An Introduction to the Autonomous Framework by Dell Technologies

August 2021

H18889

White Paper

Abstract

This white paper describes a six level Autonomous Framework for IT operations, from no automation to full autonomy. It also provides background information, expands upon each level of this framework, and provides some considerations for the road ahead.

Dell Technologies Solutions

Copyright

Contents

Executive Summary

This white paper describes a six level Autonomous Framework for IT operations, from no automation to full autonomy. It also provides background information, expands upon each level of this framework, and provides some considerations for the road ahead.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell Technologies Solutions team by email or provide your comments by completing our documentation survey.

Author: David Iovino

Introduction

In recent years, advancements in technology – especially, those in robotics, automation, and artificial intelligence – have fueled many predictions about the future. These predictions run the gamut, but the ones that are the most attention-grabbing and fun to contemplate, are the techno-dystopian and techno-utopian predictions. They go something like this:

- Techno-dystopian: Technology is driving us towards a dystopian future, where human labor is obsolete, human intelligence no longer reigns supreme, and privacy and freedom are quaint notions from the past.
- Techno-utopian: Technology is allowing us to build a better future, where the human condition is greatly improved. Human suffering and toil are things of the past and lives are lived in the pursuit of creativity and meaning.

Which of these predictions resonates with you? Well, that mainly depends on whether you lean more towards pessimism or optimism. Unfortunately, these prognostications draw our attention away from the here and now. The here and now, is our double time march towards an increasingly digital reality. This is often referred to as Digital Transformation and it is placing tremendous strain on many organizations. This strain is most acutely felt in the expanding labor requirements to manage and utilize an organization's rapidly growing portfolio of digital assets. There is a simple realization that often flies under the radar:

In the coming years, we will likely face the greatest skilled labor shortage in history.

Ok, so what is the path forward then? Well, before we can answer that, we need to take a quick look at the path already traveled.

The advancements in computer technology, since Gordon Moore first posited his now famous observation about the doubling of transistors, has been something to behold. Exponential improvements in compute, memory, and storage, along with vast improvements in bandwidth, have created an environment where incredible gains through software are being realized. Add to this the power of the internet to interconnect just about

everything and you realize that Marc Andreessen was correct when he wrote, “In short, software is eating the world”[1].

Nearly as impressive as the technological gains themselves has been their deflationary effect. The incredible reduction in the cost per unit of compute, memory, storage, and bandwidth, along with their miniaturization, has allowed software to do more and do it in more places, even in the palm of your hand. Many discrete devices from the not-so-distant past are now just apps on a smart phone. Many interactions previously done in-person are now just part of the mobile computing landscape. These incredible advancements have been a boon for business. Because of this, most companies have embraced Digital Transformation.

All of these advances in software haven’t been without drawbacks. Today, we have a complex landscape of layered software, filled with pitfalls and obligations, that put a heavy strain on organizations.

Now, add to this already complex landscape, some new design and operational patterns:

- Who develops and maintains software (Open Source)
- How software development is approached (Agile)
- The relationship between software development and operations (DevOps)
- How infrastructure and software are delivered (Cloud Computing)
- How applications should be packaged (Virtual Machines, Containers, and so on)
- How software should be architected (Cloud Native, Microservices, and so on)
- How software should reach a result (Machine Learning vs. explicit programming)
- Where computing should take place (Datacenter vs. Cloud vs. edge)

But it doesn’t end there. Software typically exists for one purpose: to receive, process, store, retrieve, and analyze data. And just as data has grown massively over the years, the measurement terminology used in discussions about data has rapidly progressed. We hardly hear about megabytes, gigabytes, and terabytes anymore. Now petabytes, exabytes and even zettabytes are commonly used. Creative terms to describe large stores of data – such as, data warehouse, data mart (typically, a subset of a data warehouse), and even data lake – are frequently thrown about. Even with the vast improvements in bandwidth, it has become harder and harder to move data.

