociotechnical systems. It takes primarily two forms: human inconsistency, both deliberate and accidental, in following rules, processes, and procedures and a lack of sensemaking, i.e., the ability to make sense out of the inputs and stimuli that are being presented to the human.

The ubiquitous presence of computers that was not present in Perrow's day can prevent human inconsistency that is accidental or even intentional. Computers forget nothing and perform processes over and over again without any deviation. Computers can even go a long way in preventing the deliberate, error-causing human acts, by sensing what the system's state should be and comparing it against what it is, and raising an alarm if the two do not match.

One of the examples that Perrow uses is a fire door being propped open when it should not have been. In today's environment, a sensor would have been placed on that door and triggered an alarm in the computer that the door was open when it should be closed. If nothing else, humans are incredibly creative in trying to bypass such systems. However, the use of computers would and does dramatically decrease the numbers of even deliberate human intent to not follow procedures and processes.

The other source of human interaction problems, sensemaking, has played a role in major system disasters. This is an area that has been explored quite well by Karl Weick. The core issue here is that humans often do not do a good job in making sense of the information streaming at them, especially in stressful situations. As a result, they sometimes do not make the right decisions and not infrequently make exactly the wrong decision in the heat of a crisis.

Weick has his own list of disasters where this is happened, including the NASA disasters, Challenger and Columbia [7], and, what can be classified as a System-of-Systems failure, the air accident involving two 747s colliding on Tennerife in the Canary Islands [8].

This is a much more difficult issue to address, as computers do not do any sensemaking. They simply execute what they have been programmed to do. However, as will be discussed below, what computers can do is perform simulations before the system is deployed to both determine how the system reacts in a wide variety of conditions and train users of the system under abnormal conditions that they might face. We also will propose that simulated systems might "front-run" in real time developing conditions in order to assist humans in making the correct sense of developing situations and overcoming the biases that can negatively affect human decision making [9].

2.3 Defining Emergence and Emergent Behavior

"Emergence" is too general a term to be of much use. In the dynamic universe in which we live, emergence is a keen sense of the obvious. Emergence of the universe itself has existed since the Big Bang.

Discussion about emergence goes back to at least the times of Aristotle who in Metaphysics introduced the concept that the whole is more than the sum of its parts [10]. The idea of emergence spans a wide, wide spectrum. It goes from the idea that a pile of bricks has the potential emergence behavior of a house [11] to a system that learns and adapts to its environment. Emergence covers the idea of both emergence of form and function or behavior.

"Emergent behavior" is a little better in that it narrows the discussion down to function, as opposed to emergent form, which refers to the physical manifestation of a system. However, as we shall discuss, it still is both too general and too ambiguous to pin down exactly what we are referring to. Is emergent behavior a result of new and unique behavior that has arisen or is it behavior that has been a possibility from inception, but simply has not been recognized [12]?

Much more recently, Holland [13] proposes a taxonomy of emergent behavior. His taxonomy is as follows:

Type 0: Constituent (Non-Emergence)
Type 1: Nominal Emergence
Type 2: Moderated Emergence
Type 3: Multiple Emergence
Type 4: Evolutionary Emergence

Of the five types, only Type 4 would qualify as behavior that results from modifications to the system as it reacts with its environment. It is dynamic emergent behavior. Evolutionary emergence involves a deliberate learning/modification behavior loop that humans exhibit. Evolutionary emergence can also be randomly generated, such as nature uses in evolutionary genetics and humans, often to their detriment, exhibit randomness in times of crisis. The Type 4 systems are actually different systems over time.

In the other four types, the emergent behavior exists from the point that the system is completed and deployed. It is static emergent behavior. It is not that something new creeps into the system. It is that we have not foreseen this behavior. Even where the behavior is a result of human interaction, the system at its inception would have performed in exactly this fashion when presented with these inputs.

The system did not change. We just "didn't see that coming!" It is these first four types of static emergent behavior that we will deal with, static emergent behavior that is built into the system and is unforeseen rather than arises from dynamic system modification or randomness.

While this taxonomy is useful in describing the type of emergent behavior as it increases in sophistication of actions and communications, it does not describe what emergent behavior is desirable and predictable. We cannot rely on the name to give

us this indication. Holland's Type 2b, Moderated Unstable, which results in rapid growth changes and general instability, e.g., explosions, sounds undesirable and probably is in a number of situations. However, this behavior is quite useful in weapon systems.

2.4 Four Categories of System Behavior

Figure 1 characterizes behavior along the lines of desirable and undesirable behaviors. It divides static emergent behavior into two categories: predicted and unpredicted. The predicted and unpredicted categories are also divided into two categories: desirable and undesirable. This gives us four final categories: Predicted Desirable (PD), Predicted Undesirable (PU), Unpredicted Desirable (UD), and Unpredicted Undesirable (UU).

The PD category is obviously the desired behavior of our system. This is the intentional design and realization of our system. In systems engineering terms, it is the requirements our system is designed to meet.

The PU category contains problems we know about and will do something about eliminating. PU problems are management issues. If we know about the PU problems and do nothing about them, that is engineering malpractice. This is the category that expensive lawsuits are made of.

The unpredicted category is our "surprise" category. The UD category contains pleasant surprises. While ignorance is never bliss, this category will only hurt our pride of not having understood our system well enough to predict these beneficial behaviors.

The UU category holds the potential for serious, catastrophic problems. It is this category of emergent behavior that we need to focus on. We need capabilities and methodologies to mitigate and/or eliminate any serious problems and even reduce unforeseen problems that are merely annoying. This is the purpose of the Digital Twin model and methodology.

Finally, we will not abandon emergent form, because that is of great importance in the manufacturing phase and operational phase. However, physical forms have a permanency that behavior does not.

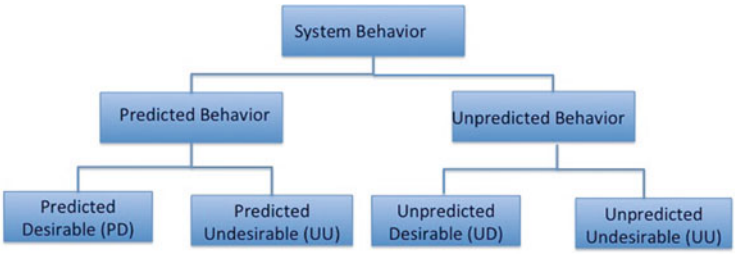

Fig. 1 Categories of system behavior

2.5 *Emergence, Emergent Behavior, and the Product Lifecycle*

Systems do not pop into existence fully formed, exhibiting behavior, emergent or otherwise. Systems get the way they are by a progression through their product lifecycle. This is shown in Fig. 2 [14]. The system's product lifecycle starts at system creation, takes on its physical attributes during the production phase, and then exist as an entity during its operational phase, when its behavior is a result of how the system was created and produced.

During the creation phase, designers create the characteristics and behaviors of the desired system. In terms of systems engineering, this means defining the requirements of the system. It is at this stage that the desirable attributes of the system are defined. Any undesirable behaviors are also identified and counter strategies are created to prevent them from occurring. The issue is that it is easier to enumerate the behaviors that are desired than it is to identify all the possible ways that the system can go wrong.

