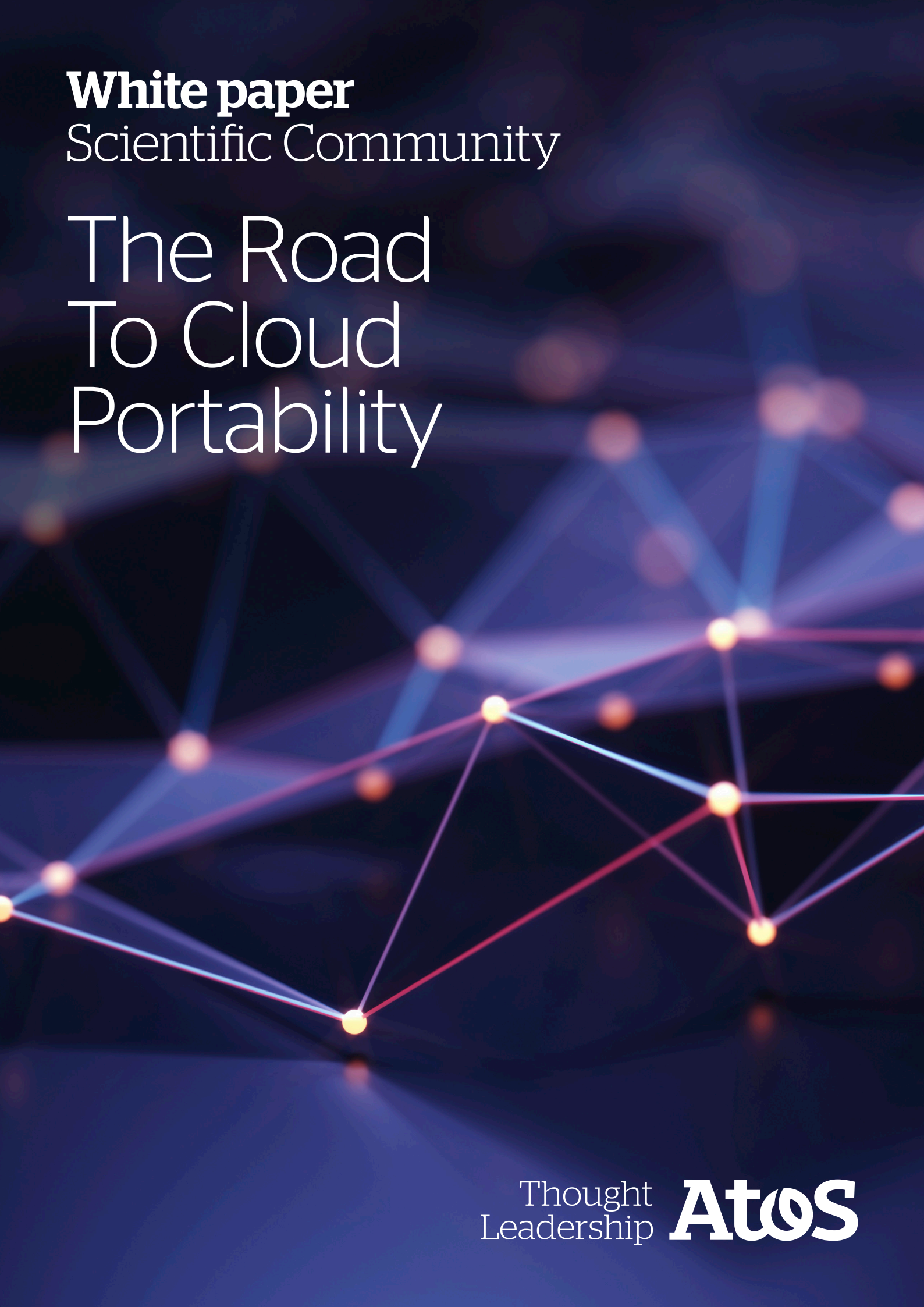
White paper
Scientific Community

The Road To Cloud Portability



Thought Leadership

Atos

About the Atos Scientific Community

Publically launched by Thierry Breton, Chairman and CEO of Atos, and sponsored by Hubert Tardieu, the Scientific Community has 135 members from all geographies where Atos operates, representing a rich mix of skills and backgrounds. Its aim is to help Atos anticipate and craft its vision of upcoming technology disruptions and the future business challenges that will be faced by the markets it serves. By making this vision available to its clients, and by investing in areas related to the findings, Atos intends to help its clients make informed decisions regarding the future of their Business Technology solutions.

Follow the Atos Scientific Community on atos.net/blog and atos.net/Scientific-Community

About the Authors

Purshottam Purswani Chief Architect, Infrastructure & Data Management in APAC

Pedro Baldanta Seijas Solution Manager, Infrastructure & Data Management in Iberia

Guy Lidbetter CTO, Global Infrastructure & Data Management

With the contribution of **Carlos Veira Lorenzo**

Introduction

The proliferation of digital technology has created an awe-inspiring number of connections, products, services, ideas, insights and efficiencies. Every business wants to be on a Digital Transformation journey to avoid being disrupted, displaced or just disappearing. Over the past decade, it has also become clear that speed has won in the marketplace: businesses that are able to innovate and experiment with new delivery models and platforms -usually with software-based solutions- are outcompeting those that follow more traditional delivery models.

The new digital era has witnessed a series of fundamental shifts in the way enterprise applications are designed and how they engage with their business systems. From monolithic mainframe applications to Client/Server and then internet based and mobile applications, the birth and maturation of the digital revolution have itself been driven by the radical adoption and transformation brought by the Cloud, bringing a huge diversity of applications in most enterprises.

Back office services, core business "money maker" services and new revenue sources are unevenly distributed among them. Different companies also show different distributions depending on sector, innovation profile, history and other factors.

However, the digitalization challenge is forcing businesses, to a greater or lesser extent, to stablish a "Transformation Continuum" as their daily business reality, both at scale and full speed.
This situation requires a complete rethinking of how certain IT challenges have been traditionally addressed:

- Multi-vendor policies have turned into multi-cloud policies; how should these be addressed?
- How should Exit Strategies for so many contracts be managed?
- How is vendor lock-in in the Cloud avoided?
- How are "mobile-first, cloud-first" requirements addressed in the absence of long and complex migration or transformation projects?

In this context, portability becomes a very interesting property that provides the right degree of flexibility and plasticity that this new business environment is demanding.

