Appledore Research Group

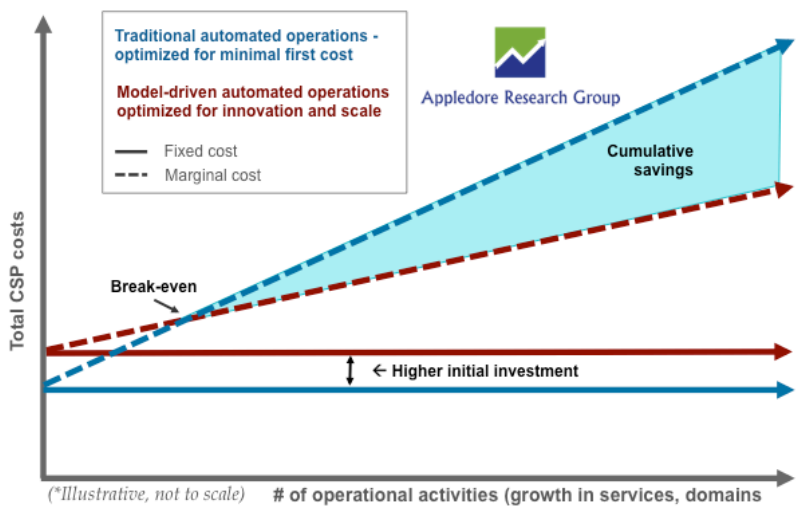
CROSS-LAYER, HYBRID ORCHESTRATION –

THE QUEST FOR OPERATIONAL AGILITY AND MANAGED RISK

White Paper
in partnership with:

ORACLE COMMUNICATIONS

February 2017



2/20/2017

Grant Lenahan, Appledore Research Group

www.appledoreresearch.com

Published by Appledore Research Group LLC • 44 Summer Street Dover, NH. 03820

Tel: +1201 602 4702 • Email: grant@appledorerg.com • www.appledoreresearch.com

CONTENTS

EXECUTIVE SUMMARY

This white paper examines what is necessary to move Orchestration beyond narrow technical success (e.g.: establishing virtual network functions and simple network services) to fully implement agile operations across all layers. Cohesive end-to-end operations is necessary to realize the agility and cost savings that are the core value proposition of Virtualization – which we define as including Network Function Virtualization (NFV), Software-Defined Networking (SDN), XaaS and "Digital Services"[1]. Quite simply: without agile operations CSP investments in a new, virtualized network will fail to realize the savings and benefits. To tackle this topic, we must expand our field of vision, beginning with a big picture look at the economics of our industry, and then looking at operations more broadly – from the role operations play in innovation and scaling the business, to the functional roles at play.

This paper provides a look at the opportunities, risks and best practices when implementing orchestration that will likely be a keystone of tomorrow's operations.

In the past, there was only modest incentive to invest in truly agile operations and/or operations and business support systems (OSS/BSS) due to rigid network infrastructure. Virtualization, in all its manifestations, lifts many of these limitations, promising to dramatically lower both CAPEX and OPEX, with profound implications for how network businesses can and should be run. These technologies also promise speed and agility that can benefit revenues, business models, service availability and therefore customer satisfaction.

The benefits of Virtualization depend on efficient and end-to-end operations that extend well beyond the simple instantiation of virtual network functions (VNFs) and network services– they demand flexible orchestration at every layer, from resources up to services and products, and smooth interaction with other OSS & BSS functions, including order management, assurance systems, and billing/commercial processes.

This paper focuses on orchestration in particular, identifying the capabilities and best practices necessary to fully realize the benefits of investments in network agility. We conclude that orchestration must have three major characteristics:

1. *Re-usable components*: modular, re-usable orchestration functions enable the dynamic generation of orchestration plans
2. *Self-realizing components*: Services and Network Functions must be self-realizing, leveraging well encapsulated data-driven rules and stateful principles
3. *Multi-layer*: Capable of spanning all layers of business and technical operations, including instantiation, configuration, and dynamic management of resources, network functions, end-to-end services and products.