It is this phase that we should be considering the "ilities" of the system: manufacturability, sustainability, supportability, and disposability. Historically, we have known this. Realistically, we have done a very poor job in this regard. We have operated in a siloed fashion, with system design and engineering performing their tasks of defining the system requirements and verifying and validating them. Design and engineering then throw the plans over the wall to manufacturing. However, this siloing exists even within the design and engineering

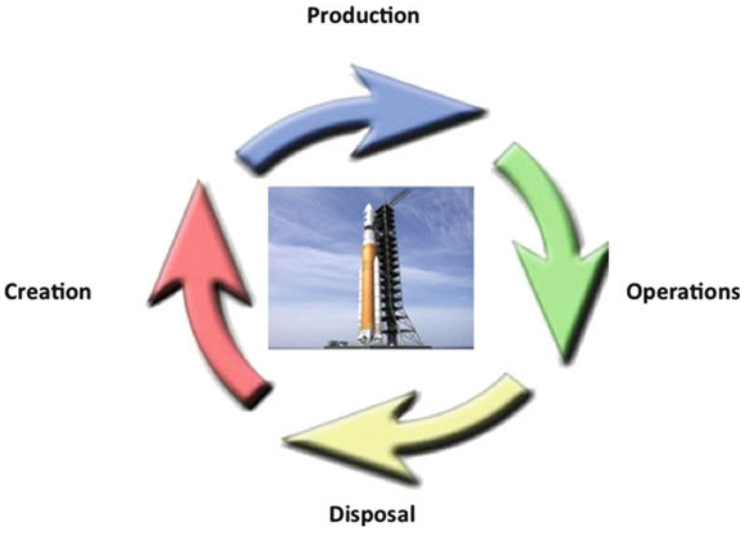

Fig. 2 Product lifecycle—4 phases

function, leading to undesirable behaviors arising because of lack of understanding at interface points between various sub-functions and assemblies.

In the production or manufacturing phase, the system emerges as a physical artifact. It is at this phase that we find whether or not our system design is realizable. We may find out that the manufacturing techniques that we have at our disposal are insufficient to realize the design that we have envisioned. It is at this phase that undesirable behaviors can also start to creep into the system as we change the design to meet the realities of manufacturing.

It is in the operational phase where we find out all our categories of behavior. We obviously would not put into operation those systems that do not meet the requirements that we set out in the creation phase (PD). We also would have eliminated all the known undesirable behaviors (PU). However, given the traditional approach to system development, we would still continue to be bedeviled by unpredicted undesirable behaviors (UU).

The lifecycle ends with the retirement and disposal of the system. This is generally not considered to have relevance to system complexity or emergent behavior. That may indeed be why there is little discussion of it. However, for some systems we might want to rethink that position. Two examples that come to mind are spent nuclear fuel and decommissioned spacecraft in orbit. While we will not dwell on the disposal issue, we will make mention of it when relevant.

We have proposed that Systems Engineering and Product Lifecycle Management highly overlap at a minimum and may in fact be the one and the same [1]. However, as a current practice, they are not generally considered as such.

In theory, Systems Engineering should concern itself with the behavior of the system throughout the entire lifecycle. Product Lifecycle Management (PLM) claims to do just that. As practiced, Systems Engineering has often degenerated into "systems accounting," with systems engineers considering their job done at the end of the create phase [14, 15]. In many organizations, once systems engineers have verified and validated the requirements, they consider the system completed.

3 The Digital Twin Concept

While the terminology has changed over time, the basic concept of the Digital Twin model has remained fairly stable from its inception in 2002. It is based on the idea that a digital informational construct about a physical system could be created as an entity on its own. This digital information would be a "twin" of the information that was embedded within the physical system itself and be linked with that physical system through the entire lifecycle of the system.

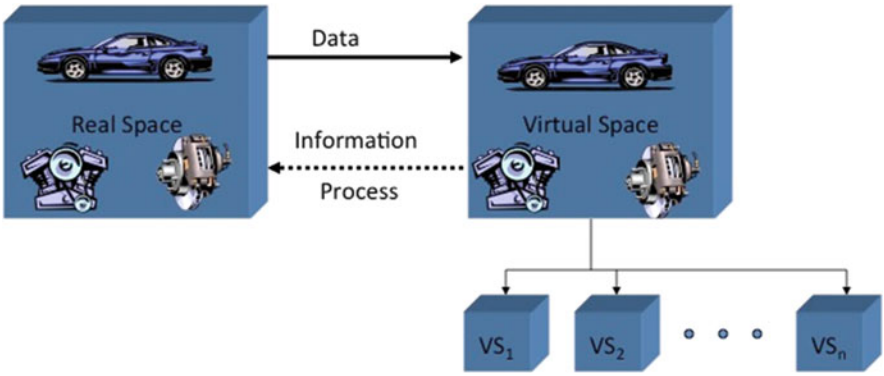

Fig. 3 Conceptual ideal for PLM. Dr. Michael Grieves, University of Michigan, Lurie Engineering Center, Dec 3, 2002

3.1 Origins of the Digital Twin Concept

The concept of the Digital Twin dates back to a University of Michigan presentation to industry in 2002 for the formation of a Product Lifecycle Management (PLM) center. The presentation slide, as shown in Fig. 3, was simply called "Conceptual Ideal for PLM." However, it did have all the elements of the Digital Twin: real space, virtual space, the link for data flow from real space to virtual space, the link for information flow from virtual space to real space and virtual sub-spaces.

The premise driving the model was that each system consisted of two systems, the physical system that has always existed and a new virtual system that contained all of the information about the physical system. This meant that there was a mirroring or twinning of systems between what existed in real space to what existed in virtual space and vice versa.

The PLM or Product Lifecycle Management in the title meant that this was not a static representation, but that the two systems would be linked throughout the entire lifecycle of the system. The virtual and real systems would be connected as the system went through the four phases of creation, production (manufacture), operation (sustainment/support), and disposal.

This conceptual model was used in the first PLM courses at the University of Michigan in early 2003, where it was referred to as the Mirrored Spaces Model. It was referenced that way in a 2005 journal article [16]. In the seminal PLM book, *Product Lifecycle Management: Driving the Next Generation of Lean Thinking,* the conceptual model was referred to as the Information Mirroring Model [17].

The concept was greatly expanded in *Virtually Perfect: Driving Innovative and Lean Products through Product Lifecycle Management* [14], where the concept was still referred to as the Information Mirroring Model. However, it is here that the

term, Digital Twin, was attached to this concept by reference to the co-author's way of describing this model. Given the descriptiveness of the phrase, Digital Twin, we have used this term for the conceptual model from that point on.

The Digital Twin has been adopted as a conceptual basis in the astronautics and aerospace area in recent years. NASA has used it in their technology roadmaps [18]. The concept has been proposed for next generation fighter aircraft and NASA vehicles [19, 20], along with a description of the challenges [20] and implementation of as-builts [21].

3.2 Defining the Digital Twin

What would be helpful are some definitions to rely on when referring to the Digital Twin and its different manifestations. We would propose the following:

Digital Twin (DT)—the Digital Twin is a set of virtual information constructs that fully describes a potential or actual physical manufactured product from the micro atomic level to the macro geometrical level. At its optimum, any information that could be obtained from inspecting a physical manufactured product can be obtained from its Digital Twin. Digital Twins are of two types: Digital Twin Prototype (DTP) and Digital Twin Instance (DTI). DT's are operated on in a Digital Twin Environment (DTE).

Digital Twin Prototype (DTP)—this type of Digital Twin describes the prototypical physical artifact. It contains the informational sets necessary to describe and produce a physical version that duplicates or twins the virtual version. These informational sets include, but are not limited to, Requirements, Fully annotated 3D model, Bill of Materials (with material specifications), Bill of Processes, Bill of Services, and Bill of Disposal.