This paper explores how new developments in technology can deliver upon the portability promise in the context of Cloud environments. It defines what is meant by Portability and then analyzes the Enterprise Application Landscape to understand the technological challenges of Portability in Cloud environments and explore what it takes to address this problem.

Cloud Portability - Scope and Context

Cloud portability is defined as "The property of a given workload whereby it can be moved from one Service Provider to another without change."

Portability is not necessarily the same as "Live Migration"[1], "Cloud Migration" or "Workload Mobility". In this paper, a "workload" is considered to be an application or service being executed. This application might be stand-alone, distributed or implement a variety of architectures.

Therefore, the fact that a specific role or component of a given application could be "Live Migrated" doesn't define that whole workload as "portable". Conversely, workload mobility technologies are usually constrained to run on specific locations (usually a region or a data center) and they are mostly used to provide high-availability and resource load balancing. There is indeed the possibility of exercising long distance mobility; however, this scenario often raises specific requirements at the infrastructure level on both sides of the transaction which results in focus shifting to the infrastructure rather than the workload.

The same applies to "Cloud Bursting"[2]. Extending resources from a remote Cloud Service Provider does not in itself make a workload portable. This requires that the application components are ready to take advantage of the Cloud Bursting capacity.

This paper explores the challenge from the opposite perspective: the role of portability at the workload level and why this is relevant.

"Multi-Cloud Applications", whose roles and components are distributed among an arbitrary number of Service Providers-represent a special case which brings its own set of challenges in terms of portability. Whilst many of the topics, ideas, and concepts covered in this paper can be applied to multi cloud scenarios, their particularities, are not directly addressed by this research. Atos Research and Innovation has published some interesting works[3] on Multi-Cloud Applications which can be consulted on this specific domain.

1. Live Migration: https://en.wikipedia.org/wiki/Live_migration
2. What is Cloud Bursting: http://www.service-architecture.com/articles/cloud-computing/cloud_bursting.html
3. Ana Juan Ferrer et al. 2012. "OPTIMIS: A holistic approach to cloud service provisioning". Futur. Gener. Comput. Syst. 28, 1 (2012), 66–77. F. D'Andria, et al. "SeaClouds: a European project on seamless management of multi-cloud applications." Software Engineering Notes of the ACM Special Interest Group on Software Engineering (SIGSOFT SEN), 39(1):1-4, January 2014. Ana Juan Ferrer, David García Pérez, and Román Sosa González. 2016. "Multi-cloud Platform-as-a-service Model, Functionalities and Approaches". Procedia Comput. Sci. 97 (2016), 63–72.

The IT Management context

The enterprise application landscape is a rolling/evolving reality that is constantly changing. Each application landscape is composed of many moving parts: all of them evolving at the same time but at different speeds. But how the stack of evolving Life Cycles aligns and is managed needs to be addressed. Otherwise, this will not just put portability in jeopardy, but also the capacity to deliver true value to the business.

Life Cycle Management is a well-stablished management discipline. The question is, however, whether there are additional insights beyond just alignment and management relevant to Cloud Portability.

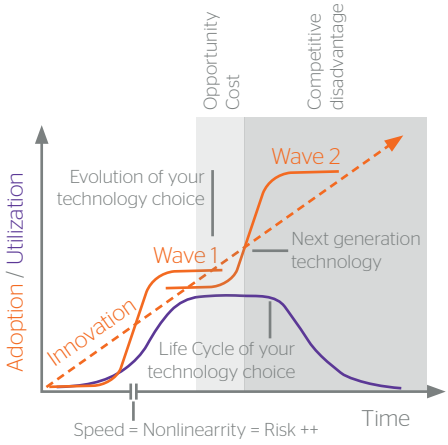


Figure 2: Wardley Evolution[4] curves for Lack of Market Alignment © Atos

The Alignment problem

In the most general sense, technologies may not be aligned/ connected with innovations within the Market. In such a case, impacts can range from moderate opportunity costs to a serious competitive disadvantage (see Figure 2 and Footnote) and business-side implications can be diverse, depending on parameters such as industry and affected activities.

Impact assessments must be run individually for each scenario to get to any further conclusions. However, given the transformational nature of the technologies discussed in this paper (like Cloud and Containerization), the assumption that the competitive impact can be significant is valid.

Likewise, when technologies are not aligned/ connected with tangible business outcomes (Figure 3), the consequences can also be similar. In this case, however, opportunity costs can be directly connected to profit and loss since resources have to be mobilized to make things happen.

Once again, the business impacts of unmet demands require tailored analysis and cannot be generalized. By looking at the nature of those demands the implications can be qualified relatively easily, such is in the following simple example:

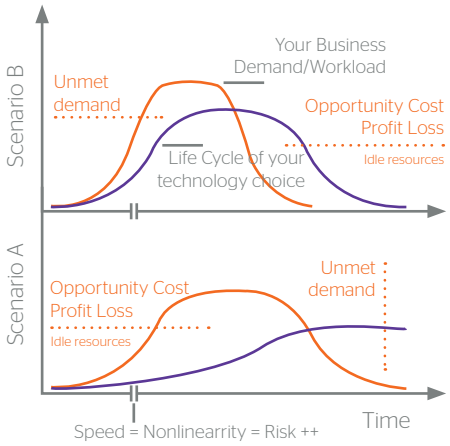


Figure 3: Lack of Business Alignment © Atos

Demand	Implication
Regulation	Competitive disadvantage (with those that do comply)
Keep up with competitors	Competitive disadvantage
Innovation	Opportunity loss

Table 1: Simple example of Impact Analysis

4. "On mapping and the evolution axis". Simon Wardley, 2014 (http://blog.gardeviance.org/2014/03/on-mapping-and-evolution-axis.html). "It has all gone a bit Wardley here". Simon Wardley, 2014 (http://blog.gardeviance.org/2014/05/it-all-gone-bit-wardley-here.html)

The IT Management context

Seasoned IT Managers are already aware of alignment issues because they are not new. They also know that this is a multi-level concern as illustrated in Figure 5. This impacts systems and ecosystems alike. The bigger the complexity, the greater the impact.

The transformational nature of modern technologies adds one more element to consider: speed, which has a significant influence over the main variable that drives a Life Cycle: time. As a result, IT Managers have to figure out strategies to deal with non-linear evolutions, uncertain or unknown risks, etc. at faster speeds. However, business acceleration has been present for some time and is hardly new. What these transformational technologies are now doing is increasing the pace of acceleration, in some cases by orders of magnitude.