Any highly manual or disconnected operations layer becomes the weak-link in end-to-end agility.

[1] All four types of "Virtualization" offer the same values – instant and automatable creation, scaling, healing, and therefore the ability for business to be far more agile and automated.

BUSINESS DRIVERS – THE ECONOMICS DRIVING CHANGE

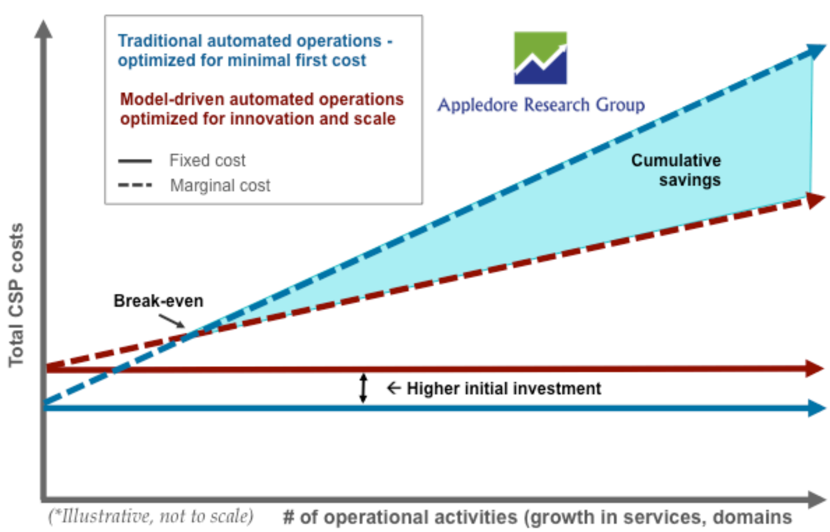
Virtualization promises agility, reduced costs, hardware consolidation and other attractive benefits. Yet, if CSPs are to realize these benefits, they must re-think their business models, assumptions and operations processes to exploit *scale* and *automation*. Absent those changes, deploying agility in the network will deliver only a fraction of its promise.

Historical operations methods (end-to-end workflows, often siloed) have resulted in the growth of these three tasks, which further increase costs and reduce agility:

1. Maintenance grows with the cumulative number of services (and operations scripts), which increases every year.
2. Integration costs grow even faster, driven by the need to integrate (potentially) every system and sometimes every workflow to every other system (up to N x (N-1) integrations!)
3. Self-contained workflows simplify initial development, but re-use is nearly zero. These, limiting agility and leading to the high integration and maintenance costs noted above.

If the promise of Virtualization is to be realized, CSPs must implement operations that are geared to minimize *cost at scale* in a world of highly customized services. This means turning away from operational designs that are expedient and that minimize the *first cost* (initial investment) of a limited set of use cases. Rather, we must design operations for a world of ongoing innovation in services and products; of mass customization based on unlimited order and partner parameters; of self-realizing, healing and scaling network functions and services –based on re-usable, highly flexible models that react dynamically to those conditions. This path has already been taken by hyper-scale web players with great success.

Figure 1: Designing for Scale and Uncertainty: trading off an initial investment for lower variable costs – and therefore significant ongoing savings and agility



Source: Appledore Research Group

Some key takeaways to Figure 1 include:

- Virtualization demands that we design for scale in operations.
- Modest initial investment yields large pay off in lower costs and greater agility over time. *(Investment includes layered, object-based, model and policy-driven orchestration)*
- This new cost structure makes many new products and business models economically feasible.