Digital Twin Instance (DTI)—this type of Digital Twin describes a specific corresponding physical product that an individual Digital Twin remains linked to throughout the life of that physical product. Depending on the use cases required for it, this type of Digital Twin may contain, but again is not limited to, the following information sets: A fully annotated 3D model with Geometric Dimensioning and Tolerancing (GD&T) that describes the geometry of the physical instance and its components, a Bill of Materials that lists current components and all past components, a Bill of Process that lists the operations that were performed in creating this physical instance, along with the results of any measurements and tests on the instance, a Service Record that describes past services performed and components replaced, and Operational States captured from actual sensor data, current, past actual, and future predicted.

Digital Twin Environment (DTE)—this is an integrated, multi-domain physics application space for operating on Digital Twins for a variety of purposes. These purposes would include:

Predictive—the Digital Twin would be used for predicting future behavior and performance of the physical product. At the Prototype stage, the prediction would

be of the behavior of the designed product with components that vary between its high and low tolerances in order to ascertain that the as-designed product met the proposed requirements. In the Instance stage, the prediction would be a specific instance of a specific physical product that incorporated actual components and component history. The predictive performance would be based from current point in the product's lifecycle at its current state and move forward. Multiple instances of the product could be aggregated to provide a range of possible future states.

Interrogative—this would apply to DTI's. Digital Twin Instances could be interrogated for the current and past histories. Irrespective of where their physical counterpart resided in the world, individual instances could be interrogated for their current system state: fuel amount, throttle settings, geographical location, structure stress, or any other characteristic that was instrumented. Multiple instances of products would provide data that would be correlated for predicting future states. For example, correlating component sensor readings with subsequent failures of that component would result in an alert of possible component failure being generated when that sensor pattern was reported. The aggregate of actual failures could provide Bayesian probabilities for predictive uses.

3.3 The Digital Twin Model Throughout the Lifecycle

As indicated by the 2002 slide in Fig. 3, the reference to PLM indicated that this conceptual model was and is intended to be a dynamic model that changes over the lifecycle of the system. The system emerges virtually at the beginning of its lifecycle, takes physical form in the production phase, continues through its operational life, and is eventually retired and disposed of.

In the create phase, the physical system does not yet exist. The system starts to take shape in virtual space as a Digital Twin Prototype (DTP). This is not a new phenomenon. For most of human history, the virtual space where this system was created existed only in people's minds. It is only in the last quarter of the twentieth century that this virtual space could exist within the digital space of computers.

This opened up an entire new way of system creation. Prior to this leap in technology, the system would have to have been implemented in physical form, initially in sketches and blueprints but shortly thereafter made into costly prototypes, because simply existing in people's minds meant very limited group sharing and understanding of both form and behavior.

In addition, while human minds are a marvel, they have severe limitations for tasks like these. The fidelity and permanence of our human memory leaves a great deal to be desired. Our ability to create and maintain detailed information in our memories over a long period of time is not very good. Even for simple objects, asking us to accurately visualize its shape is a task that most of us would be hard-pressed to do with any precision. Ask most of us to spatially manipulate complex shapes, and the results would be hopelessly inadequate.

However, the exponential advances in digital technologies means that the form of the system can be fully and richly modeled in three dimensions. In the past, emergent form in complex and even complicated system was a problem because it was very difficult to insure that all the 2D diagrams fit together when translated into 3D objects.

In addition, where parts of the system move, understanding conflicts and clashes ranged from difficult to impossible. There was substantial wasted time and costs in translating 2D blueprints to 3D physical models, uncovering form problems, and going back to the 2D blueprints to resolve the problems and beginning the cycle anew.

With 3D models, the entire system can be brought together in virtual space, and the conflicts and clashes discovered cheaply and quickly. It is only once that these issues of form have been resolved that the translation to physical models need to occur.

While uncovering emergent form issues is a tremendous improvement over the iterative and costly two-dimensional blueprints to physical models, the ability to simulate behavior of the system in digital form is a quantum leap in discovering and understanding emergent behavior. System creators can now test and understand how their systems will behave under a wide variety of environments, using virtual space and simulation.

Also as shown in Fig. 3, the ability to have multiple virtual spaces as indicated by the blocks labeled $VS_1 \ldots VS_n$ meant that that the system could be put through destructive tests inexpensively. When physical prototypes were the only means of testing, a destructive test meant the end of that costly prototype and potentially its environment. A physical rocket that blows up on the launch pad destroys the rocket and launch pad, the cost of which is enormous. The virtual rocket only blows up the virtual rocket and virtual launch pad, which can be recreated in a new virtual space at close to zero cost.

The create phase is the phase in which we do the bulk of the work in filling in the system's four emergent areas: PD, PU, UD, and UU. While the traditional emphasis has been on verifying and validating the requirements or predicted desirable (PD) and eliminating the problems and failures or the predicted undesirable (PU), the DTP model is also an opportunity to identify and eliminate the unpredicted undesirable (UU). By varying simulation parameters across the possible range they can take, we can investigate the non-linear behavior that may have combinations or discontinuities that lead to catastrophic problems.

Once the virtual system is completed and validated, the information is used in real space to create a physical twin. If we have done our modeling and simulation correctly, meaning we have accurately modeled and simulated the real world in virtual space over a range of possibilities, we should have dramatically reduced the number of UUs.

This is not to say we can model and simulate all possibilities. Because of all the possible permutations and combinations in a complex system, exploring all possibilities may not be feasible in the time allowed. However, the exponential advances

in computing capability mean that we can keep expanding the possibilities that we can examine.

It is in this create phase that we can attempt to mitigate or eradicate the major source of UUs—ones caused by human interaction. We can test the virtual system under a wide variety of conditions with a wide variety of human actors. System designers often do not allow for conditions that they cannot conceive of occurring. No one would think of interacting with system in such a way—until people actually do just that in moments of panic in a crisis.

Before this ability to simulate our systems, we often tested systems using the most competent and experienced personnel because we could not afford expensive failures of physical prototypes. But most systems are operated by a relatively wide range of personnel. There is an old joke that goes, "What do they call the medical student who graduates at the bottom of his or her class?" Answer, "Doctor." We can now afford to virtually test systems with a diversity of personnel, including the least qualified personnel, because virtual failures are not only inexpensive, but they point out UUs that we have not considered.

We next move into the following phase of the lifecycle, the production phase. Here we start to build physical systems with specific and potentially unique configurations. We need to reflect these configurations, the as-builts, as a DTI in virtual space so that we can have knowledge of the exact specifications and makeup of these systems without having to be in possession of the physical systems.

So in terms of the Digital Twin, the flow goes in the opposite direction from the create phase. The physical system is built. The data about that physical build is sent to virtual space. A virtual representation of that exact physical system is created in digital space.

In the support/sustain phase, we find out whether our predictions about the system behavior were accurate. The real and virtual systems maintain their linkage. Changes to the real system occur in both form, i.e., replacement parts, and behavior, i.e., state changes. It is during this phase that we find out whether our predicted desirable performance actually occurs and whether we eliminated the predicted undesirable behaviors.

This is the phase when we see those nasty unpredicted undesirable behaviors. If we have done a good job in ferreting out UUs in the create phase with modeling and simulation, then these UUs will be annoyances but will cause only minor problems. However, as has often been the case in complex systems in the past, these UUs can be major and costly problems to resolve. In the extreme cases, these UUs can be catastrophic failures with loss of life and property.