To illustrate this point:

- Will ignoring containerization as an encapsulation strategy lead to a competitive disadvantage?
- Will a virtualization-centric strategy in the age of containerization represent an opportunity loss or remain at the core of unmet business demands?
- What will be the impact of accelerated speeds in a business context and how can the transformational effects of cloud and containerization technologies be calibrated in a particular situation?

The Service perspective

This vision of Life Cycles remains a traditional one: technology driven or asset driven, which will drive choices about technology. However, it doesn't provide significant insights to outline a portability strategy in a cloud-based environment. In other words, a different perspective is required.

Typically, Services are based upon one or more solutions and a set of practices, processes, and people. Likewise, solutions are made up of arbitrary combinations of products and other software components, such as the applications themselves. Once those applications are executed, they become workloads with an identity. All of these elements are connected to each other and all of them have their own life cycle. However, the activities and high-level IT concerns for each of them are not the same. Mapping them altogether provides a first step to getting a proper perspective: Table 2 describes different high-level activities and IT business concerns[5] mapped to different Life Cycle stages and scopes.

Life Cycle	Stage			
Scope	**Planning**	**Short Term (Operational)**	**Medium Term (Tactical)**	**Long Term (Strategic)**
Service	Definition	Delivery Configuration	Support Scalability	Performance (Financial & Operational) Innovation Outsourcing
Solution Product Component	Definition	Release Availability Capacity Security	Update Maintenance Scalability	Upgrade Consolidation Migration Integration Transformation Decommissioning
Workload	Proof of Concept	Operations Availability Capacity Security	Mobility Scalability	

Table 2: Mapping of business concerns to Life Cycle stages

5. As ITIL-inspired generalizations.

This is very generalized and requires fine-tuning for each specific case. However, it can help identify the right spots in which Portability, and other related business concerns, really belong, as illustrated in Table 3.

Portability is a strategic property closely related to value proposition, multi-sourcing and exit strategies[6] from a service perspective. Portability must also be planned and designed as part of the solution.

Finally, it is noticeable that Portability; solution resiliency; workload mobility and elasticity; and operational agility are also closely related.

This also means that, in practical terms, solutions, services or workloads for which Portability isn't particularly compelling are easy to identify: those with known, predictable and short life cycles and which implement relatively self-contained architectures. The remainder can make use of the previous map to lay out proper situational awareness and trigger the appropriate action plan.

In summary, now that Portability can be positioned in the context of related business concerns and their respective life cycle stages and scopes, the "what" to do; "when" to do it and, most importantly, "why" it should be done, are much easier to understand.

Life Cycle	Stage			
Scope	**Planning**	**Short Term (Operational)**	**Medium Term (Tactical)**	**Long Term (Strategic)**
Service	Multi-sourcing Business value Exit Strategy			Portability
Solution Product Component	Resiliency Agility Portability	Agility		Portability
Workload		Mobility	Elasticity Portability	

Table 3: Mapping of Portability to Life Cycle stages

6. This paper is not focused on the inherent risks of the cloud. However, to provide some context to the rationale behind Multi-sourcing and Exit strategies in the cloud, it may be useful to bring in a couple of examples that showcase the consequences of lock-in: "Firebase costs increased by 7000%!" (https://medium.com/@contact_16315/firebase-costs-increased-by-7-000-81dc0a27271d); "End of support for FormsCentral" (https://helpx.adobe.com/acrobat/kb/end-of-support-formscentral.html).

Challenges and Roadblocks

Portability of workloads in Cloud environments presents a number of challenges that must be identified and understood before considering such an undertaking.

The first and most obvious challenge touches the computing environment that defines the target environments: are there restrictions on the computing stack that prevent workloads from moving from source to destination?

Licensing and commercial strategies can also impose limitations to transfer licenses, solution support or introduce significant changes in the cost drivers used when a workload moves from one Service Provider to the next.

IT landscapes show a diverse distribution of different application and service architectures at any given point in time. This often raises the conundrum of finding "universal" solutions to complex problems when it comes to moving workloads around cloud players.

Similarly, it is key to maintain uniform and trustworthy security levels since any compromise in workload's security would jeopardize their portability properties.

Another set of challenges appears when data volumes and service dependencies get brought to the fore. It is not just about how hard it is to move one workload in the cloud, but also how the Service Levels committed will be affected when doing so.

Organizations will also be impacted by workload portability in different functions: Procurement, Legal and HR are the most obvious but SIAM-based relationships could also be affected.

So, there are many aspects to take into account[7], both technical and non-technical These all tend to be described and analyzed as discrete elements, but they are all interrelated, reinforcing each other and combining to deliver a higher order outcome: an application or a service. In other words, making the effort of maintaining a holistic view is key to get a proper perspective of the cloud portability challenge.

As a result, it would not be a surprise that managing the threats of potential Analysis Paralysis dysfunctions also becomes aconcern.

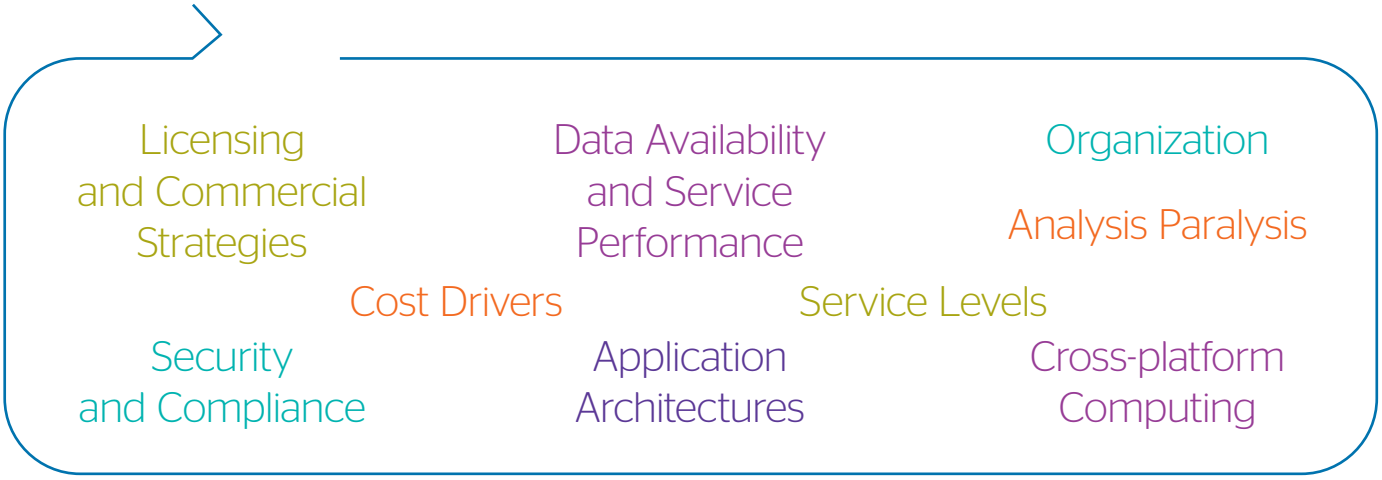


Figure 4: Summary of Challenges and Roadblocks[8] © Atos

7. A deeper description and analysis of roadblocks and challenges can be found on Annexes chapter.
8. Namely: stand-alone, Client/Server and Multi-tier applications of any kind.