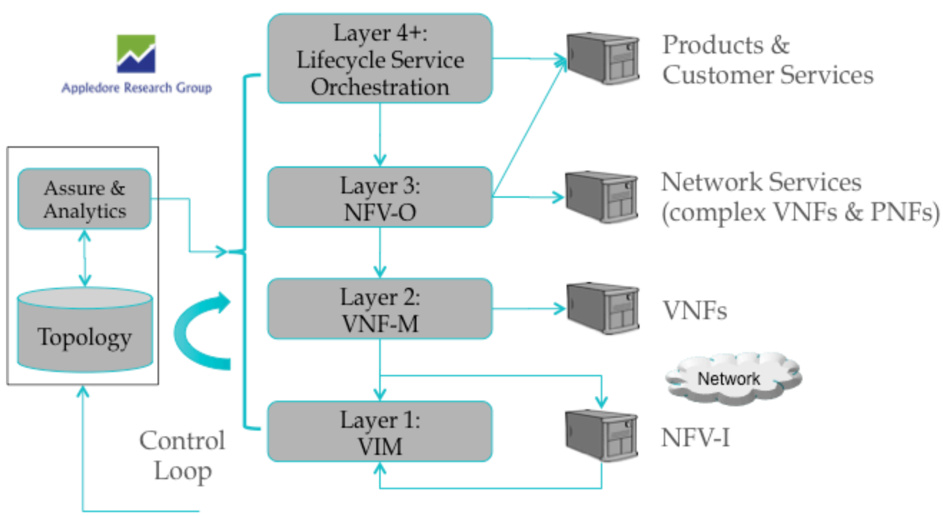
In almost every case, for a single service, it is simpler and faster to diagram out the operations steps (labor, configuration, whatever) and create a custom solution. It minimizes dependencies on existing systems, and requires little investment in "infrastructure" – e.g. hierarchical catalog structures, carefully built objects, and data structures that are generic rather than tied to a specific goal. Yet, over time, custom solutions become complex and costly to manage. This dated approach becomes increasingly costly with scale.

The essential take-away is as follows: we got to where we are making logical, near-term decisions; these decisions have had much longer-term implications than were anticipated; and finally, we can't always anticipate what changes will take place.

TAKING A BROAD PERSPECTIVE ON ORCHESTRATION

Operations are a tremendously broad domain – extending greatly beyond the ambition of ETSI MANO today. Consequently, the industry must view orchestration broadly – its impact extending from innovation through the life cycles of resources, services, and the commercial product offers. A fundamental best practice is that the orchestration processes become shared resources across many business processes, from fulfillment through assurance and capacity expansion.

Figure 2: End-to-End Agility Demands Coordinated and Abstracted Orchestration across Multiple Layers



Source: Appledore Research Group

Figure 2 illustrates this reality – note that orchestration occurs at various layers, from the lowest infrastructure, to the commercial orchestration necessary for products and customer services. Assurance and analytics provide data and intelligence into orchestration at various layers, to initiate automatic scaling and healing.

There is a critical corollary to this best practice: orchestration must be able to react, automatically, to unknown future events, conditions and requirements. For example, it must accommodate the specific needs of an order. Similarly, it must react to network conditions and events (congestion, resource failures). And it must do all this while reducing – not increasing – complexity and risk.

TODAY'S ENVIRONMENT: SEMI-STATIC, HIGHLY ENGINEERED OPERATIONS

Traditional OSS approaches have two broad categories of weaknesses:

1. They have typically been created as self-contained, end-to-end procedural work-flows ("silos") rather than from re-usable objects
2. They are prescriptive rather than adaptable based on models, policies and context

Silos

The first weakness, end-to-end self-contained procedural workflows, results in low re-use. Low re-use, in turn results in maintenance costs rising first with the number of instances, and second with the number of integration points, which can be as high as N*(N-1) where N = the number of systems/workflows that may communicate with each other. In effect, as we create more independent procedures, without regard for re-use or modularity, the *costs rise faster than the number of services.* Worse, those costs re-occur nearly forever, since major systems are rarely retired.

It is worth noting that the cost of creating two independent procedural workflows, maintenance aside, is essentially twice the cost of creating one; in other words, innovation costs rise linearly with the number of service types or "procedures". When we add both development and ongoing maintenance costs together, the end result is two sets of costs -- one that increases as fast as innovation progresses, and another that increases faster. This means *negative* economies of scope.