In this phase the linkage between the real system and virtual system goes both ways. As the physical system undergoes changes we capture those changes in the virtual system so that we know the exact configuration of each system in use. On the other side, we can use the information from our virtual systems to predict performance and failures of the physical systems. We can aggregate information over a range of systems to correlate specific state changes with the high probability of future failures.

As mentioned before, the final phase, disposal/decommissioning, is often ignored as an actual phase. There are two reasons in the context of this topic why the disposal phase should receive closer attention. The first is that knowledge about a system's behavior is often lost when the system is retired. The next generation of the system often has similar problems that could have been avoided by using knowledge about the predecessor system. While the physical system may need to be retired, the information about it can be retained at little cost.

Second, while the topic at hand is emergent behavior of the system as it is in use, there is the issue of emergent impact of the system on the environment upon disposal. Without maintaining the design information about what material is in the system and how it is to be disposed of properly, the system may be disposed of in a haphazard and improper way.

3.4 System Engineering Models and the Digital Twin

Systems Engineering is commonly represented by a few different models: the Waterfall Model, the Spiral Model, and the Vee Model (Fig. 4) [14, 15, 17]. What these models have in common is a sequential perspective. The Waterfall model is clearly sequential as the sequence flows from design to operation. The Spiral model reflects the same, although there is an iterative aspect to it. The Vee model implies a deconstruction and push down of requirements to the component level and a building back up from components to the complete system.

While these are conceptual models, the messy reality of systems development is that the ideal forward flow from inception to system is simply that—an ideal. What actually happens is that there is a great deal of determining that the system as designed does not really deliver the desired behavior, cannot be manufactured, and is not supportable or sustainable at the desired cost levels. Even when the design goes according to plan using the Vee model, the claim is that it leads to highly fragile systems [22].

The Digital Twin implementation model, as shown in Fig. 5, attempts to convey a sense of being iterative and simultaneous in the development process. Unlike the Waterfall or even Spiral Models, the downstream functional areas as conventionally thought of are brought upstream into the create phase. The "ilities" are part of the considerations of system design.

In fact, these areas can and should influence the design. For example being able to manufacture a honeycombed part through additive manufacturing would result in a part meeting its performance requirement at a significant savings in weight. Without that consideration, designers would specify a more expensive material in order to meet the weight and performance requirements.

What makes this new model feasible is the ability to work in the virtual space of the Digital Twin. The classic sequential models of Systems Engineering were necessitated by the need to work with physical objects. Designs had to be translated into expensive physical prototypes in order to do the downstream work of say,

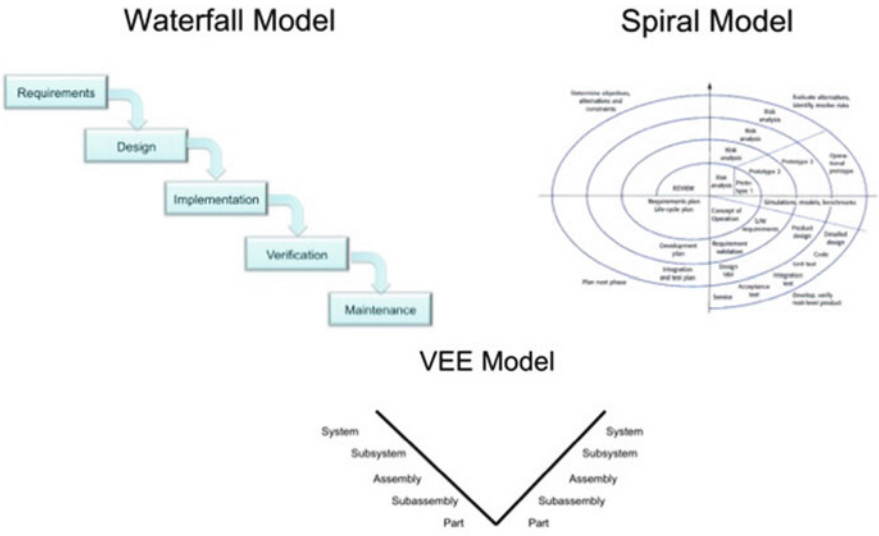

Fig. 4 System engineering model

manufacturing. This meant that only a subset of designs could be considered, because the cost of getting it wrong and having to go back and redesign was expensive and time consuming.

The Digital Twin changes that with its ability to model and simulate digitally. As indicated in Fig. 5, downstream functional areas can influence design because working with digital models in the create phase is much cheaper and faster and will continue to move in that direction.

While this new model is advantageous with traditional subtractive manufacturing methods, it will be required as additive manufacturing technologies advance and become mainstream production capabilities. As described by Witherell et al. [23] in another chapter ("Additive Manufacturing: A Trans-disciplinary Experience") here, the design-to-production process for additive manufacturing is much more integrative than subtractive manufacturing. Integration is a major hallmark of the Digital Twin Implementation Model. Additionally, the authors describe Digital Twin like modeling and simulation as having high potential for additive manufacturing part qualification.

While Digital Twin Implementation Model needs more detail and maturation, the concepts behind it, addressing the "ilities" as early as possible, integration across the lifecycle, and fast iterations, address the shortcomings of the current Systems Engineering models. The maturation and adoption of additive manufacturing will only serve to highlight this need.

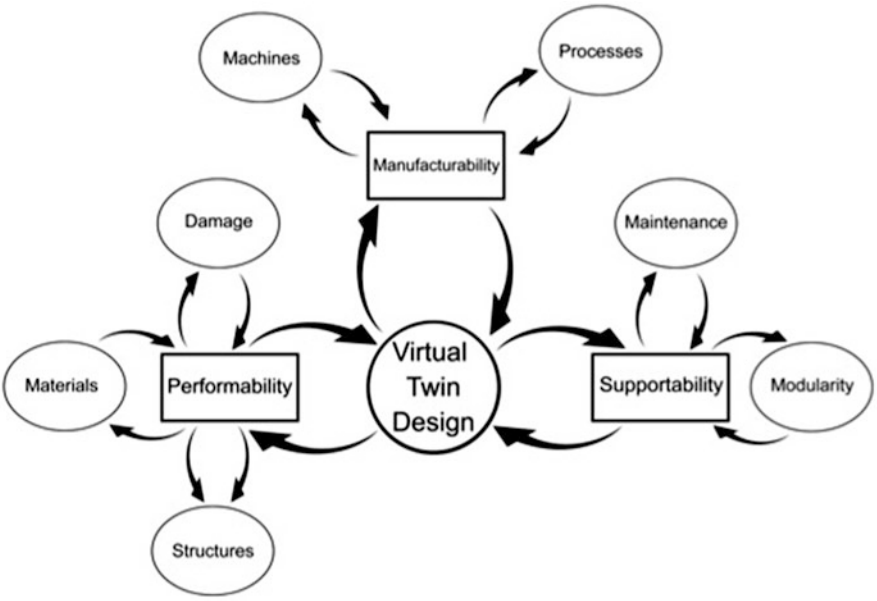

Fig. 5 Digital twin implementation model

3.5 The Digital Twin and Big Data

The Digital Twin model requires massive amounts of information and computing capability. Its progression will rely on advances in computer and communications technology. As things currently stand, this should not be a barrier to the advancement of this model. Moore's law is still alive and well, with not only computing technology growing exponentially, but also storage and bandwidth.

The advantages of using the virtual model rather than relying on expensive prototypes can be illustrated in Fig. 6. This is the cost of working with physical material versus virtual. If we assume that the costs are equal today, and project those costs into the future, we can see how they diverge with physical costs increasing at the rate of inflation and virtual cost decreasing on an exponential basis.