At some point, the term *data gravity*[2] entered the lexicon to express the idea that the placement of data affects the placement of software. In addition, data has become more and more valuable, due in part to the need for large datasets to train machine learning models.

When it comes to this valuable data, two acronyms you never want to hear are DL (data loss) and DU (data unavailability), an indication that the protection and availability of data

[1] Marc Andressen, "Why Software Is Eating The World", Wall Street Journal, August 20, 2011

[2] Dave McCrory is generally credited with coining the term “Data Gravity” while working for Dell in 2010.

has become a vast area-of-interest by itself. Data is encrypted, replicated, duplicated, cloned, snapshotted, backed up, and so on.

Today's IT environment – one of layered software, unfathomable amounts of data, and the vast infrastructure necessary to make it all possible – is a challenging one, to say the least.

All of this must be managed, maintained, upgraded, protected, secured, and repaired. Traditionally, this has fallen to human operators who need to perform thousands of operations over the course of a year.

Early on, they did this with CLIs (command-line interfaces), then with UIs (user interfaces) and more recently, with APIs (application programming interfaces), each with its own strengths and weaknesses.

- CLIs provide limited visibility, require knowledge of command structure and sequencing of operations, but lend themselves to the scripting of repetitive tasks.
- UIs provide broader visibility, embed structure and sequencing knowledge, but don't lend themselves to the scripting of repetitive tasks.
- API design dictates the level of visibility provided and how much structure and sequencing knowledge is embedded, but its value becomes clear when scripting repetitive tasks.

It is impossible to scale out human operators the way we scale out software and hardware. Consider for a moment just the rapid expansion of data and its increasing value to organizations. Even if we exclude all other factors (such as cost), there may not be enough human operators to manage it, never mind the rest of the IT environment. There is a real risk that human operators, or lack thereof, will become the bottleneck.

Add to this the fact that with each manual operation, the risk of human error is ever present. As complexity increases, and with it the number operations, so does the risk. Again, what's the path forward, then?

Well, humans need to get out of the business of day-to-day operations and into higher-level, more creative activities. These are the activities in which they can leverage the general problem-solving abilities they possess as humans: the knowledge they have gained through technical training, and the wisdom they have won through experience. This means that day-to-day operations must be transitioned to automation, but isn't automation hard?

No, automation is a journey.

This white paper presents a six-level framework to help the reader along that journey.

A nod to the SAE

Over the years, a great deal of effort has gone into designing, building, and testing autonomous vehicles. Along the way, much effort has also gone into ensuring that public policy provides the proper support and constraints, and that terminology and definitions become standardized. All of these efforts have yielded value that can be applied outside

of the autonomous vehicle space, but it is the work on terminology and definitions that was most useful to our efforts.

The Society of Automotive Engineers (SAE) has done a great deal of work on terminology and definitions, resulting in the following document: *SAE Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (J3016)*.

The latest version can be found here:
https://www.sae.org/standards/content/j3016_202104/

While exploring autonomous operations for the data center, we saw the work that SAE had done to establish an industry wide taxonomy for Automated Driving Systems (ADS) and felt something similar was required in the world of IT Operations. In IT there are too many nuances in the definition of autonomous operations and lack of a recognized standard. Given our place as the industry leading infrastructure provider we have an obligation to bring clarity to the market.