Prescriptive Workflows

The second weakness is the prescriptive nature of those workflows, built end-to-end for a service, as opposed to parametric, adaptable orchestration driven by models, rules/policies and context. The problem with prescriptive workflows is that they are inflexible, and may demand human intervention. Dealing with several scenarios requires multiple workflows, or at least branches. The need for human intervention and multiple workflows can be overcome with brute force, but another limitation of them cannot: *prescriptive workflows demand that we know, in advance, the situation we are designing for*. We must, in effect, guess at the future. History says we're not good at it.

Impact over Time

In a simple, initial analysis, these architectural weaknesses (silos, lack of re-use, prescriptive and inflexible workflows) appear harmless – technical niceties that have been overlooked for solid business reasons – speed and cost. However, if we look at the result over time we see that the root cause of escalating cost, complexity and sclerosis is in fact our collective lack of architectural rigor.

TOMORROW'S ENVIRONMENT: AGILE, AUTOMATED & FLEXIBLE. (OR NOT)

There are two parallel critical paths to agility and low cost. First is the path through the network, which is becoming flexible, configurable, and "on demand". Second is the path through operations, which must be similarly flexible and dynamic. Virtualization creates a compelling opportunity to transform operations and business models – where there will be fewer "hard limits" imposed by the network.

Enabling the Digital Marketplace

On-demand services, digital collaboration, and mass customization will all be essential tools in the quest for new revenues market share and markets. Clearly, all of these require highly customized operations that allow services to be instantiated, automatically and dynamically, in response to order parameters, infrastructure capacity and network state. We call this "context aware" instantiation. Policies and rules, together with model-driven parametric orchestration, are the foundation for context-aware operations and therefore self-realization.

Mass customization, on-demand services and myriad digital collaboration partners inevitably means many more customized services, each of which demands unique orchestration. The traditional approach would be to create several procedures, and anticipate the most common options. A much better approach would rely on orchestration that reacts to the individual context of an order, self-realizing based on those specifics (and available capacity for example). Automation and self-realization will be essential if we are to achieve both the scale and low costs needed to make these semi-custom services profitable, and to compete with over-the-top (OTT) players.

Fostering Innovation

Beyond that however, we must also consider the industry's capacity to innovate. New product offers and new services require human creativity. In today's environment, such creativity requires technically-fluent service creators, followed later by highly specialized operations experts to create the procedural process. This clearly does not scale nor is it cost-effective.

Tomorrow's promised economics demand that business managers and creative professionals be able to create new products and services independently. It also means that the operations "objects", from configuration requirements to monitoring methods to critical threshold-crossing specifications, should be predefined such that these services and their operations are self-realizing. But flexibility and self-realization come from careful planning, and complete on-boarding of "well enabled" VNFs and services, meaning that specifications, configuration requirements, testing methods, and other necessary data must all be on-boarded along with service and resource components. Such completely "onboarded" objects allow both the assembly of services by nonexperts as well as the assembly of associated operations logic, eliminating the need to engage technical operations experts.

AN ILLUSTRATIVE EXAMPLE:

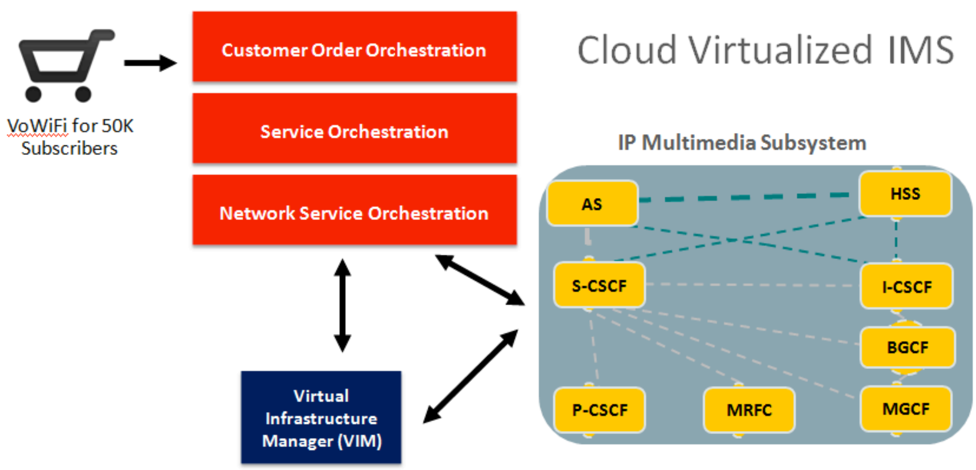
Terms like "model driven", "policy driven" and "context-aware" can be abstract. The following, very simplified example, is intended to make these terms and their importance more concrete. Consider a customer service that uses "infrastructure" provided by a network service: Voice-Over-WiFi built using a vIMS network service.

Each time an instance of the vIMS network service is instantiated, so are certain, dedicated supporting vIMS components (e.g.: AS, BGCF, CSCF, PCRF and their connectivity). The intent of such unique instances could be to tailor the size, performance, and other attributes to the service's or customer's needs, while also maintaining separation to ensure (for example) security or performance independence. An example of such specific needs might be a government agency or enterprise that wants its calls segregated for security or performance reasons.

Whenever a "VoWiFi service" instance is created, several layers of orchestration and configuration are likely to occur:

- *Orchestration of the product* (orchestrating commercial aspects of the service, such as billing)
- *Orchestration / configuration of the overall VoWiFi service* (this might include deployment and configuration of physical and virtual components such as wireless routers, gateways, in addition to the vIMS)
- *Orchestration of the VNFs and their connectivity* (these constitute the network service instance)

Figure 3: Simple View of Order, Network, and Multiple Layers of Necessary Orchestration



Source: Oracle Communications

Each time a "vIMS network service" is created, many "context" constraints must be considered[2]:

- Are there geographic proximity or latency requirements to consider?
- What scale does it demand?
- What SLA(s) must be supported?
- Are there security restrictions?
- Where do I have NFV-infrastructure capable of supporting this service?
- What is the congestion/performance of various data centers, in/egress facilities, etc.?

Constraints come from various sources; order data, inventory data, fault/performance topology data, and other sources. Typically, they are not deterministic: ("if this, put the VM there"), but rather a series of constraints that must be met at instantiation time, finding a solution that works in the context of available facilities, etc. This is where models, policies/rules, and context come together – allowing for self-realizing instantiation, with a high degree of flexibility, and rules/policies that ensure that conflicting constraints are met.

The key take away from this simple example is that "orchestration" is actually a collection of activities, at different layers, that combines data and requirements from several different sources and processes, to realize and ultimately manage both infrastructure and the products/services that depend on that infrastructure.

A MODERN RECIPE FOR ORCHESTRATION-DRIVEN OPERATIONS:

In the example above, we introduced many of the qualities of a service, with an eye toward modularity, layering, flexibility and re-use. In this section we look at modern practices to realize services. The key objectives are automation, re-use, flexibility, and self-realization – all of which reduce costs and facilitate agility in both innovation and operations.