The other advantage is that with atom-based models, they have to exist in a certain geographical location. The parts of the system cannot be distributed if we want to execute the system itself. With virtual systems, the components of the system can be widely distributed, without the user even knowing where the components of the system reside. All this requires coordination among the virtual components. While this is a nontrivial requirement, the trajectory of computing technology is enabling this capability.

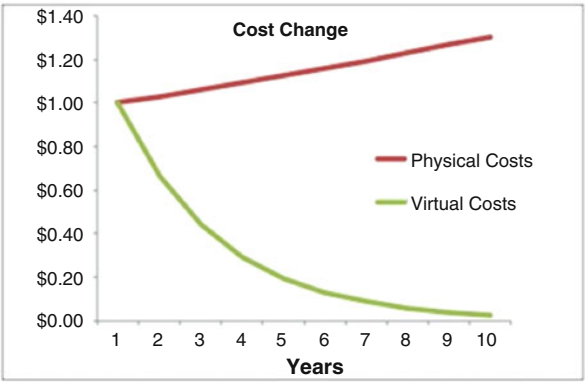

Fig. 6 Real vs. virtual costs

We are also driving deeper and deeper into the make-up of physical structures. The expectation is that we will eventually model at the atom level, incorporating the physical rules of behavior that are embedded with physical components and systems. Boeing's "Atoms to Airplanes" is an example of the initiatives in this area.

While there are many challenges in the area of big data, all indications are that the necessary technical capabilities exist or will exist to enable the Digital Twin.[2]

4 Value of the Digital Twin Model

So what is the value of the Digital Twin model? The Digital Twin is about information. While much has been written about information, it still is a relatively fuzzy concept. Many focus on the "inform" part of information and deal with it as a transmission issue.

However, the core premise for the Digital Twin model is that information is a replacement for wasted physical resources, i.e., time, energy, and material. Take any task, designing an airplane, manufacturing a fuel tank, or diagnosing and repairing an automobile. As shown in Fig. 7 [14, 15, 17], we can represent that task quantitatively as the sum of the cost of all resources required to complete the task. Since we live in a monetized society, we can cost the labor time, the material costs, and the energy costs over the life of the task.

Furthermore, as shown on the left side of the figure, we can divide the tasks into two parts. The bottom part is the minimum amount of physical resources that it would take to perform the task. If we were omniscient and omnipotent, we would

[2]The technical issues will most likely not be what gate this advancement. The lack of interoperability between vendor software systems that implement different aspects of this Digital Twin functionality will be the major issue. The user community will need to be proactive in encouraging the various vendors to play nice in the digital sandbox.

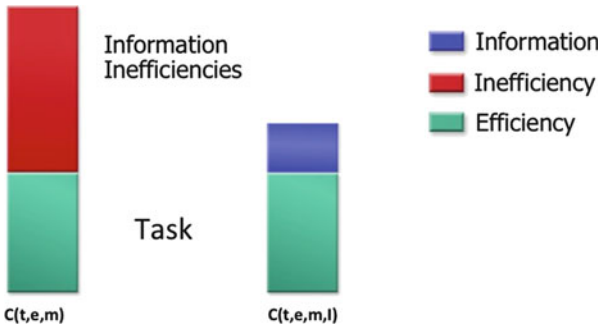

Fig. 7 Information as time, energy, material trade-off

know what the minimum amount of resources it took to perform the task and then perform the task in the exact way that minimizes those resources.

The upper part of the bar represents the waste in resources that we incur in actually performing the task. Because we are not omniscient, we waste resources in a wide variety of ways. We design parts that do not fit with each other. We design entire systems that do not produce the behavior we think that they will produce. We design components that cannot be manufactured. We produce system designs that require far more upkeep than we predicted.

On the right side of the Fig. 7, we have the same task. The amount of resources needed to perform the task without waste is exactly the same. But on the upper part of the bar we have information that allows us to replace some, but not usually all, of the wasted resources necessary to perform this task.

Information is never free to acquire or use, so there is always some use of resources in developing and using information. The key assumption here always is that the cost of these resources of information will be less than the cost of the wasted resources. For repetitive tasks and tasks of complexity, this is usually a reasonable assumption.

Obviously this figure is meant to be illustrative and ideal and not definitive. Humans are neither omniscient nor omnipotent. There will always be information that we do not have or task execution that is not perfect. There will always be wasted physical resources that information does not replace. However, this does not invalidate the point that the use of information can greatly reduce the waste of physical resources.

If we look at the Digital Twin through its lifecycle, we can see where we can obtain this value in each of the phases.

In the create phase, being able to model and simulate both form and behavior via a DTP allow us to save wasted physical resources and the time-consuming and costly production of physical prototypes. Determining predicted undesirable

(PU) behavior and discovering unpredicted undesirable (UU) behavior with simulations save resources both in the create phase and downstream.

In the production phase, simulating manufacture before it occurs allows us to reduce trial and error manufacturing. Collecting the as-built information allows us to understand what components with what specifications are in each system that we produce.

In the operations or support/sustainment phase, the Digital Twin allows us to understand how to more efficiently and effectively maintain the system. In addition, if we can prevent undesirable behaviors, both predicted and unpredicted, we can potentially save the cost of unforeseen "normal accidents". While the claim that human life is priceless, we do put a cost on life as we limit the tests for the states that systems can take. That is we only test for a subset of states the systems can take. However, with the use of the Digital Twin we can dramatically lower the cost that we incur in preventing loss of life by testing more states.

In the disposal phase, having the information about how the system was designed to be safely retired and/or decommissioned will greatly reduce the impact cost on the environment.

The Digital Twin has the potential to have a major impact on reduction of wasted resources in the lifecycle of our systems. Preventing a catastrophe caused by undesirable emergent behavior that results in loss of life is, as they said in the MasterCard ad, priceless.

4.1 Evaluating Virtual Systems

If we are going to create virtual systems, then we will need to evaluate how well these virtual systems mirror their physical counterparts. Grieves has proposed a series of tests, called the Grieves' Tests of Virtuality (GTV), to evaluate how well virtual systems mirror their physical counterparts ([17], pp. 18–19).

These test are based on the Turing tests of intelligence, proposed by Alan Turing in 1950 [24]. Turing called this the "Imitation Game".[3] The premise of the test was that an observer, isolated from a human and a computer could ask questions to both. If the observer could not tell the computer from the human, the computer passes the test. To date, in spite of some claims to the contrary, no computer has passed this test.

However, with all deference to Turing, he had the right idea, but was focusing on the wrong subject. The imitation that computers may be capable of is not imitating human intelligence, but imitating the physical world, *except* for human intelligence.

The premise of the GTV tests is very similar to Turing's Imitation Game. In the test, a human observer is positioned in a room that has two screens. One screen is a

[3]We suspect that Turing would be astounded that this would be the name of a movie featuring his life and work.

video feed from an actual room where a physical system is present. The other screen is connected to a computer.

There are three tests that the observer can perform. These three tests are: a Sensory Visual Test, a Performance Test, and a Reflectivity Test.

When first proposed, the second two tests were the same. However, the first test was called the Visual Test. It was pointed out that we have other senses than simply our eyes. This indeed is true. However, our visual sense is our highest bandwidth sense.

