SET Brazil

SMPTE

SINCE 1916

SMPTE Motion Imaging Journal

Covering emerging technologies in film, broadcast, and the connected media ecosystem

Good Things Come in Small Packages: Microservices for Media and the Need for Standardization

By Chris Lennon

Abstract

The concept of microservices is not new. The notion of breaking down traditional monolithic application program interfaces (APIs) into much smaller, business task-focused services is something that has been embraced by many industries. The media industry is just beginning to realize the benefits of this approach, particularly with the move to cloud-based services. We will look at why microservices are now emerging as a hot topic for media, where they make sense, and how standardization within SMPTE can help to leverage this trend for the benefit of the industry at large.

Keywords

Containers, interoperability, media, microservices, standards, virtual machines, workflow

We will look at why microservices are now emerging as a hot topic for media, where they make sense, and how standardization within SMPTE can help to leverage this trend for the benefit of the industry at large.

Introduction

Media workflow has for many years involved a compendium of diverse systems working together. How they have worked together has evolved significantly. As software-based systems become the predominant means of managing all aspects of media, we have embraced the IT world's forward march in terms of approaches and technologies.

In the infancy of media software systems, the state of the art was interchanged using files—not just video files, but data files as well. These files would be placed in watch folders, where processes would detect their presence and take action. We thought that was pretty advanced, but that was 25 years ago.

We then moved on to message-based interactions, rather than purely file-based, where one system could message another to initiate action. This was a huge step forward that was first advanced by the Broadcast eXchange Format (BXF) suite of SMPTE standards, which was new about 15 years ago.

Service-oriented architectures (SOA) came next, with application program interface (API)-based connections taking over. These too took a messaging approach to system interactions but encouraged vendors to publish APIs, providing standardized ways of getting data into and out of them as well as invoking actions within their systems. This was a significant advance, one pushed by initiatives like the Framework for Interoperable Media Services (FIMS) from the Advanced Media Workflow Association (AMWA). This was the latest and greatest approach starting a decade ago, up until fairly recent times.

All of these initiatives mentioned were attempts to design, define, and standardize based on a set of technologies that originated from outside our industry and then adapt them for use inside our industry, targeting greater degrees of cooperation than had previously existed. The motivation for this was clear—systems must work together to create efficient processing chains and workflows. No single vendor can afford to compete across all products and services required to support the content supply chain, so we had to find a way to pull together best-of-breed technologies from across the ecosystem. Those previous attempts were organized around cooperation at the operational level, meaning that the vendors themselves continued to design and produce technology in their respective commercial silos, and cooperation was the best effort at the macro level.

Today, as we move into virtualized environments and cloud, we need to go further. Operational and financial pressures are forcing us as an industry to find even greater efficiencies. Once again, we find ourselves looking for new operational patterns to bring into the industry that can create those opportunities

Digital Object Identifier 10.5594/JMI.2019.2917596
Date of publication: 23 July 2019

for efficiency. One of the most influential approaches that exist in the greater world of compute today is microservices. However, the reality of interoperability in a microservices world has fallen short of its promise. Can standardization help? We will explore that and more.

What Exactly Are Microservices?

Before going too far, we should all agree on the definition of microservices. Like many industry buzzwords, it seems that everyone has a different definition. This can be dangerous, as we can each be talking about something a little different when referring to the same term, which doesn't help when we are dealing with interoperability. So let us level-set the basic definition of the term *microservices*.

For direction, we look to a recognized body that has already done the work of constructing a well-thought-out definition. The National Institute of Standards and Technology (NIST) provides the following definition in their Special Publication 800-180[1]:

> A microservice is a basic element that results from the architectural decomposition of an application's components into loosely coupled patterns consisting of self-contained services that communicate with each other using a standard communications protocol and a set of well-defined APIs, independent of any vendor, product, or technology.

The key words here are "basic element" and "self-contained services." Microservices are, by definition, services that are focused on a single well-defined task or capability.

Microservices communicate with one another using an interprocess communication (IPC) mechanism. These can either use an asynchronous messaging approach or one that utilizes synchronous requests and corresponding responses. IPC protocols like HTTP (for synchronous approaches), Advanced Message Queuing Protocol (AMQP) (for asynchronous approaches), or Transmission Control Protocol (TCP) are the most common.