Overcoming the Challenge

Most enterprise environments have the diversity of architectures in common that can be found in their respective portfolios of applications and services. This is the result of the natural evolution of technology, the application of different IT Management paradigms over time and a mirror of the different business challenges that each environment went through (mergers and acquisitions, expansions, recessions, etc.).

Addressing the portability challenge in Cloud environments must begin from a complete understanding of this structural diversity. Otherwise, it will be impossible to provide a sensible answer for each case and fulfill the expectations of the business.

Therefore, to facilitate this understanding, workloads can be grouped into the three defined categories:

- Legacy workloads. These follow traditional layered designs of enterprise architectures[8].
- Modern workloads. These follow so-called cloud-native architectures[9].
- Transient and Hybrid workloads. These are sometimes also referred to as Enterprise Cloud Applications and, because of their nature; they tend to exhibit encapsulated versions of the previous two.

The rationale behind this classification is to align analysis with the most common journey to the Cloud. Enterprise Applications must first make use of some kind of encapsulation technology to minimize architectural changes; and, over subsequent evolutionary steps, be transformed to leverage the full potential of a given Cloud environment.

This taxonomy also introduces the key mechanism that makes this transition possible: the "encapsulation" process. Up until recently, the encapsulation technology of choice has been server virtualization, which remains prevalent. However, the rise of containerization[10] is changing this landscape quite rapidly. Containerization technologies are at the core of Portability in Cloud environments because they represent an encapsulation layer that can be nested on top of traditional environments (physical or virtual) with negligible overhead on the target workload. The result is that those who leverage containerization regain control of their workloads and enjoy a cleaner distinction between "Dev" and "Ops"[11]. With containerization, enterprises will adopt fast-evolving Function as a Service (FaaS) platforms allowing business users to develop, run, and manage application functionalities without the complexity of building and maintaining the infrastructure.

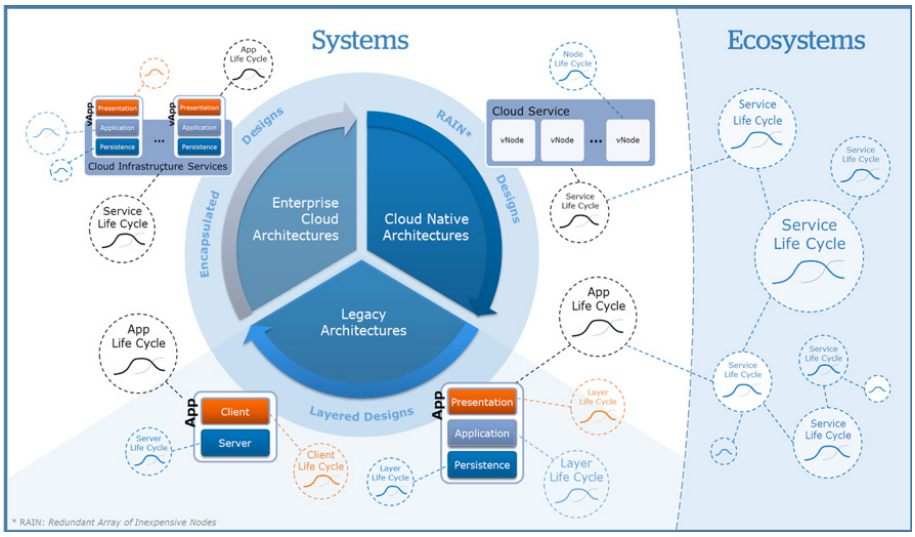


Figure 5: The mesh of the IT Landscape Life Cycle Management © Atos

8. Namely: stand-alone, Client/Server and Multi-tier applications of any kind.
9. RAIN (Redundant Array of Inexpensive Nodes) designs, "12 Factor" Apps, Microservices, Serverless, etc.
10. Lightweight virtualization technique (https://en.wikipedia.org/wiki/Operating-system-level_virtualization) around which a very dynamic ecosystem has been developed (refer to Figure 6 in section "Making choices" for a more comprehensive overview)
11. "Dev" and "Ops": common abbreviation referred to "Development" and "Operations".

Overcoming the Challenge

The "Architect's Approach"

The structural diversity of IT environments and the set of challenges and roadblocks already covered, suggest that universal and simple solutions to a complex problem like this are not possible. "Silver bullets" do not exist.

Portability is no different to any other IT endeavor since it is just a combination of Application, Technology, People and Management Processes. In other words: Portability is not a "product" or a "thing"; it is a "property" that can be achieved by following different paths.

That being the case, focus on shortcuts and recipes ("if I do/use/apply X, I will achieve portability"), will quickly expose their limitations even if appealing or pragmatic for a set of limited cases. This is why it becomes necessary to take a step back and embrace the "Architect's Approach": Portability must be designed, built and maintained rather than "purchased and consumed".

Things change over time and at different speeds which results in a different enterprise application landscape in any given context.

The answer to Cloud Portability must therefore be a combination of the core variables (Application, Technology, People and Processes)[12] allied with "Time". The question is then, how to resist "Time" based demands effectively in a field where expedience has so often been the winner.

The instruments that make this possible are Design Principles, which, when defined properly, enable a privileged position that rides waves of technological evolution, adapts to paradigm changes and provides the necessary guidance for building solutions to real business problems.

The Design Principles that facilitate the Portability in Cloud environments are:

1. Dependency Encapsulation
2. "Bring Your Own Trust"
3. Data flows and Locality matter
4. Minimize friction
5. Remove prescriptiveness
6. Focus on Open Standards and, where appropriate, Open Source

Design Principles alone, notwithstanding, are not enough and the scope of the portability ambition must also be defined.