The other senses are important, and we do not want to give them short shrift. In fact, with respect to the sense of hearing, the throaty roar of a Chevrolet Corvette and the rumbling of a Harley-Davidson motorcycle are clearly distinctive. In fact, with respect to the Corvette, engineers actually tune the exhaust system in order to give it the right tone and pitch. So a Sensory Audio Test certainly is not only feasible but is actually being done.

In addition, the work in haptics technology continues to evolve. We have the ability to put on special gloves and actually feel virtual surfaces, with the gloves providing the necessary feedback to convince us that we have touched a solid object. We can grasp a virtual steering wheel. We can turn a virtual wrench and feel the resistance of the virtual bolt.

However, because of the importance of the visual sense, we are going to only focus on a Sensory Visual Test. We will leave it up to the reader to fashion tests for the other senses. It will be along the lines of the sensory visual test.

For the Sensory Visual Test, the observer can ask that any spatial manipulation can be performed on both the physical and virtual system. These systems can be rotated, lifted, and even disassembled. If the observer cannot tell which is the physical system and which is the virtual system, then the Sensory Visual Test is said to be passed.

For the Performance Test, the observer can ask to have any action performed on the physical and virtual system. He or she can ask to have power applied to it; to have stresses placed on it; to place it in a wind tunnel; or any other action that could be applied to the system. Again, if the observer cannot tell the difference between the behavior of the physical system and the behavior of the virtual system, then the virtual system is said to have passed the Performance Test.

The last test is the Reflectivity Test. This test requires a little imagination, because the observer is going to know that the physical product is out and about in the physical world. Again, we have the same observer. This observer looks at the same two screens, one of which holds the physical system and the other of which holds the virtual system.

The observer can ask to see the status of any aspect of the product. For example, he or she could ask for the odometer reading of an automobile or the fuel gauge reading in a helicopter. Or, he or she could ask to see the model and serial number of the fuel pump on an airplane or the nozzle pressure of a jet engine at takeoff. If this observer can get the same information from the virtual system as he or she could get from inspecting the physical product, the virtual system is said to pass the Reflectivity Test.

4.2 Virtuality Test Progress

So as we come to the middle of the second decade of the twenty-first century, how are we doing with our Tests of Virtuality?

4.2.1 Visual Tests

If we look at the Sensory Visual Test, the answer is that we have made a great deal of progress since the year 2000. While at the beginning of this millennium, the visual representation of systems was getting pretty good, it was fairly easy to tell the virtual system from the physical system. As the years moved along, the visual representations kept getting better and better.

Today, in many situations, virtual systems pass the Sensory Visual Test. Many companies now have life-size power walls upon which they project their digital designs. By rotating them on the screen, these virtual systems can be looked at from any angle, and, in the case of automobiles, doors and hoods can be opened and closed. The view from the interior of the car is as if we were sitting in it.

Since the technological improvements will continue, for all practical purposes we can say that we are passing the Sensory Visual Test. No doubt we will continue to expand our requirements in this area.

The requirement for the 3-D version is to no longer confine the observer to looking through the windows of a physical and virtual room. In this test, we can require that the observer be allowed to walk around the system. In keeping with this being a visual test, we can confine the observer to simply visually inspecting the systems, still allowing him or her to perform any spatial operations. We have some of this capability in 3D Caves and Virtual Reality (VR) goggles today. Holographic displays of tomorrow are within technological reach.

What this means for the Digital Twin is that we should have no UUs of emergent form. We can fit together the entire system virtually, even allowing for tolerances. As components are being manufactured, we can scan and replace the designed parts with actual as-built parts. Our Digital Twin can both predict what the actual form should be and then reflect what the actual form is. There should be no surprises in form anymore.

4.2.2 Performance Tests

The Performance Test is moving along more slowly. The requirements for this test are much more expansive. We not only have to sense the virtual system as if it was physical, but this virtual system must also behave exactly like the physical system. The requirement of this test is that we can have any operation performed on this system that we could have performed on the corresponding physical system.

This means that our virtual environment would need to have all the physical laws of the universe built into it—a daunting task. But what we can hope for in the near future is to build in to our virtual environment selected laws of the universe.

This is where we are today. The issue is that we can abstract certain features of systems today and subject them to specific kind of tests. These tests are generally referred to as "solvers", and there are different solvers for different kind of problems.

We have solvers for finite element analysis, stress testing, heat analysis, material deformation, aerodynamic airflow, and other tests. The issue is that we generally have to prepare the system for the test by abstracting the relevant information that we are interested in studying. This abstracted information is then used in the solver. This information is changed until the solver produces the results that we are looking for. The resulting information then needs to be incorporated back into the virtual system.

Because this information has been abstracted and subject to manipulation in the solver, integrating it back into the virtual system may require interpretation and trial and error. In the example of finite element analysis, the trend is to integrate these solvers with a virtual object so as to eliminate abstraction and reinterpretation of the solver results back into the virtual product.

We would expect this trend to continue. Although unlike the Sensory Visual Test, there may not be a specific point in time where we can say that we have "passed" the Performance Test. It may be that we simply integrate more and more solvers and make them multi-domain with their virtual systems.

One of the main issues of the Performance Tests is how do we evaluate performance? What we have done with physical systems is to create instruments that measure the characteristics that we are interested in. We then correlate those instrument readings with performance characteristics so as to make decisions as to whether we are getting the required performance. Some of the instruments that we have created are temperature gauges, force meters, flow meters, and impact gauges.

While we do use our visual senses to evaluate performance, they are generally not broad or precise enough and need to be augmented to really assess performance. For example, we insert smoke into a wind tunnel test in order to "see" aerodynamic performance. We use high-speed photography in order to capture the results of a crash tests, because our natural vision is not fast enough to either see the details of the crash as it occurs nor to accurately capture the sequence of events.

The virtual environment cannot only emulate the creation of our instruments, but it can translate those results so that our senses can be brought to bear in evaluating virtual product performance. So for example we can see the heat of a virtual system by using a color-coded gradient to indicate the temperature of each element of that system. We can see the aerodynamic properties of the vehicle in motion by making visible the virtual particles of air.

It is with these Performance Tests that we can seek to validate the Predicted Desirable behaviors of our system, validate that we have eliminated the Predictable Undesirable behaviors, and attempt to uncover Unpredicted Undesirable Behaviors. We still have much more to do in this area.

4.2.3 Reflectivity Tests

With respect to the Reflectivity Test, it is clearly in its infancy. The idea of a company maintaining a link with its product after it has left the factory door is very much a twenty-first century concept. The links in the past were only maintained by taking possession of the physical product on a periodic basis, such as bringing it into a repair hub or bringing it back to the factory for an overhaul.

Today, it is difficult to find a product that does not have a microprocessor integrated into it. There is more computing value in today's cars than there is steel value. But it is not just large ticket items such as automobiles and airplanes. Almost any product that we can think of has a microprocessor in it. This includes washers, dryers, refrigerators, thermostats, pacemakers, and even running shoes.

With these microprocessors and their ability to communicate, companies can continue to stay in touch with their product long after the product has left their factory door. This will allow for these products to communicate back the current state of their condition at all times.

This embedding of connections is becoming much more ubiquitous with the rise of the Internet of Things (IoT) [25]. Components within a system can interact with each other, exchanging statuses and making request for services. This can and most likely will result in emerging behaviors that are currently unforeseen and unpredicted.

It is going to be important to have a Digital Twin that reflects this activity for a number of reasons. First, capturing these interactions will be critical in understanding what emerging behaviors occurred and having an audit trail of activities that led up to these behaviors. Second, only by seeing what is occurring can humans have a hope of stepping in when UU behaviors start occurring.