The Autonomous Framework

Here are the essential elements of the Autonomous Framework:

		Levels	System Responsibility	Human Responsibility		Notes
				Intervention	Input	
Autonomous	Declarative	Level 5 Full Autonomy	System automatically takes action to align with organizational priorities. Automatic alignment with these priorities is expected with or without human input. System can handle all operations without exception.	None	May optionally influence the system by defining organizational priorities	Self-actualized system
		Level 4 Supervised Autonomy	System automatically takes action to achieve service-level objectives. Automatic alignment with these outcomes is expected. System can handle all operations with few exceptions.	Intervenes with few operations	Defines service-level objectives and indicators	SLO-driven automation
Automated		Level 3 Conditional Automation	System uses inferred decision making to achieve generalized outcomes, including insights, recommendations, and actions. System can handle most operations with some exceptions.	Intervenes with many operations	Communicates generalized outcomes	Recommendation-driven automation
		Level 2 Partial Automation	System uses rules-based decision making to achieve explicit outcomes. System is dependent on human for all input and intervention.	Intervenes with most operations	Defines the prescriptive rules to achieve explicit outcomes	Policy-based automation (Infrastructure as Code)
	Imperative	Level 1 Operator Assisted	System is driven by both scripted and manual actions. Human input is required for all operations.	Intervenes with all operations	Creates and initiates scripts to perform actions; Performs many manual actions	Script-based automation
		Level 0 No Automation	System is exclusively driven by manual actions.	Intervenes with all operations	Initiates all actions	Manual

Expanding upon the framework

It is important to note that all levels of the Autonomous Framework use the same scope, simply defined as the 'system'. This allows anyone utilizing the framework the flexibility to define what constitutes a system. For the purposes of this white paper, it is defined as follows:

An arrangement of hardware and software that facilitates meaningful outcomes and is treated as a single entity for automation purposes.

For each level, the expected degree of automated control is described under **System Responsibility**. The expected need for human input and intervention is described under **Human Responsibility**.

It is likely that most organizations have systems in which individual components or aspects are represented at different levels in the framework. This includes Level 0, for those components or aspects that don't lend themselves to automation at all.

With that in mind, use this framework to provide a frame of reference:

- When evaluating the current automation level of the various components or aspects of a system
- When developing a plan for improving the automation level of these components or aspects
- When evaluating the automation capabilities of new software or hardware components

Level 0 (No Automation)

This level describes a system in which every single action is performed by a human operator.

Level 1 (Operator Assisted)

This level describes a system in which rudimentary automation is employed. Human operators have a long history of automating repetitive tasks. We refer to this type of rudimentary automation as *script-based automation*.

Some characteristics of this type of automation are:

- Focuses on a very specific task or set of tasks with the goal of reducing repetitive actions by a human operator.
- Reduces the risk of human error inherent in repetitive manual actions.
- Typically stands alone, is initiated by human operators and is not part of a larger automation framework.
- In many cases, it is independently maintained and not part of a software development ecosystem.
- Involves the programmatic performance of previously manual actions and can be categorized as imperative programming.
- Includes limited error handling and relies heavily on a human operator to intervene.
- Trades one type of work (performance of repetitive tasks) for a different, more creative type of work (script development).
- In many cases, the lower skill tasks are automated first, leaving the higher skill tasks to human operators. If the reduction in lower skill (repetitive) tasks does not significantly outweigh the increase in higher skill (script development, intervention) tasks, then the realized benefit will be minimal or even counterproductive.

Take away: This level is best summed up using the title of a popular technical book: automate the boring stuff[3].

Level 2 (Partial Automation)

This level represents a system in which the concepts of Infrastructure as Code are utilized, including the principal of using a declarative approach. We refer to this type of automation as *policy-based automation*.

Some characteristics of this type of automation are:

- Creates definitions using a high-level descriptive language in machine-readable files
- Maintains machine-readable files under version control.
- Performs deployments and configurations using a Continuous Configuration Automation (CCA) tool.
- Avoids (for the most part) manual deployments and configurations.
- Improves consistency throughout the environment because humans are removed from the deployment and configuration process.
- Improves accountability throughout the environment because changes can be easily tracked.
- Organizational resiliency improves because the human intelligence required to perform deployments and configurations is captured in machine-readable form.
- Reduces the labor requirement for day-to-day operations.
- Requires new skills to create and test definition files.
- In many cases, the organization has embraced new operational patterns (DevOps).
- In many cases, the organization has embraced new design patterns (such as Cloud Native, Microservices).