Is full portability required or just the ability to move between Public Cloud and Private Cloud? Are there other combinations? Is it the same for each and every workload? Will different approaches apply to different groups or categories of the application map?

The answer to these questions will determine the set of tradeoffs to be faced possible limitations in the outcome and how strictly the Design Principles will be applied to a particular workload.

1. Dependency Encapsulation

The dominant encapsulation technology has traditionally been Operating System virtualization. Whilst there are alternative application level technologies that can serve this purpose, it is the Containerization Ecosystem[13] that provides better agnostic properties, thus satisfying Dependency Encapsulation more efficiently. Nevertheless, containers are not enough and other strategies must be considered to implement this principle (see next page).

Depending on the scope of portability ambition, different tradeoffs can be applied and decisions taken about an encapsulation strategy.

12. This paper is focused primarily on the first two (Application and Technology) since, from the perspective of Portability, they enable the remaining ones.
13. See Figure 6 in section "Making choices".

Strategy	Description	Examples
Replacing key infrastructure services with SaaS alternatives	Workloads will still consume and interface with SaaS alternatives without the burden of maintaining a strong local dependency. Interesting targets for replacement could be monitoring, telemetry, alerting, back-ups, provisioning (build, test & deploy), security (Identity Management, integrity and access control) and even other support services such as billing, ticketing or CRM systems[14].	PagerDuty, NewRelic, Loggly, CrashPlan, ServiceNow, ...
Orchestration Tiering	The most common form of lock-in is API hooking/coupling. It is usual to adopt a Cloud Service Provider's instrumentation services. However, a portable scenario requires its own independent Command and Control plane. This is the only way services can be fully decoupled from the underlying infrastructure and, as a result, be free to move from one Cloud Service Provider to the next.	Mesosphere, Kubernetes, Docker Swarm, Cloud Foundry, ...
Agnostic PaaS offerings	Application services adopting vendor-specific PaaS platforms can only be considered portable across PaaS instances of the same flavor. However, to consider them portable components, competitive marketplace of equivalent alternatives is needed. This also usually means the existence of on-premise deployment options that offer some potential for customized encapsulation. In any case, it is obvious that the fact that these are already Cloud Services means that they are already "properly encapsulated dependencies".	Cloud Foundry, Azure/Azure Stack, Docker, ...
SaaS offerings providing functionally equivalent alternatives	Workloads or services that have adopted vendor-specific SaaS cannot be considered portable by definition[15]. Their only chance is to have access to a competitive marketplace of equivalent alternatives. This also usually means the existence of on-premise deployment options that offer some potential for customized encapsulation. In any case, it is obvious that the fact that these are already Cloud Services means that they are already "properly encapsulated dependencies".	GitLab, Jira, WordPress, Mattermost, Office 365, Elasticsearch, ...
"Bring your Own Infrastructure or Application Roles"	Mainstream Cloud Service Providers provide both value-added services and higher order components that are pretty much unique to them. These can range from API-driven load balancers and scale-out data stores to full Big Data platforms. They are all great choices but, from a portability perspective, they all lock you in. To become portable, those roles need to be encapsulated and delivered as a set of virtual appliances.	HAProxy, VyOS, SoftEther, F5 Virtual Edition, Hadoop, Kafka, ...

Table 4: Encapsulation strategies

14. When considering substitution of services, Martin Fowler's perspective, "Utility vs. Strategic" (https://martinfowler.com/bliki/UtilityVsStrategicDichotomy.html) also becomes a key decision driver.
15. "Cloud Migrations" are explicitly excluded from the definition of Portability. This way, moving from Salesforce to Dynamics 365 would be a migration, but not a portability exercise.

Overcoming the Challenge

2. "Bring Your Own Trust"

Trusted Computing[16] is a wide and complex domain, heavily connected with the business context in terms of constraints and use cases. For certain scenarios, portability won't be the driving business concern. That is why it is very important to frame the discussion about Trust to those workloads or services that have realistic portability ambitions.

Portable workloads should move among Cloud Service Providers regardless of the supporting infrastructure. However, not every Cloud Service Provider delivers the same security levels and a given workload's security requirements may not be compatible with a particular Cloud Service Provider.

By implementing a separate security layer, potential Trust-related incompatibilities can be resolved. This usually requires a means for encrypting both storage and communications. Depending on the application context, requirements for Key Management and Key Distribution must also be analyzed in detail[17].

Does this mean that workloads not implementing this principle cannot be considered portable? Not completely but it does limit portability to those Cloud Service Providers with similar trust levels.

3. Data flows matter, Locality matters

Resource Allocation, Resource Affinity, and Locality-Awareness are hard structural problems when nested layers of virtualization come into play. With full abstraction/isolation between layers in place, resource schedulers at each level can't coordinate decisions with their peers. As a result, they can't take into account of how persistence data flows are managed for a specific application or service.

The consequence is normally paid for in terms of Availability (inability to manage failure domains consistently), Performance (unpredictable contention/latency issues), Service Delivery (timeouts and occasional functional errors) and Service Levels (inability to provide a predictable user experience).

This leaves full responsibility to the scheduler higher up the stack to make assumptions about the underlying infrastructure to provide allocation and affinity rules. In either case, this is why Advanced Resource Schedulers and Cluster Managers like Kubernetes, Mesos or Docker Swarm are gaining so much attention.

Nevertheless, Advanced Resource Schedulers are not always necessary. Different applications and architectures may need specific approaches which means, for example, that static allocation can be a perfectly valid solution in certain cases. The limitations of Cluster Managers need to be understood to see how well they fit the architectures and applications they must deal with and the key is to design solutions that take these factors into account.

4. Minimize friction

This principle is quite broad since sources of friction that prevent portability can be everywhere but this doesn't mean that strategies to minimize friction cannot be applied. Abstraction[18], Indirection[19], and Adaptation[20] are just a few that can be borrowed from Software Engineering disciplines. One of the most effective ways of reducing friction is applying Dependency Injection[21] tools and techniques to achieve dynamic and declarative configurations in distributed systems.

5. Remove prescriptiveness

Portability and Plasticity are closely related. Workloads aiming to be portable must exercise plastic designs and, in general, prescriptiveness is a significant source of rigidity.

For example, an application relies on a platform that mandates a particular way for logging events. Applications that don't fit naturally into that facility will inevitably encounter problems[22]. Likewise, unless the platform can be encapsulated, or, the tools relied on are also available on all the target Cloud Service Provider platforms, similar problems will occur.