In its special publication on microservices, NIST goes on to specify that microservices are built around capabilities, built on SOA concepts, and are implemented using Agile technologies. But is this enough? As a technical definition of the philosophy and approach to building a microservice, that definition suffices to convey the core tenets of microservices. It stops short, however, of defining a solution for our industry.

In other industries, microservices are built around transactional systems where failure and recovery encompass finite units of data, each of which most likely has a commoditized value. For our industry, we require greater attention to detail, as we deal in data and content of very high value. We operate under tightly controlled timing constraints, and efficiency is always a compromise between cost and performance. Also, as an industry, at the business level, we need to think beyond the technical definition and focus on the outcome of embracing a microservices architecture. To that end, we simplify the definition and focus it on the result, not the journey.

A microservice is the smallest service or process that can be implemented that still serves a viable, performant business requirement. As a network service, it exposes standardized APIs and communicates with other microservices using industry standard protocols.

The focus on minimalism and the business outcome is important, as it helps us hone in on the core values of moving to a microservices-based media architecture. First and foremost, microservices are by definition minimalistic. The focus on the smallest unit of viable performant business value means that we are carrying less risk as we deploy a solution, lower overhead, less baggage, and tighter focus. This, in turn, speeds deployment times, lowers risk on any one piece of technology, improves performance, and potentially provides for choice. It brings us full circle to how the industry typically operates, pulling together best-of-breed technologies from vendors of complementary functionality.

Another benefit of microservices is reuse. As the microservices themselves become deployed more often, we begin to build trust and recognize the value of that particular function as part of multiple processing chains. This, in turn, promotes the reuse of the best components across a wide variety of workflows; an outcome that just is not possible using traditionally designed applications. Additionally, the notion of working with smaller units of functionality promotes selective scale. With each function isolated into a smaller deployment model, it is by definition utilizing the least amount of resources possible. This more directly aligns resources to resource utilization. If you have a smaller service, you can run it on less costly infrastructure, and, conversely, specialized infrastructure requirements such as graphics processing unit (GPU) or high-performance storage can be more tightly allocated and colocated next to the associated microservices.

To fully realize the value of microservices, we need as an industry to take an additional step beyond what Google, Amazon, and Microsoft have done to define the best practices around microservices and wrap our industry's requirements around those definitions. This requires taking the benefits of microservices and applying them to the unique requirements we have in media. Standardized media microservices allow implementations to exchange data with each other, using and leveraging the standards and practices of the industry.

While some vendors have already developed their own microservices within their own suite of products,

it was becoming clear that what was needed to ensure interoperability at a microservices level were standards. This is where SMPTE comes in. A new effort under its 34CS Technology Committee on Media Systems, Control, and Services is targeting this specific area.

Microservices and Containers

A key concept that is closely associated with the deployment of microservices is that of containers. A container, or application container, is the encapsulation of the unique requirements of each process or application into a functional subset of its running environment. Unlike virtual machines, where the entire operating system (OS) must be deployed just to deploy one application or process, a container only requires the minimal subset of dependences that would allow the application to run. This is perfect for running microservices. This allows the deployable unit, i.e., the container, to be much lighter weight, easier to maintain, and more rapidly deployed than a virtual machine. It also allows, where appropriate, for multiple microservices to be deployed on a shared instance of an OS.

Each application container is isolated from other application containers, allowing them to independently start up, as well as scale up and down. This is a key advantage of microservices.

Are containers absolutely required for deployment of microservices? No, but they do provide a nice way to manage and deploy them. The container helps to manage resources (computing environments) in which the microservices will run. This allows microservices to take full advantage of virtual machines, which we will discuss next.

Microservices and Virtual Machines

In parallel to the progression of technology that has brought us to microservices is the subject of deployment and provisioning of applications and processes. The early days of virtualization utilized a concept of virtual machines, which are just as they sound. A virtual machine is a software implementation of a complete platform that enables execution of the OS and the corresponding applications in a cloud environment. Virtual machines can be utilized to deploy microservices and are fully compatible with microservices but technology has advanced on this front as well.