Take away: This level is best summed up by invoking the Cattle vs. Pets[4] proposition.

Level 3 (Conditional Automation)

This level represents a system in which 'advanced techniques' have been introduced to make recommendations, provide insights, and in some cases perform actions. The expectation is that predictive and prescriptive analytics are present, and that model development is due, at least in part, to machine learning. The result is that the system focus is *inferred decision-making*. An additional expectation is that these 'advanced techniques' are weighted towards recommendations and insights, so we refer to this type of automation as *recommendation-driven automation*.

Some characteristics of this type of automation are:

- Collects large amounts of telemetry data with the intent of using it as part of the decision-making process.

[3] Al Sweigart, "Automate the Boring Stuff with Python", https://automatetheboringstuff.com/.

[4] Bill Baker, an engineer at Microsoft, is generally credited for originating this concept, when he used the preposition "Cattle, not Pets", in a presentation about SQL deployments.

- Typically centralizes and curates the storage of telemetry data (cleansing, normalization, and so on).
- Uses data visualization tools.
- Uses analytics extensively to predict when a resource will be depleted, and in some cases automatically replenishes the resource.
- Uses analytics extensively to predict changes in usage patterns and in some cases, adjusts resources automatically to address the change.
- Uses analytics to detect anomalies, to predict what that anomaly could mean, and in some cases, to take action to address it.
- In many cases, the organization has embraced new operational patterns (MLOps, DataOps).

Take away: This level can be summed up by the statement "We have introduced the power of AI/ML into our systems".

Level 4 (Supervised Autonomy)

This level represents a system that has advanced from automation to an early state of autonomy, where inferred decision-making has progressed to the point that it can perform most actions, making the system capable of automatic alignment. Human input is still required, through the definition of Service Level Objectives (SLOs), so we refer to this type of automation as *SLO-driven automation*.

Some characteristics of this type of automation are:

- System can translate Service Level Objectives into all the underlying patterns required to maintain the prescribed service levels.
- A comprehensive portfolio of analytics detects any behavior that may affect the prescribed service levels.
- Nearly all behaviors can be automatically acted upon to maintain service levels.
- Human intervention is required only in a few instances.

Take away: This level can be summed up by the statement "Our systems are powered by AI/ML".

Level 5 (Full Autonomy)

This level represents a system that has transitioned to full autonomy, in which human input is optional. We refer to this as a *self-actualized system*.

Some characteristics of this type of automation are:

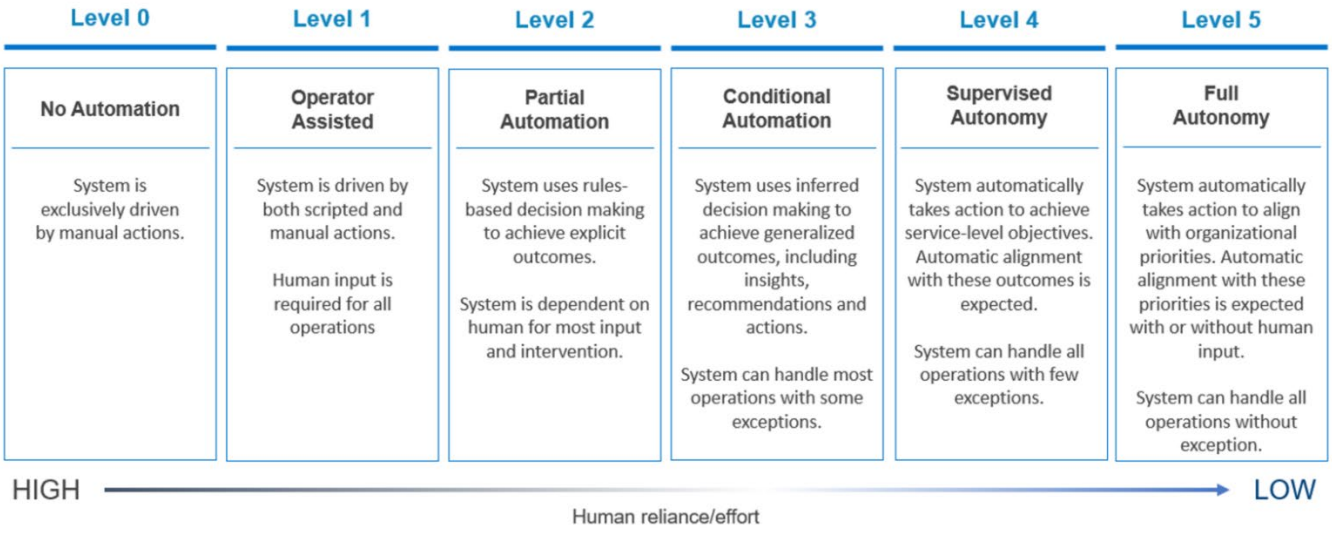
- System has sufficient insight into organizational requirements that human input is no longer required.
- Organizational requirements are automatically met without exception.
- Human intervention is never required.