We are on the path of having autonomy in vehicles such as automobiles, farm equipment, and drones. We have not given as much thought to reflectivity as we should. The concepts of the Digital Twin are something we should strongly consider as we move in this direction.

5 Digital Twin Obstacles and Possibilities

As with all new concepts, there are both obstacles and possibilities that present themselves. In this section, we will discuss a few of them.

5.1 Obstacles

As we see it, there are three main obstacles that will need to be addressed. These obstacles are organizational siloing, knowledge of the physical world, and the number of possible states that systems can take.

Organizational siloing probably presents the biggest obstacle to the Digital Twin. Organizations are divided into functions such as design, engineering, manufacturing, and support. There is a natural siloing of information within these functional areas. Each of these informational silos has information about the systems. However there may be very little sharing across functions.

To use but one example, most organizations still have the issue of engineering and manufacturing having different bill of materials for the same components. Even though the component that engineering deals with and the component that manufacturing deals with are one and the same, each of these areas approaches the parts that make up these components in a different way. The Digital Twin concept requires a homogeneous perspective of this information that persists across functional boundaries.

Even within functions there is siloing and fragmentation. The domains of mechanical engineering, electrical engineering, programming, and systems engineering exists separate and apart from each other. In many if not most organizations, simulation exists specific to a domain and does not have a multi-domain focus.

Even with areas that one might think should be the same such as manufacturing and materials, the work in one domain does not carry over to another. Mechanical engineers may be developing a structure that requires a certain weight limit. The material people may have different perspectives on how material affects not only weight but also structure. It is not until later in the development cycle that these two issues, which should be related, are reconciled, adding additional costs and delays.

These are cultural issues that will need to be addressed. Cultural issues are much more difficult to address than technical issues. We expect that the technology needed to address these issues will be available much earlier than the cultural changes necessary to fully adopt and make use of the technology.

The next obstacle is simply our understanding of the physical world. Technology in general requires us to capture and understand physical phenomena [26]. The Digital Twin concept is built on understanding and being able to simulate natural phenomena. While this area is rapidly increasing, the Digital Twin requires that we need to know how our system will react to the forces that it will encounter. Being able to understand, model, and simulate structures, materials, and force phenomena will be critical in doing meaningful digital analysis of how our system will actually perform.

Currently we are hard at work at understanding how structures respond to forces and how materials respond, fracture, and deteriorate in the face of the substantial forces that we encounter in airplane and rocket flight.

The third obstacle is simply the sheer number of states that the system can take over time. If we are to tease out undesirable emergent behaviors, we will need to simulate the conditions that systems face under a range of parameter changes, where we may to be dealing with thousands of parameters. We simply may not have the amount of computing capability required to perform all the computations that we require. This is a problem that diminishes in time as computing and its associated technologies continue to advance at exponential rates.

5.2 Possibilities

We are already seeing the possibility of trading costly and time-consuming physical prototypes for modeling and simulation. We are seeing great strides in these areas. As an example, Boeing recently patented an "Atoms to Airplanes" perspective that deals with the modeling and simulation of composite materials.

There are two interesting possibilities to deal with complexity that we will discuss. The first is capturing in-use information to feed back into the creation phase. The second possibility is the idea of "front-running" a simulation in real time. The first possibility could allow us to uncover complexity issues, i.e. UUs, and deal with them before the system is deployed. The second could help mitigate UUs that arise as the system operates.

With the ability to capture data on how the system is produced and used in the Production and Operational phases, we can collect system states that would collectively be profiles that we can use in future development activities to simulate the new system's behavior. Running these states through the simulation of a new system could help point out particular profiles that would give UUs. Since these profiles would reflect human interactions, designers would give particular attention to those profiles where humans interacted in unexpected ways.

As Fig. 5 shows, these profiles could be run via simulations in a simultaneous and iterative fashion, well before a physical system is produced. While there might be many more possible system states, this would cover many states that might not be considered by the system designer. It would be helpful in identifying complexity because uncovering UUs would point out areas that need simplification.

The second possible opportunity is to help mitigate complexity. In this possibility we would have our simulation "front-run" the system in actual use in real-time. This means that we would run the simulation with real-time feeds for what is happening to the system in actual use. The simulation could present a window into the future of the possible system states that might occur.

This might be helpful in the case of system crises, where it is unclear as to what is happening with the system. As was pointed out earlier, humans often have a problem with sensemaking. They jump to a final conclusion almost immediately, even though working through the issue with a systematic methodology would present other possibilities they should consider [27]. Once there, they lock onto a paradigm that they cannot easily discard. System front running would show them

other possibilities of what is occurring and possibly help in making better sense of the situation.

This obviously would work best when the crisis is developing over time. The BP Gulf Disaster would be such an example [28]. There were a number of indications that something was going awry. However, the operators were so locked into a frame of reference that they could not see other alternatives.

It would even be useful in events that are fast moving and looking seconds ahead could be the difference between life and death. The example of this is the Air France 447 flight that crashed off of the coast of Brazil. A simulation front running in advance of what was occurring could have alerted the pilots that they were doing exactly the wrong thing in pulling the nose of the airplane up and that continuing their actions would result in a fatal stall.

Obviously the front running capability requires computing capability that can run faster than the physical activity it is simulating. As computing power continues to increase, there will be more and more instances where front running will be possible.

With the introduction of new, conceptual models, there will always be both unforeseen obstacles and possibilities. The key will be to remain alert to both address the obstacles and exploit the possibilities.

6 A NASA Approach to the Digital Twin

NASA has three major issues that most organizations do not have. These issues are: (a) the systems that they create are very expensive; (b) they make very few of these systems; and (c) the systems that they make have not been made before.

NASA's systems certainly qualify as complex systems. Components in its launch systems range from the nanometer scale to components that are tens of meters in size. NASA also has system of systems, because launch systems include, rockets, solid-state boosters, capsules, launch pads, and flight operations. NASA's launch systems are susceptible to "normal accidents" where minor problems cascade into major catastrophes. An O-Ring that failed caused the space shuttle Challenger's explosion. A strike of insulating foam falling off at launch critically damaged the space shuttle Columbia.

In the past, NASA has addressed these issues with additional resources. They had the resources in order to do expensive, time-consuming testing with physical prototypes. However, this is no longer a viable alternative. NASA has added affordability as a major criterion for their space launch capabilities. They are under extreme pressure to create their systems, especially their new Space Launch System, using their resources much more efficiently and effectively than may have been done in the past.

It is in this new climate that NASA is investigating the usage of the Digital Twin model. The proponents of this model within NASA believe that it can take cost out

of the budget and time off the schedule. This has the possibility in resulting in major resource savings for NASA.

NASA recently completed a project in developing a composite tank. While the project was successful, the project pointed out issues that need addressing. A major issue was that there were major discrepancies between the performance predicted by the Advanced Concepts Office (ACO), the performance predicted by the in-depth analysis of experts, and the actual physical tests. In addition, the in-depth analysis was not completed until after the production of the composite tank was well underway.

Both the gaps between the two predicted results and the actual results and the timing of the in-depth analysis are problems that need to be resolved and done better. We need the predicted behavior and the actual behavior to be much closer. We also need the in-depth analysis to be completed before we begin actual production.

The proponents of the Digital Twin concept within NASA think that this concept has the opportunity to help in both these areas. By focusing on creating better and more complete information about the virtual system, NASA can do a better job in predicting the actual performance of its systems and reducing the possibility of UU problems.

