www.pipelinepub.com
Volume 20, Issue 7

The Emergene of Common Platforms

By: James O'Brien

The concepts of OSS (Operations Support Systems) and BSS (Business Support Systems) originated in the telecommunications industry. Historically, they were largely separate ecosystems. Above the level of hardware (often not even that), they had little in common. Over the last fifteen years, however, the differences have become far less pronounced with the emergence of cloudification/cloud-native (containerization) and SaaS.

Cloudification and new methods, such as adopting containerization and microservices, bring challenges as architectures become moving targets. These new methods require new skill sets and technologies. In modern environments, the goal is to build common platforms and practices that can cover as many domains as possible using, if not a single platform, then at least a single approach.

This is not a problem exclusive to the telco industry. It's important to note that these issues are pervasive across various industries that heavily rely on distributed infrastructure, such as energy and manufacturing.

BSS, OSS, and Network are not so Different Anymore

Cloud, OSS, BSS and (Core) networks now use the same deployment architecture. They run or aspire to run as much as possible in containerized environments. Moreover, the convergence of BSS, OSS, and Network domains is not just a technical shift; it is a monumental transformation shaping the future of OSS/BSS.

OSS and Networking are now very closely related. Many classical OSS functions have shrunk or been replaced with standard tools used in container environments, such as Prometheus/Grafana for monitoring, Terraform, Ansible, or ArgoCD and so on for infra and container deployment. Although these domains have retained a high level of separation regarding the domain knowledge needed to understand business & networking needs, most of the underlying technologies required to run them have converged.

The challenges, however, with cloudification, public or private, such as managing distributed computing infrastructure and networking, meeting resiliency and latency requirements, and complying with legal and regulatory obligations, are significant and require careful consideration.

What Changed with Cloudification

In today's dynamic environments, systems are increasingly adaptable. Businesses now expect greater agility, a significant driver behind the shift towards cloud technology. As a result, tools like Kubernetes have become standardized essentials in this transformation.

With cloudification the differences between OSS and Networking have arguably all but disappeared. NFVs (Network Function Virtualization) and OSS apps are primarily deployed and managed similarly. Many tasks that classically required an OSS tool can be covered by (now) "standard" cloud automation and observability tooling. CI/CD (Continuous Integration/Continuous Deployment) has also become the name of the game. This is a set of practices that enable frequent, automated changes to a codebase, ensuring that software is always in a releasable state.

The advent of these new technologies necessitates the development of new methods to harness their potential. Understanding how to extract value from these technologies requires significant time and effort. Modern platforms and development require massive amounts of packaging and wrapper code. Most of the work involves automating the deployment and management of application environments, such as Git repos, Terraform, networking, and so on.

Challenges in Modern Environments

Telcos and other infrastructure-intensive industries have a strong focus on engineering and operations. This makes the engineering-heavy approaches found in cloud and cloud-native very attractive to a telco audience. However, there are some serious risks to this.

Problems of Scale

Large, complex enterprises implementing large programs require methods and approaches that work at that scale. Most of the proven methods in cloud-native technology have focused on solving problems at a (relatively small) scale. While success stories often involve companies that operate at massive scale, such companies typically do not have complex business models in regulated industries and were able to grow organically. In other words, they did not have to start big and they were driven by few applications.

Telecommunications and other regulated industries, on the other hand, face significant constraints and challenges. They must start big, initiating projects at an enterprise scale, and cannot deploy transformations incrementally, such as replacing an Integrated Management System (IMS). Stringent regulatory requirements for critical infrastructure also bind these sectors and their more complex and specialized technical applications.

Typically, these projects are centered around something other than application development, which is often sourced from Independent Software Vendors. Instead, they focus on system integration, constructing private or hybrid clouds, and developing pipelines to deploy code from other companies. While public clouds have many advantages, they also present challenges. Managing them at scale is particularly complex because it requires translating an organization's structures and rules into security rules in the cloud. This comes even before optimizing the platform from a financial perspective, which has become a discipline in its own right (FinOps). Skills in the cloud space also tend to be cloud-siloed, which means advice and architecture are not necessarily neutral but heavily influenced by the cloud provider.

Another challenge is regulation. OSS & Network platform owners have obligations and requirements that they cannot easily outsource to a public cloud. In these environments,

moving to the public cloud is a complex endeavor. BSS systems owners also have their own regulatory and legal obligations that can add complexities, though they are usually less related to the properties of the cloud itself.

Standards won't save you. Telcos are used to well-defined standard architecture frameworks. These work well as process/functional placement models, but managing cloud-native software (in a private cloud in many cases) creates a lot of additional complexity. When architecture is dynamic, the challenge is managing automation and deploying the infrastructure and applications, which is not defined in the standards.

Scale and & Skill Sets

Iterative approaches to project and program management have become commonplace in most industries. Although there are many variations, these methods view production as a decentralized but industrial process.

In the modern landscape, favored skill sets include DevOps cloud-native practices and agile methodologies. These approaches typically emphasize smaller, tactical teams engaged in custom development and code packaging.

In many projects, however, the definition of "done" becomes elusive, as solutions and functionality may lack clear definition initially. In technical, infrastructure-centric environments like networking, there is significant risk due to minimal input into requirements, often with few or no direct users. These large, capital-intensive projects result in extended timelines and feedback loops. Additionally, solutions such as those for network switching tend to be complex and abstract.

Having enough constraints on teams to ensure standard approaches and consistency is better than allowing total freedom. Platform engineering has evolved as an approach to putting guardrails on DevOps to do this.

Architecture and Scale

Architecture should be a key concern in large projects as without a solid approach to architecture, the definition of "done" needed for program management does not exist and is not verified and re-verified.

Moreover, the massive emphasis on application architecture can lead to the neglect of higher-level architecture because it's not mapped to features for individual products in the program. The dangers of this might not be immediately apparent. While using program-level architectural principles and approaches can help, there still needs to be a strong architecture team.

Highly structured environments like telco networks did not tend to have much need for Enterprise Architecture (EA), because transformation programs are moving to environments where standards do not address many aspects of EA.

In the case of agile projects, project management rather than architects usually ensures the execution of the delivery process. This can lead to little time for architectural work. Also, the roles of lead engineer and architect can become confused. On a small scale, engineering skills are more important. However, in a large project, this can become a huge issue.

In modern projects, architecture is not static; it needs to be regularly revisited and revised. Paradoxically, this can get neglected, especially at a program level in these kinds of projects.

Pitfalls and Program Management

Domains in telco and other industries have become very similar, and there is a much more significant technology overlap than existed previously. New technologies require new approaches to managing them. But while adopting new methods, architecture management at a program level still needs to be maintained and aligned with program management.

New environments and methods can be complex to manage, and it can take time to understand how a project is doing. Some things to watch out for are: 1) Technical discussions dominating program-level discussions signify that large-scale architecture and operations management still need to be fully considered. 2) Activity and progress towards completion do not sync. Progress and targets appear just within sight but are never quite achieved. Again, this can be a symptom of a disconnect between architecture and program management.