Take away: This level can be summed up by the following quote: "The factory of the future will have only two employees, a man and a dog. The man will be there to feed the dog. The dog will be there to keep the man from touching the equipment." – Warren G. Bennis.

Automation changes the relationship between humans and systems

At the lower levels of the framework, humans spend a significant amount of time supporting the system. As you move to higher levels, the system becomes more self-sufficient and humans are freed up for higher-level work.

Human focus can shift away from the trees and toward the forest.



Some thoughts for the road ahead

As stated in the introduction, automation is a journey. This section provides some observations and insights for your consideration, whether you are just taking the first steps of the automation journey or are already a mile down this road.

Transformation

When we read a story – a really good story – that centers around a journey, the destination ends up being a secondary component of that story. The primary, and more interesting component, is how the journey changes the characters. Like any good story, the automation journey changes the characters.

"People, Process, and Technology" is a phrase that has been around for decades. Its origins can be traced back to the paper 'Applied Organization Change in Industry' by Harold Leavitt. The phrase has been used extensively, especially by those pitching organizational transformation services. It has been visually represented variously as a Venn diagram, a triangle, and a three-legged stool. It is so widely used and has been around for so long because it reduces a complex subject to three simple words that ring true in our minds. It conveys meaning and feels powerful. It looks as though we have found three excellent characters around which to develop the automation journey story.

People

The most adaptable of our three characters. The ability of humans to learn new technologies and adapt to new situations is simply amazing. The importance of this is

often overlooked in a world dazzled by new technology. But there is an unfortunate side effect of this adaptability. It can cover up weaknesses in process and technology.

Culture

For an organization to make meaningful strides on the automation journey, it will require people to drive the transformation of process and technology. For this to happen, a culture is needed that allows people to thrive.

A few things to consider:

- Innovation comes from people. Whether an innovative idea originates inside an organization or outside, it must be discovered, tried, evaluated, and if useful, adopted. It is important that organizations foster an environment that encourages experimentation with innovative ideas.
- Some of these experiments will result in failure. Effort and sometimes money will be wasted, but this is necessary. Organizations need to foster an environment in which trying and failing is acceptable, and the philosophy of fail fast is promoted.
- People will make mistakes, it is inevitable. Some of these mistakes could have a significant negative impact. It is better to treat these events as learning opportunities, than as witch hunts. Organizations should adopt the concept of the blameless postmortem.

Development

As an organization moves toward higher levels in the autonomous framework, new skills are required. These new skills will vary greatly, by organization, by role and by level. To transform their skillsets, people need time and support.