6. Focus on Open Source & Open Standards

Portability and Interoperability have much in common and it is hard to conceive of one without the other. Conversely, closed and proprietary solutions impose their own set of roadblocks to portability. When considering these aspects, Open Standards especially and Open Source will usually be more friendly choices with regards to portability ambitions.

However, Open Standards are only as relevant as the level of adoption they achieve. De-facto standards may not be open but can be well established and provide reasonable interoperability. Likewise, COTS solutions may not be open but can be quite friendly to be consumed on a wide variety of platforms with support from the provider.

In summary, even though the preference may be for both Open Standards and Open Source, an open-minded approach is still required to what is the best solution.

16. Trusted Systems: https://en.wikipedia.org/wiki/Trusted_system
17. For instance, Weave Net secures communication among containers using encryption based on a shared secret and a proprietary protocol (https://www.weave.works/docs/net/latest/using-weave/security-untrusted-networks/ ; https://www.weave.works/documentation/net-latest-how-it-works/net-latest-encryption/)
18. Abstraction: https://en.wikipedia.org/wiki/Abstraction_(software_engineering)
19. Indirection patterns: https://en.wikipedia.org/wiki/Indirection
20. Adapter pattern: https://en.wikipedia.org/wiki/Adapter_pattern
21. Dependency Injection pattern: https://en.wikipedia.org/wiki/Dependency_injection
22. That could be the case, for instance, of workloads that run inside a Docker container but don't log events to the standard output.

Making choices

Armed with solid Design Principles and a proper knowledge of how life cycles look like in terms of alignment and service orientation, the challenge of implementing portable workloads in the right way can be addressed.

To cover this last mile, a comprehensive understanding of the technology landscape is required. This will inform the potential tools and solutions that can be considered to achieve the goal. The more complete this understanding is, the better the quality of decision that can be made. In this whitepaper, a case study is provided based on Containerization technology. The defining characteristics of early stages of evolution can also be easily recognized: lack of consolidation (large number of players), volatility (in newcomers and leavers) and lack of maturity (early innovations hitting the market as Minimum Viable Products... or worse). Does this mean that these technologies are not valid and should not be considered? No, this is just one aspect of the decision-making process. In fact, at this point, life cycle alignment now becomes really valuable. This map provides a perspective of the different building blocks necessary to put together solutions that will eventually provide workload portability. However, a more strategic perspective will most likely be needed. Taking this into account, the potential solutions can be organized using the criteria outlined in Table 5.

Solution	Definition	Business Rationale
Structured	Based on well-established market products or services (SaaS) that fulfill a higher order function that provides some kind of vertical or horizontal integration (the "Cathedral" paradigm).	• The function/service delivered is not a source of competitive advantage. • Consume innovation "as-a-Product" or "as-a-Service". • Leverage an established ecosystem and market position. • Leverage an established competitive market on that specific segment. • Proven and cohesive experience (risk management). • Long-term engagement. • Time to market.
Unstructured	Custom built implementations that may assemble lower level third party components (the "Bazaar" paradigm).	• The solutions market is not mature enough (uncertainty). • Very specific need not covered by a competitive market. • The function/service delivered is a source of competitive advantage. • Freedom to innovate (Flexibility and Control).
Semi-structured or Hybrid	Arbitrary combinations of the previous ones.	• The solutions market is still in a transient stage (lack of maturity). • The business is in a transient stage.

Table 5: Categorization of the Containerization Ecosystem

Overcoming the Challenge

Structured solutions tend to hide the underlying complexities that businesses don't consider at the expense of some kind of solution lock-in. This becomes particularity relevant if an Exit Strategy or Life Cycle Management becomes necessary. Structured Solutions may also not be ready to support the full range of Application Architectures in scope (Traditional/Legacy, Enterprise Cloud, and Cloud Native).

Even for the application types Structured Solutions excel at, it is quite likely that they don't meet all the conditions to manage portable workloads. For example, portable workloads, as defined in this paper, must implement their own security layer to "Bring their Own Trust" to be portable across Cloud Service Providers regardless of the underlying supporting infrastructure.
What if the chosen solution lacks support for encrypted communications or the encryption provided is not "good enough"?
Workload portability could then not be delivered among Cloud Service Providers offering incompatible Trust levels.

With all these nuances in mind, the strategic vision of the available solutions within the Containerization Ecosystem can be mapped to provide a visual understanding of how the technology strategy connects with the current marketplace offering.

Lastly, it is worth revisiting the potential challenges and roadblocks introduced earlier in this paper as they will also play a significant role in the product selection and solution design processes.

The concept of nested virtualization is the element that gives back control of assets previously controlled by Cloud Service Providers. However, this capacity also brings the responsibility of doing proper resource scheduling at that level. In "Data flows matter, Locality matters", the relevance of concepts like Resource Allocation, Resource Affinity, and Locality-Awareness were discussed, not only for each individual workload, but also for competing ones when hosted on multiservice or multitenant environments. This should now be considered in the context of a product selection and solution design process and two key points should be highlighted:

1. Some workloads may not have a good fit with a (single) resource scheduling approach.

2. Resource Scheduling is a function, but not necessarily a (single) product.

To recap, "one size may not fit all" since the architecture of distributed systems plays a fundamental role beyond the mere selection of specific products. A vision of the whole ecosystem is also required to understand the all the potential solutions: both from a building blocks and strategic perspective. As a result, the definition and implementation of successful solutions that match the portability ambition and fit the workload environment becomes easier, regardless of how structured they are.

Conclusion

The digitalization challenge is forcing companies and governments to acknowledge a "Transformation Continuum" has become a reality, both at scale and at full speed. This requires a complete rethinking of how certain IT challenges have been traditionally addressed: multi-cloud policies and exit strategies at scale, extreme time-to-market requirements, etc. In this situation, workload portability becomes a compelling property that activates the flexibility and plasticity demanded and provides unique protection against the risks and threats of moving to a cloud platform.

However, Cloud Portability is not an absolute goal that must be pursued "at all costs". It must be properly framed in the context of the business services provided and the lifecycles of the stack of components used to run them.

Cloud Portability is not a "product" or a "thing"; it is a "property" that must be designed, built and maintained rather than "purchased" or "consumed" as a "silver bullet". In other words:

1. Workloads must be transformed to incorporate this capability.
2. Service roadmaps must be updated to ensure this property is exploited.