In addition, unlike physical systems, which either exists or don't exist, the Digital Twin allows for maturity of information so that analysis can begin much earlier and not wait until the design is complete. This should allow for in-depth analysis to start much earlier than it has in the past and finish before the manufacture of the actual system.

The Digital Twin is not the main thrust of this project. However, success in proving the viability of its concepts has the possibility to have a major impact on affordability within NASA.

7 Conclusion

The premise driving the Digital Twin concept was that each system consisted of two systems, the physical system that has always existed and a new virtual system that contained all of the information about the physical system. Because information is a replacement for wasted physical resources, we can use much less costly resources in creating, producing, and operating systems. Through modeling and simulation in virtual space we can better understand the emergent form and behaviors of systems and diminish the "I didn't see that coming" factor.

Virtual systems can be a great assist in dealing with the four categories of system behaviors. We can ensure we will obtain the Predicted Desirable (PD), eliminate the Predictable Undesirable (PU), and decrease the Unpredicted Undesirable (UU) [If we have Unpredicted Desirable (UD), that will not hurt us, although it does point out that we do not full understand our system].

Complex systems are susceptible to "normal accidents". While the term "emergent behavior" is used, we would contend that it is behavior that potentially existed from the inception of the system, although one of its main causes is the system's interaction with humans who behave in unexpected ways. This is often caused by a failure in sensemaking. By the Digital Twin's simulation, we may be able to reduce the UUs caused by unexpected human interaction. We specifically are not dealing with evolving systems: systems that have randomness built in or change the rules that govern their behaviors in operation.

Systems do not burst forth fully formed. They progress through a lifecycle of creation, production, operation, and disposal. With "physical-only" systems, this was a linear progression. The Digital Twin allows for a more iterative, simultaneous development to consider the "ilities". While there are obstacles ahead, especially cultural ones, we have made significant progress in the last decade and a half as we have shown via the Grieves Virtual Tests. We should expect future advances, as computing technology shows no sign of slowing its rate of progress.

Finally the Digital Twin and its reflection of the physical system mean that we can potentially use the virtual system even when the physical system is in operation. Two potential uses are capturing and using in-use information and system front running.

The Digital Twin concept has the opportunity to change how we view system design, manufacturing and operation, reduce the UUs of complex systems, and augment Systems Engineering.

References

1. Grieves, M. (2012). Virtually indistinguishable: Systems engineering and PLM. In L. Rivest, A. Bouras, & B. Louhichi (Eds.), *Product lifecycle management: Towards knowledge-rich enterprises* (pp. 226–242). Montreal: Springer.
2. Ackoff, R. L. (1971). Towards a system of systems concepts. *Management Science (pre-1986) 17*(11), 661.
3. Sargut, G., & McGrath, R. G. (2011). Learning to live with complexity. *Harvard Business Review, 89*(9), 68–76.
4. Nature. (2008). Language: Disputed definitions. *Nature, 455*, 1023–1028.
5. Mitchell, M. (2009). *Complexity: a guided tour*. Oxford: Oxford University Press.
6. Perrow, C. (1984). *Normal accidents: Living with high-risk technologies*. New York: Basic Books.
7. Weick, K. E. (2005). Making sense of blurred images: Mindful organizing in mission STS-107. In M. Farjoun & W. Starbuck (Eds.), *Organization at the limit: Lessons from the Columbia disaster* (pp. 159–177). Malden, MA: Blackwell.
8. Weick, K. E. (1990). The vulnerable system: An analysis of the Tenerife air disaster. *Journal of Management, 16*(3), 571–593.
9. Troyer, L. (2015). Expanding sociotechnical systems theory through the trans-disciplinary lens of complexity theory. In F.-J. Kahlen, S. Flumerfelt, & A. Alves (Eds.), *Trans-disciplinary perspectives on system complexity*. New York: Springer.
10. Aristotle, H. T., & Armstrong, G. C. (1933). *The metaphysics*. London, New York: W. Heinemann, G. P. Putnam's Sons.

11. Ablowitz, R. (1939). The theory of emergence. *Philosophy of Science, 6*(1), 16.
12. Felder, W. N. (2013). *Interactions among components in complex systems, AIAA CASE 2013 Framing Paper*, Complex Aerospace Systems Exchange, San Diego, CA, 10-12 Sept 2013, http://www.aiaa.org/uploadedFiles/Events/Conferences/2013_Conferences/2013__Aviation/Detailed_Program/CASE%202013%20Framing%20Paper.pdf, Accessed 22 June16.
13. Holland, O. T. (2007). Taxonomy for the modeling and simulation of emergent behavior systems. In *Proceedings of the 2007 spring simulation multiconference-Volume 2, Society for Computer Simulation International*.
14. Grieves, M. (2011). *Virtually perfect: Driving innovative and lean products through product lifecycle management*. Cocoa Beach, FL: Space Coast Press.
15. Grieves, M. (2009). *Virtually perfect: Driving innovative and lean products through product lifecycle management*. Cocoa Beach, FL: Space Coast Press.
16. Grieves, M. (2005). Product lifecycle management: The new paradigm for enterprises. *International Journal Product Development, 2*(1/2), 71–84.
17. Grieves, M. (2006). *Product lifecycle management: Driving the next generation of lean thinking*. New York: McGraw-Hill.
18. Piascik, R., Vickers, J., Lowry, D., Scotti, S., Stewart, J., & Calomino, A. (2010). *Technology area 12: Materials, structures, mechanical systems, and manufacturing road map*. NASA Office of Chief Technologist.
19. Glaessgen, E. H., & Stargel, D. (2012). *The digital twin paradigm for future NASA and US air force vehicles*. AAIA 53rd Structures, Structural Dynamics, and Materials Conference, Honolulu, Hawaii.
20. Eric, J. T., Anthony, R. I., Thomas, G. E., & Michael Spottswood, S. (2011). Reengineering aircraft structural life prediction using a digital twin. *International Journal of Aerospace Engineering, 2011*(154798), 14. doi:10.1155/2011/154798.
21. Cerrone, A., Hochhalter, J., Heber, G., & Ingraffea, A. (2014). On the effects of modeling as-manufactured geometry: Toward digital twin. *International Journal of Aerospace Engineering* 2014.
22. Eremenko, P., & Wiedenman, N. L. (2010). *Adaptive Vehicle Make (AVM)*. Driving Innovation for Superior Defense Manufacturing, Washington, DC, NDIA – DARPA.
23. Witherell, P., Jones, A., & Lu, Y. (2015). Additive manufacturing: A trans-disciplinary experience. In F.-J. Kahlen, S. Flumerfelt, & A. Alves (Eds.), *Trans-disciplinary perspectives on system complexity*. New York: Springer.
24. Turing, A. (1950). Computing machinery and intelligence. *Mind, LIX*(236), 433–460.
25. Porter, M. E., & Heppelmann, J. E. (2014). How smart, connected products are transforming competition. *Harvard Business Review, 92*(11), 64–88.
26. Arthur, W. B. (2009). *The nature of technology: What it is and how it evolves*. New York: Free Press.
27. Grieves, M. (2014). Process, practice, and innovation. In P. Gupta & B. Trusko (Eds.), *Global innovation science handbook* (pp. 159–172). New York: McGraw Hill Professional.
28. Graham, B., Reilly, W., Beinecke, F., Boesch, D., Garcia, T., Murray, C., et al. (2011). *Deep water: The gulf oil disaster and the future of offshore drilling*. National Commission on the BP Deepwater Horizon Oil Spill.