Object-Based and Model- and Policy-Driven

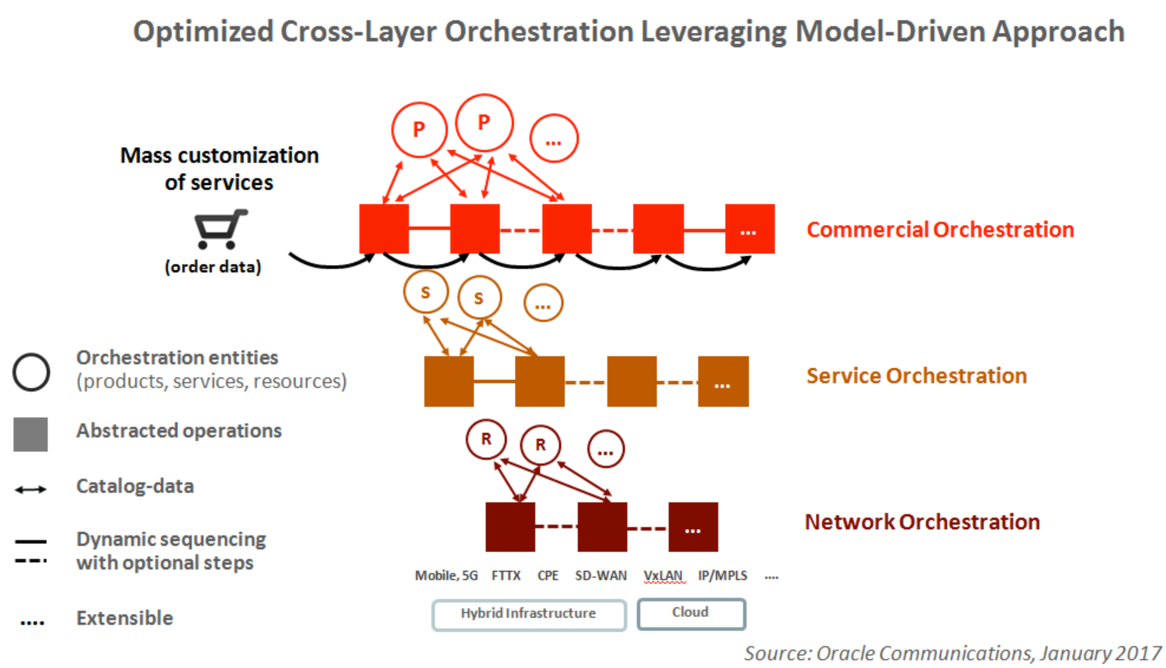
At the core of all those objectives are object-based design, together with model-driven orchestration and policy-driven orchestration. While policy and context is a complex topic[2], it is essential to understand that with proper orchestration design, network functions and/or services can be instantiated such that they meet a) the service requirement, b) the order specifics and c) the realities of available infrastructure capacity and conditions, and do so automatically. This same design flexibility, driven by models and policy/rules, also enables automation via self-healing and self-scaling to be nothing more than re-instantiations, but with changed context (e.g.: a failure). Because all of this flexibility comes from a small set of base models and orchestration methods, this also means fewer unique items to build, maintain, update, and integrate, and as a result less cost, less work, less time.

Support for Hybrid Networks

These methods need not be limited to new virtualized resources and services; they can also be applied to existing (physical) technology domains as well, yielding a broader base of agility. Our research has shown that networks will be hybrid for at least 15 years; consequently it's important that operations support hybrid environments and that we achieve the greatest agility possible for traditional technologies as well as new.

[2] For a more complete discussion of policy and its operation and benefits, please refer to "*Closed Loop Automation in Virtualized Networks*", and "*The Role of Policy in Virtualized Networks*", both available from www.appledoreresearch.com

Figure 4: Catalog, Layering, Objects, Model data and Policies Combine to Delivery Agility



What to Avoid

Before we discuss best practices, it is useful to consider some insidious practices and half-steps that *appear* to deliver progress, but in reality modernize the sclerosis and fail to deliver re-use and agility. For example:

- *Don't* build large workflows and simply attach them to product catalog entries
- *Don't* build traditional workflows, rather than spending the addition effort make them model driven
- *Don't* stop at model driven – those too will be static. Ensure that you have a policy/rules provision to make them contextual at run-time
- *Don't* mix resources into services, or services into products. Maintain layering, abstraction and separation of concerns (or there will be minimal re-use and higher maintenance)

Best Practices

The following table summarizes what we believe are the most critical "best practices" for orchestration in both hybrid and virtualized networks. We believe that these serve the causes of cost reduction, agility and risk reduction simultaneously.