Fortunately, many of the concepts considered in this paper are not unique to Cloud Portability. It is quite likely that a starting from scratch approach will be needed as workloads will be implementing many of these features already. It is also likely that Cloud Native Applications, by their nature[23], become easier to transform than traditional ones. The natural evolution of technology will improve the maturity, productivity and popularity of the engineering practices and technologies. Therefore, it is relatively safe to anticipate that the Road to Cloud Portability will become easier as Cloud-Native workloads become a more prevalent form of software architecture.

Cloud Portability spans beyond just a single workload and requires a proper understanding of the structural diversity of the Enterprise Application landscape. This knowledge will facilitate:

- The definition of scope and ambition for Cloud portability.
- The identification, ranking and prioritization of candidates among Enterprise workloads.
- The outlining of a Strategic Portability Roadmap.

IT is a combination of Application, Technology, People and Management Processes. This paper has focused primarily on the first two given, from the perspective of Cloud Portability, they enable the remaining ones (People and Processes). However, the true value of Cloud Portability comes when this capability is activated and turned into action.

This suggests that companies committing to Cloud Portability as a genuinely shared goal, breaking down internal barriers will embed it as cross-functional collaboration initiatives. This is where further developments in the People and Processes space will come to the fore.

Cloud Portability and the concepts discussed throughout this paper will play a fundamental role, not just today, but also in the coming years. Yes, they will require effort, discipline, talent and expertise. They will also require organizations that embrace this challenge to sustain top-level commitment to define and execute strategic plans to realize the substantial business rewards on offer. Many companies will find the road to Cloud Portability a key enabler for reaching even greater heights.

23. CNAs are, by definition, already in the Cloud. This also means that they "tend" to adhere to modern software engineering technologies and practices. However, we must remember that CNAs also face their own challenges to Cloud Portability as it has been described in this paper.

Appendix

This White Paper has considered how a cloud portability strategy could be addressed within the complex multi-platform / multi-workload environments that exist in most large organizations today.

The following Annexes provide more detail on the following key areas that will require attention.

- Cross platform computing
- Licensing and Commercial Strategies
- Cost Drivers
- Application Architectures
- Data Protection, Security & Compliance
- Service Levels
- Organization

Cross-platform Computing

After decades of evolution, Computing Technologies are legion. In any portability evaluation, mapping of all the available technologies between the source and all potential destinations is essential. Table 6, provides the key aspects to be considered:

Technology	Source	Destination	Remarks
CPU Architecture	• Intel: x86/x64, IA-64 (Itanium) • Others: SPARC, POWER, MIPS • Legacy: PA-RISC, Alpha • GPUs: OpenCL, CUDA • Emerging: ARM, FPGA (OpenRISC)	• Intel: x86/x64 • ARM • GPUs	• Niche Cloud players also support other processors (especially those targeting Computing Services on the Edge). • The supported instruction set level matters (ie. AES-NI, SGX, compile-time optimizations, etc.)
Operating System	• Windows • Linux & BSD • Unix (Solaris, AIX, HP-UX, True64, ...) • Mainframe (zOS, MVS, GECOS, OpenVMS) • Others (MacOS X)	• Windows • Linux & BSD	• Virtualization technologies used in IaaS can support more OSs (especially on Private environments).
Run-times	• Any	• Java/JVM • NET/CRL • Others (PHP, Ruby, Python, JavaScript, Golang, ...)	• Key related trend: source-to-source compilation (transpilers)[24].
Libraries	• Any	• Any	• Static vs. dynamic compilation. • Binary compatibility[25] is key.
Distributed Computing Protocols	• RPC-based (DCOM/COM+, CORBA, RMI, ...) • Web-based (SOAP, REST, WebSockets, ...) • HPC (MPI, RDMA, ...) • Others (Protocol Buffers, D-Bus, ...)	• Web-based (SOAP, REST, WebSockets, ...)	• Binary compatibility is key. • Protocol version compatibility is key.

Table 6: Source and Destination mapping for Cloud Portability[26]

24. "Transparent Compilation". Martin Fowler, 2013 (https://www.martinfowler.com/bliki/TransparentCompilation.html)
25. Binary Compatibility: https://en.wikipedia.org/wiki/Binary_code_compatibility
26. Remark: both virtualization and containerization technologies are omitted from this table to focus on the basic foundations of computing workloads and facilitate the comparison of computing stacks. Other higher order architectural components of enterprise applications and services (such as databases, messaging systems, etc.) have also been excluded for the same reason. In no case, this decision suggests that any of them may be ignored at all.

This is a classic "find the lowest common denominator" exercise. However, the outcome is not necessarily black or white and this is a great opportunity to identify the different types of applications with respect to portability:

1. Workloads with straightforward potential to become portable.

2. Workloads that might require migration/transformation to increase their portability. Potential transformation projects should be ranked in different categories depending on difficulty/effort, budget, risk and/or other criteria to facilitate prioritization and decision making.

3. Workloads with no chance of becoming portable.

Because Cloud is becoming a computing commodity, midrange systems and applications are ideal candidates to become portable workloads. However, even type 3 workloads could find their way into the Cloud[27]. As described later, this could facilitate the portability of other workloads that may have them as a hard dependency.

Licensing and Commercial Strategies

Enterprise Software is always subject to some kind of licensing, subscription or contract. Similarly, Cloud Service Providers provide their own set of Terms and Conditions defining the boundaries and restrictions applicable in each case which can impact on portability.

These include:

- Inability to transfer licenses. This limitation could be applicable to all or some Cloud Service Providers due to the market strategy of the Software Vendor. However, it could also be triggered because of commercial bindings to the physical magnitude(s) of the platform (number of CPU, cores, nodes, cluster size/type, etc.)
- Lack of specific license tiers. Certain vendors won't offer full-featured licenses through Cloud Service Providers. Others provide them only through their own Cloud Services offerings.
- Lack of vendor support and/or certification. Certain vendors impose a highly restrictive compatibility matrix (hypervisor, OS, networking, etc.). Some even may exclude Cloud or virtualized deployments from their certification schemes.
- Too rigid terms for elastic environments. Some Cloud Service Providers provide very granular elasticity services to the point of offering per minute billing. However, Enterprise Licenses may not contemplate this possibility.
- Uneven minimum commitments. Sometimes an outsourcer or Cloud Service Provider may imposes limitations in the form of minimum commitments and rules for scaling up, down, in or out.
- Special conditions. Both Cloud Service Providers and Software Vendors may offer/apply non-standard conditions as benefits, incentives, discounts or rewards for a variety of reasons. When portable workloads are affected by these, proper impact assessments must be conducted in order to prepare them for eventual context changes.