It is impossible to list all the new skills required, but a few examples are:

- Proficiency working with a version control system (git) and source code control tools (GitHub, GitLab, Bitbucket, and so on), because Level 2 expects machine-readable definition files to be under version control.
- Proficiency working with the RESTful API, because it provides a common interface for automating individual components of a system.
- Experience with tools that use a declarative approach to defining infrastructure, because Level 2 expects this approach.
- Experience with one or more of these tools (Ansible, Terraform, Puppet, Chef, Pulumi, and so on), because Level 2 typically means the use of CCA tools.
- Experience with container platforms (such as Kubernetes and Docker) because many times, new architectural patterns (Cloud Native, Microservices, and so on) favor containers over virtual machines.
- A solid understanding of DevOps principles, because a new operational pattern around Infrastructure as Code emerged a while back.
- A general understanding of analytics, because Levels 3-5 expect analytics to play a significant role.

- A working knowledge of machine learning model training, because Levels 3-5 expect machine learning to play an increasing role in the development of models.
- A general understanding of data management techniques is desirable, because Levels 3-5 are increasingly dependent on data.
- A general understanding of MLOps and DataOps is desirable, because new operational patterns have developed around machine learning and the data needed to drive it.

Communication

As people navigate the process of transformation during the automation journey, it is increasingly important that communication practices also evolve.

A few things to consider:

- New communication and collaboration tools may be necessary to improve the information workflows, especially in the face of an increasing amount of remote work. Don't assume that the current tools will be sufficient.
- With the above statement in mind, as new data sources and ways of instrumenting automation are added, it is crucial that these elements are not the ones driving change in communication and collaboration tools. Technology should be able to integrate into existing information workflows.
- When changes are happening to process and technology, it is important that these changes are effectively communicated to people.
- Until an organization reaches the upper levels of the framework, human operators will be needed for day-to-day operations. People need to understand how the above changes will affect them.
- Proposed changes should be discussed with people early on and buy in should be sorted out.

Process

The most steadfast of our three characters. Processes embody the sentiment "This is how it's done here." They are the codification of the operational steps necessary to complete some task or set of tasks. Good processes share some key attributes: they are as simple as possible, well documented, and properly communicated. Good processes can be an ally on the automation journey, while bad ones will most likely be a hindrance. Organizations have some level of processes defined for a dizzying number of areas within their IT environment. Some examples:

- Capacity-planning
- Change management
- Service management
- Performance management
- Problem management
- Security management
- User access management

- Backup and recovery management
- Disaster-recovery management
- Configuration management
- Release management
- Auditing

A few things to consider:

- As an organization transitions into the two upper levels of the framework (those labeled **Supervised Autonomy** and **Full Autonomy**), the system is expected to self-align. This means that most processes will need to be automated, so the evaluation of all of an organization's processes will eventually be required.
- There are many undocumented processes that exist in an organization. These may be embedded in "tribal knowledge" or job descriptions, but they are processes nonetheless and must be considered along the way.
- Breaking down processes into individual components and mapping them to the associated people and technology will be labor intensive, but necessary.
- In addition, processes should be mapped to the framework level at which they can be automated or eliminated.
- Evaluating and updating processes should be performed continuously. The goal is to eventually automate every process, where it makes sense to do so, within an organization.
- The long-term benefits of fixing or eliminating a bad process outweighs the short-term benefits of automating a good one.
- It is impossible to know whether a process is good or bad, if it can't effectively be measured.

Technology

The most dynamic of our three characters. Technology can provide tremendous benefits to any organization that effectively uses it. As discussed in the introduction, much of these benefits have been realized through software, which resulted in a complex landscape of layered software. This is not the only source of complexity in the age of digital business, but it can be the source of many obstacles during the automation journey. Then again, a journey without obstacles wouldn't be much of a journey. It would be more like a stroll.

There are an endless number of considerations when it comes to technology, but we consider a few of them below:

Data

As every aspect of the business comes online, everything is now able to be measured by leveraging data, including IT. Data has become an important and strategic asset for any organization. It has even been referred to as the new oil[5] , or as a capital asset. Data must be considered at every step of the autonomation journey.