Of Monoliths and Molecules

When asked what differentiates a microservice approach and traditional SOA approaches, the simplest response is that services in a pure SOA world tend to be monolithic, and microservices are the opposite—each service is focused on small, independently deployable capabilities.

SOA has traditionally consisted of APIs with many input and outputs. Think of them as multipurpose services. A good analogy would be a Swiss Army knife. It is incredibly capable, but, if you have only one specific task to ask of it, perhaps it is overkill.

In microservices, you would have a separate tool for each job the Swiss Army knife performs—a screwdriver, knife, corkscrew, and so on. You could then have microservice modules for specific jobs or tasks, like remove a screw, cut, or remove a cork. The idea of getting a microservice defined at the appropriate level of granularity is to boil it down into the smallest independent useful tasks that constitute that process.

However, like many things that become more complex upon closer study, the differentiation between traditional SOA and microservice architecture may not be quite that simple. Microservices themselves are based on SOA, so saying that the two are totally different things is not quite correct. It may be more accurate to say that microservices are the next logical step in the evolution of SOA.

Thus, a big differentiating factor between traditional SOA approaches and microservice approaches is granularity. When a service is defined at a granular level that takes it down to a specific action of business goal or function, then it is a microservice. A microservice's level of granularity is often considered the smallest unit of execution that can be put into a network, which on its own has tangible business value.

By moving toward these very small services, which perform larger tasks when coupled together using IPC mechanisms as outlined earlier, many advantages are realized. Single microservices can be reused by multiple processes, which is certainly efficient. Their limited scope means that they are inherently simpler, quicker to deploy, and can be rapidly and reliably tested. The risk of deploying new microservices is thus also reduced, leading to more stable system deployments.

Microservices and the Cloud

One of the primary motivations for the adoption of microservices across all industries has been the cloud. The main reason for this is the unique ability of cloud implementations to rapidly scale up and down as needed, providing previously unheard of flexibility to users.

Formerly, when scale up of capabilities was required, users had to purchase, install, and support additional hardware in their facilities. Even then, there were often limits to how much was possible. To scale down was almost as painful, as equipment formerly needed in a larger scale implementation had to either be removed or simply sat idling, costing users real money.

Having microservices running on a public cloud properly aligns the cost of infrastructure to the scale of the microservices that should be run. Users can now scale up and down as needs change. Because you are not limited to the resources of a single, physical machine, this scalability is a key reason why the true realization of the power of microservices depends on them running in a virtual machine environment. Organizations

can share resources across multiple systems, applications, and even departments, adjusting as utilization levels change.

Of course, private cloud environments can enjoy many of the benefits of public cloud services. Private clouds can make sense in many cases where a cloud-based deployment is desired but there are reasons that a public cloud might not be desired. This is not unusual in larger organizations, in particular. Such organizations often already have large data centers that can be used for private cloud hosting. A private cloud environment can also be quite attractive to organizations and/or applications that are predictable in scale and for which performance is critical. For this style of installation, it can make sense to utilize the benefits of cloud deployment methodologies while directly managing the overall performance of the network. Another benefit of the private cloud is colocation to other services that are directly connected to the core infrastructure. Finally, use of a private cloud can help with speed and response times, as in some cases, the closest hosting site for a public cloud can be some distance away from an organization's site, impacting these items.

Standardization's Role

It should be clear by now that microservices have a plethora of advantages. It makes sense that so many vendors are basing their next-generation systems on them. However, the real power of microservices is not truly unleashed until they have some level of standardization, and thus interoperability across vendors and platforms. This allows users to select best-of-breed services from the vendors who provide them, building complete systems that provide the user with the perfect fit for their needs today and the ability to modify that at will in the future.

The traditional purpose of standardization over the years has been to foster interoperability across systems. In SMPTE's case, this goes all the way back to its very first standards, specifying things like film width, image format, perforations, and so on. Without SMPTE's standards, many of the basic tenets of media interoperability we take for granted today would not exist. Just a few examples include timecode, cable and connector specifications, media file interoperability (MXF), and metadata exchange (BXF).