Table #1: Best Practices for Orchestration in Virtualized and Hybrid Networks

Layered Orchestration
goals: support E2E operations, not merely network ops; separation of concerns
	• Orchestration must operate at multiple layers, from infrastructure up through the product/service layer inclusive of operational and billing processes.
	• These layers are also important for de-coupling and simplification of control loops – which are necessary for automation.
	• Important corollaries are:
	  o Separation of concerns between layers (as well as functions) to allow for clean integration and growth of libraries.
	  o Mapping to the hierarchical structure of resources, services and products in the catalog.
	  o When catalog and layering of objects align, it is possible to perform business/service/product innovation via the catalog, using abstracted data that can be understood and manipulated by business analysts, and even customers via self-service.

Componentized (Object-based) ***goals: high levels of re-use, reduced integration costs, reduced maintenance costs, rapid inovation and agility***	• Develop self-contained, loosely coupled components. • Adhere to layering and model/rules driven characteristics above. • Include operational models, data and logic (thresholds, test script IDs, customer activation logic, etc.) • Goals/benefits: re-use, cost reduction, rapid innovation, reduced innovation costs, reduced maintenance costs (maintain a single object), reduced integration costs, and consistent implemntations of any give function • Overall design should support "recursive assembly" – e.g.: the ability to combine simple objects into complex objects and then re-use *those*. This is a great example of where loose coupling and a single highly re-usable object is critical – so that all "upstream" resources, services and products "inherit" the characteristics (and updates) of the lower level object(s).
Models and Rules ***goals: self realizing, context-aware***	• Models that capture the characteristics of network services, resources and services, including components, topology, operational tasks/logic, affinity data, configuration parameters (and many more). • Policies that can modify these models (actions) at run-time based on current context (e.g.: orders, inventory, network condition). • *This means avoiding logic that depends on pre-knowledge of conditions.* • Result: one base model; many instances, all self-realizing based on context. Tremendous flexibility without the need to "foresee the future".

IN SUMMARY:

Virtualized networks, cloud technology and digital services promise huge benefits in terms of flexibility, cost and agility – and therefore new revenues. These benefits can be easily lost without concurrent transformation of operations. The industry is rightly concerned about the complexity and risks associated with the introduction of virtualized infrastructure, new operations methods, and its integration with today's OSS/BSS. It is also rightly concerned with the implication of doing too little and falling behind. The approach endorsed herein helps minimize both sets of risks.

Appledore Research Group www.appledoreresearch.com

This paper focuses on a vision of next-generation orchestration that simultaneously achieves order-of-magnitude improvements in agility and cost, while also reducing complexity and risk in the medium and long term. In effect, we engineer processes and costs for both economies of scale and scope. It also engineers them to reduce maintenance costs and integration costs – including integration with existing legacy – by creating an environment where fewer point-to-point integrations exist. It is fundamentally more efficient and scalable.

Achieving this nirvana, however, demands that the industry change its operating business assumptions, and therefore its long-term / near term cost trade-offs. The hyper-scale cloud competitors are already showing the way, with nearly full automation. They are both a model to emulate, and a potential competitor that CSPs must develop the agility and cost structure to co-exist with. This future demands that we invest in a properly architected set of tools for orchestration, model development, logic development, rule development, libraries, catalog integration and layered orchestration. This demands adherence to an architectural vision, and a willingness to invest now for much larger benefits later. The potential payback, to those that make this commitment, is very large –if we design appropriately.

ORACLE
COMMUNICATIONS

Appledore Research Group

BIBLIOGRAPHY:

All research is available individually or under subscription, from Appledore Research Group.

http://www.appledoreresearch.com

"*Managing The Telco Cloud*", ARG 2015.

"*The Role of Policy in Virtualized Networks*", ARG 2015.

"*The Economics of Virtualized Networks*", ARG 2016.

"*The Role of Closed-Loop Automation in Virtualized Networks*", ARG 2016.

Research Note: "*Closed Loop Automation and the New Role of Assurance*", ARG 2016 (free)