[5] Clive Humby, a British mathematician, is generally credited with coining the phrase "Data is the new oil" in 2006.

A few things to consider:

- It is useful to think of data in terms of the DIKW pyramid. Where Data forms the foundation layer. As Data is processed and context is added, we transition to the Information layer. As relationships are established and questions become answers, we transition to the upper two levels (Knowledge and Wisdom).
- Data needs to be stored and protected. All the tasks associated with this must be automatable, so pick your data storage and data protection components wisely.
- Data will grow, so data storage and data protection requirements must be anticipated and not reacted to. Analytics are a must.
- As higher levels of automation are achieved, the information about your system will become nearly as important as the information within your system.
- Data has weight, so any decision to move data must be thought through.

Compute

No matter how complex the layered software environment becomes, it eventually comes back to hardware-based computation. There have been many developments in compute that affect today's landscape.

Packaging and consumption model:

- Before VMware introduced ESX, most organizations thought of compute as a physical server. VMware changed that forever and now the virtual machine stands in for the physical server.
- Before AWS introduced EC2, most organizations thought of compute as something to be bought or leased. AWS changed that forever and now, renting compute by the hour (or minute) is a reality.
- Since the release of EC2, AWS (and other public cloud vendors) have introduced new services that package compute in many new ways.
- The recent popularity of container platforms has also changed the way compute is packaged.

Location:

- The increased usage of public cloud services has changed where compute takes place. This impacts the next section (Connectivity) in particular.
- The amount of data that is generated far away from the datacenter has steadily increased over the years. The cost and latency impact of moving all this data to a centralized location can be enormous. The need to move compute closer to the source of the data has become a major consideration and so, the term edge computing entered the lexicon.
- With the advent of machine learning, new location-centric considerations entered the discussion. Training has different compute and data requirements than inference. Where each should be done is an ongoing question.

A few things to consider:

- With new packaging come additional components that need to be automated.

- With new consumption models, come new decisions about when and how computation should happen. Automation must be able to address this.
- With new locations, come new decisions about where computation should happen. Automation must also be able to address this.

Connectivity

When most people read the word ‘connectivity’ in this context, they immediately think reachability through a traditional TCP/IP based network. This is correct but insufficient when thinking about connectivity for automation purposes. Automation connectivity goes beyond the ability to establish a connection using a combination of protocols and port numbers (such as TCP/443, a well-known HTTPS port). It means that automation software has access to each action to be performed or aspect to be queried, and that the response data can be interpreted.

A few things to consider:

- If a component only provides access through a UI, then that component lacks automation connectivity. No matter how elegant the UI, it is an automation dead-end.
- If a component provides access through a CLI, then it isn’t an automation dead-end, but it is more cumbersome and increases risk.
- If response data from a CLI was designed for human operators (that is, text) and doesn’t have a machine-readable option, then it adds to complexity and increases risk.
- APIs are the best option today, especially those that are considered RESTful.
- If a component does provide an API, but that API doesn’t cover 100 percent of the required actions, then automation connectivity is limited, and may make it impossible to reach the upper levels of the framework.
- If polices that govern a system are human-centric and not interpretable by the system, then automation connectivity is limited. This may make it impossible to reach the autonomy Levels 4 and 5.

Conclusion

It is important not to overlook the near-term impacts of Digital Transformation and the skilled labor shortage it is creating. Automation does not represent the replacement of a skilled labor workforce, but instead, the transformation of that workforce towards higher-level and more creative work, as more remedial tasks are removed from their plates. This transformation can’t simply happen through the purchase of new technology. It must happen over time and as part of the automation journey. The three main characters in that journey – people, process, and technology – must all be transformed along the way.

The Autonomous Framework and this white paper intend to bring a greater degree of order and structure to the autonomous journey. It is our sincere hope that you have found some value in them.