While great benefit can be realized via microservices within a specific vendor's ecosystem without standards, what happens if you want to utilize solutions from multiple vendors? This will always be the reality of the media business, as no single vendor (despite many attempts) provides best-of-breed solutions from production all the way to distribution.

This is where SMPTE and standards come in. The goal here is to provide a standardized framework for microservices that all vendors can point to with their implementations. Using this approach, true interoperability can be realized, both between applications and across cloud environments, including private clouds, and across multiple public cloud environments.

A Microservice-Like Approach to Standards Development

At the most basic level, when a group within SMPTE decided to tackle microservices for media, several architectural items were identified as needing to be addressed first. This creates a foundation upon which the microservices themselves can be built.

Many of these architectural items will reside in a standards document, containing normative requirements that implementers must follow to ensure interoperability. As a companion to this standard, a recommended practice (RP) or engineering guideline (EG) is anticipated, containing more informative information and recommendations for implementers.

Next, an agreement has been reached on the need for decomposition of media microservices into logical areas of focus. This is anticipated to be an ongoing effort that will require some form of governance, as new services are likely to be added on for years to come.

Finally, the actual definition of the microservices themselves must be completed. To manage this properly, the best approach is to avoid a monolithic approach to document development (just like microservices do). Trying to define all of the microservices needed to manage all aspects of media management could be a bit like trying to boil the ocean. The task is far too large and complex to manage all at once.

As a result, the approach the SMPTE Drafting Group plans on taking is to potentially utilize an open source approach. Many companies have already created and deployed microservices for media successfully. It makes sense to tap this pool of already-defined microservices. The intent is to provide an open forum for anyone to submit microservice(s) for consideration. A governance body within SMPTE would then review and consider each of these, approving those that fit into an approved register and filtering out those that do not.

We also believe that some more agile tools in SMPTE's toolbox may be useful in this exercise. Registered Disclosure Documents (RDDs) allow for quick approval and publication of already-deployed capabilities. Technical Specifications (TSPs) are a brand new lightweight path to documenting specifications for industry use within SMPTE, which have the benefit of a clear path to later standardization, should that be desired.

By utilizing some traditional approaches (standards, EGs, and RPs), coupled with more agile ones (RDDs, TSPs, and open source), we believe SMPTE can provide a level of standardization that makes sense while remaining agile, which is critical in this area. This layered approach is shown in **Fig. 1**. This allows the

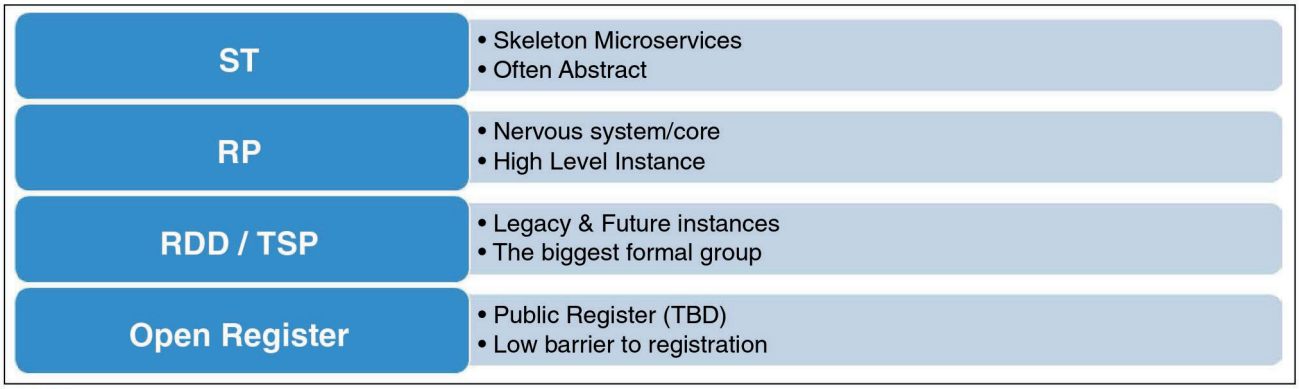

FIGURE 1. Bringing SMPTE's many tools to bear.

simultaneous development of many microservices, as shown in **Fig. 2**.

This approach addresses the boiling the ocean concern. By allowing the experts in each area to submit their already-developed microservices, and providing a framework into which they can all fit and interoperate, SMPTE will be able to rapidly develop a wide-ranging suite of microservices, realizing the promise of plug and play of best-of-breed offerings across the full media ecosystem.

For example, let us assume that one vendor has an excellent set of microservices that together address the needs of program management for a linear channel. Another vendor has an equally excellent set of microservices that focus on the full ad sales chain. If both were willing to submit their suite of microservices to SMPTE for inclusion in its approved set of microservices, and were willing to have these microservices adhere to SMPTE's basic microservices architecture, specified in its microservices standards, we could quickly end up with a set of microservices that service much of what is needed on the business end of standing up the components of a linear channel. Another example could include marrying a best-of-breed standards convertor and ingest solution or a playout server with a third-party encoder microservice.

What you end up with here is a virtual puzzle, as shown in **Fig. 3**, with each piece consisting of a set of microservices that fit together to fill a specific need. These puzzle pieces then fit together to fill a larger need. Ultimately, the hope is that the puzzle is complete, with no missing pieces, offering users a full set of interoperable microservices to address their many, diverse needs.

There are endless other examples of microservices being used in the media industry. It is hard to imagine

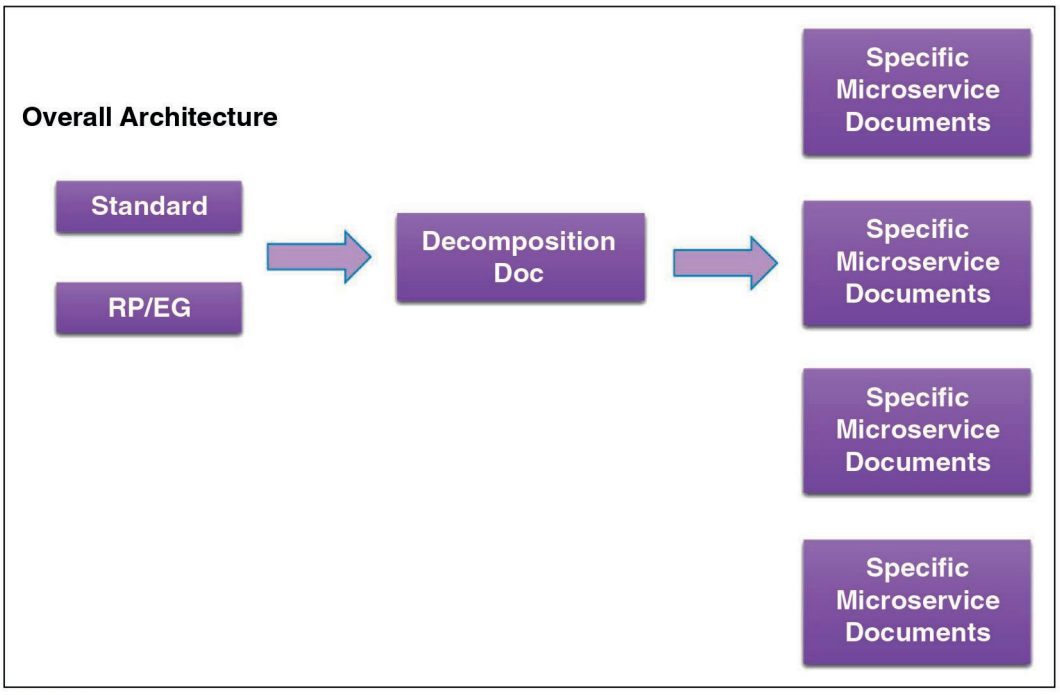

FIGURE 2. An approach to document development.

FIGURE 3. Microservices fit together like puzzle pieces.

an area that is not well-suited to them. From production, program planning, ad management, and schedule/rights management, all the way through the media supply chain to distribution, microservices can be brought to bear in a wide variety of areas.

Many Inputs

As with most standardization efforts, true success is realized through the diversity of participants and inputs into the process. The ability to pull the best and brightest in the industry from several disciplines and companies makes the end product not only better but more likely to be embraced by the industry when published.

This is true with SMPTE's microservices effort. Inputs have been considered from sources literally around the world. The Media Microservice Edge Consortium (MMEC) began work in 2017 in defining a set of microservices for the media industry. While initially focused on the Chinese market, it quickly became clear that, to be successful, the effort had to go global. With global vendors involved and the desire for an end product that could benefit media companies worldwide, the MMEC sought to provide its findings as input to the SMPTE process.

The group has also examined inputs to its work from the AMWA—relating to the FIMS project, NIST, Imagine Communications, and the European Broadcast Union (EBU). Other inputs are welcome and expected as the group progresses in its work.

Conclusion

The logical question at this point is where do we expect to be at the conclusion of this work? The ideal scenario is that the industry ends up with a standardized means that users can select from among best-of-breed solutions for each service defined.

Consider the analogy of building a car. If one were to begin with a blank sheet of paper, the task may seem overwhelming, with the thousands of parts that must fit together and combine to create an overall system that gets you from point A to point B.

An Agile approach to building a car is to break the process of building a car into the smallest logical components. This allows the selection of the best-of-breed producers of each component to be used. Let us say that a company provides the best seats, another company provides the best display screens for the dashboards, and yet a third provides the best sheet metal for the body of the car. Although the overall car is one product, it can benefit from selecting each of these providers for the various components. In the same way, each of them in turn can break their processes down into subprocesses. Take the seat manufacturer, for example. They may use one supplier for the upholstery, another for the basic structure, and yet another for the electronic controls. And so it goes.

By using this approach, the process of building a car becomes much simpler. It ends up being a matter of invoking all of the subprocesses from the various manufacturers at the right time and having them all come together to form a fully assembled car. It all works as long as you have a framework; in this case, a design, but in our industry the standards to pull it all together.

Now, apply this analogy to the media operations of a large network and you can see how this approach could be quite beneficial. Expertise and capabilities onsite no longer need to cover every aspect of media operations. Instead, the focus shifts to choosing and integrating all of the right pieces into a single cohesive flow that is easily modified and scaled as needs change.

A media industry with standardized microservices is one that is well-positioned for a future that is centered around being quick to adapt, not only to new systems and ways of doing things but also to the need to add and remove consumer-facing outlets almost on a whim. Many see SMPTE's work in this area as crucial to the media business of tomorrow.

The end result of SMPTE's efforts in this area is still taking shape. We have been successful in getting the participation of dozens of organizations in helping to determine what problems we should try to solve here, and even more fundamental, what can we solve?

The goal should be to standardize the minimum that is needed to enable interoperability. Beyond that, pointing to standards developed in other standards development organizations (SDOs), best practices and perhaps work that falls short of standards but still assists in truly interoperable implementations of microservices may be appropriate.

We are just beginning the journey, and it is not too late for you to join. If you have the expertise and you would like to contribute to the cause of standardizing microservices for the media industry, please

consider joining the 34CS Drafting Group on Media Microservices Overall Architecture. Much work lies ahead, and we need your help to take an agile approach to scaling up our effort to produce these much-needed services in a way that will foster interoperability across media systems.

Acknowledgments

We thank the members of the SMPTE Drafting Group on Media Microservices Overall Architecture for their help in leading SMPTE into the world of microservices. We also thank Brick Eksten, CTO of Imagine Communications, for his insight and guidance on the topic of microservices.

Reference

1. A. Karmel, R. Chandramouli, and M. Iorga, "NIST Definition of Microservices, Application Containers and System Virtual Machines," NIST Special Publication 800-180 (Draft), Feb. 2016.

About the Author

Chris Lennon serves as the president and CEO of MediAnswers, Monument, CO, which specializes in media software systems consulting and development. He has been in the media business for more than 30 years. He also serves as standards director for SMPTE. He is known as the father of the widely used Broadcast eXchange Format (BXF) standard, as well as the OBID standard for audio watermarking, among others. In his spare time, he is an accomplished racer, an author, and a high-performance driving coach.

Presented at the SMPTE 2018 Annual Technical Conference & Exhibition, Los Angeles, CA, 22–25 October 2018.