Cloud Service Providers and Software Vendors are both trying to get as much customer loyalty as they can. Of course, what vendors call "loyalty", customers will often think "lock in" and often for good reason. They may even see both cloud and portability as threats to their market position and business model. In such cases, Licensing, Terms and Conditions and Commercial strategies will be used. This means that the situation will never be static and must be regularly reviewed whenever portability is targeted for workloads.

Cost Drivers

Cost drivers[28] are one of the most powerful instruments that both Cloud Service Providers and Software Vendors can use to become relevant in the marketplace. The definition of Cost Drivers is also part of a Commercial Strategy and, as a result, it is also subject to changes and evolutions. This means that they must also be regularly reviewed.

Cost drivers can also be combined, bundled or be connected to one another in complex and unexpected ways. That is why individualized and multifaceted analyses for each of the market players involved is necessary.

Workloads could be technically portable, but portability may still be restricted due to the cost implications. The opposite could also be true where moving could turn out to be either a strategic or a tactical opportunity.

27. Moving, for instance, Mainframe workloads to the Cloud with LzLabs (https://www.lzlabs.com/products).
28. Cost Drivers determine the complete cost of an activity, reflecting any linkages or interrelationships that affect it, directly or indirectly https://en.wikipedia.org/wiki/Cost_driver

Appendix

Application Architectures

The three enterprise workload groups, Legacy, Modern and Transient/Hybrid encompass all application architectures: from Standalone and Client/Server to Event-based or Microservices. Each has their own set of constraints, dependencies, assumptions, and topologies. Whilst many of them might be known or explicit, others may not.
As an added complication, there could also be implementation mistakes, design flaws or any arbitrary combination of them.

All of these elements risk being technology challenges to portability that can change over time as the applications or services evolve. The most common sources of external dependencies that might prevent portability will include:

- Cloud Service Portfolios: major or subtle differences between seemingly equivalent cloud services; services that are unique to a given provider
- Market Products (COTS[29]): might be provided with different versions, patch levels, third party components, etc.
- Physical devices: HSMs[30], license key locks, random number generators or any other function-specific appliance that happens to be in use.

Some Cloud Service Providers may also not be ready to support a clustering architecture, vertical scaling needs, network topology or have no presence in specific regions for the set of services being demanded of them.

The information on Application Architecture can be huge and trigger connections with other business concerns. But this is essential in deciding whether a workload is a candidate for portability and the transformation that would be required and hence if it is even worthwhile.

Data Protection, Security & Compliance

One of the most problematic and immediate technical roadblocks with portability are backup and restore services. Backups and data retention policies tend to generate a huge amount of "Data Gravity". System and Data Recovery are also subject to strict SLA scrutiny. The key questions are, therefore:

- Is it possible to design a backup strategy for portable workloads that meet the business RPO and RTO[31] objectives?
- Could a data recovery strategy[32] be implemented that it is compatible without compromising SLAs?

Security is fundamental to any application architecture, but more so in a public Cloud environment. Portable workloads introduce the challenge of maintaining uniform and trustworthy security levels whenever a workload is moved from one Cloud Service Provider to another. Any compromise in the security architecture would jeopardize the portability.

Challenges may be found at any point in the Security Management Framework. This means that the workload itself (including Service/Application Owners and federated third parties[33]) and the target Cloud Service Providers must both be considered. Policies, Processes, Tools and Techniques need to be checked and, of course, the People implications.

Other aspects also need attention.

1. Legal environment. Data Sovereignty with associated industry and government regulations[34] on where data is located and processed, and Intellectual Property threats.

2. Certifications and Control Frameworks to adhere to - SOX, Dodd-Frank, Basilea, PCI-DSS, HIPAA, FISMA, ISO 27001, SSAE 16, ISAE 3402, etc.

3. Disaster Recovery and Business Continuity. Regardless of how portable the workload is, DR plans are still required and regular validation that they are functional and fit for purpose.

4. Information or subsystems protected with DRM[35], IRM[36] or PKI[37] technologies. These shielding techniques tend to be quite sticky, particularly when they are tied to specific vendors or Cloud Service Providers.

These factors will translate to a set of technical requirements. Many are not new but some will be quite specific to Cloud-based deployments. For example, with APIs and Administrative "Control Planes[38]", some Cloud Service Providers may not provide the means to:

- Segregate roles and responsibilities with the granularity required.
- Trace, debug or audit API calls as needed.
- Adopt a Federated Identity Management system.

29. Commercial Off-The Shelf
30. Hardware Security Module: https://en.wikipedia.org/wiki/Hardware_security_module
31. Recovery Point Objective, Recovery Time Objective
32. Different data sets may have different requirements and SLAs. i.e.: hot data, warm data, cold data, ...
33 Refer to Figure 6 in section "Making choices" to explore federation protocols and technologies which are key for Identity and Access Management (IAM) solutions: OAuth, SAML, etc.
34. Particularly, both EU members and companies willing to do business with the EU, will increase their attention to the GDPR: https://en.wikipedia.org/wiki/General_Data_Protection_Regulation
35. Digital Rights Management
36. Information Rights Management
37. Public Key Infrastructure
38. This concept is used here in its extended meaning that involves any kind of Command and Control facility. This term was born originally in the realm of Software Defined Networking: https://en.wikipedia.org/wiki/Control_plane

Data Availability and Service Performance

A major roadblock to portability is "Data Gravity". The bigger the volumes of data, the greater the gravity (workloads "gravitate" around that data) and therefore the harder it is to move those workloads from one Service Provider to the next.

Since Von Newman defined the core computing paradigm that rules most of today's computing architectures, data locality - having the data as close to the processing units as possible - has been an implicit requirement for most of them. Breaking this rule usually means running into performance/latency issues and functional failures. When the data gets accumulated in large volumes this requirement becomes ever more important.

Nevertheless, from a portability perspective data gravity itself is not necessarily a problem: Moving it could be "just" a matter of time, money and availability windows to execute the transfer[39]. In other words, it may not be desirable to move a very large portable workload "too often" even though it is possible.

There is a further issue that other workloads, which "gravitate" around it, introduce external dependencies. Those systems could also have second-order dependencies with further workloads and so on. The problem is the dependency graph because it is quite likely to find non-portable workloads among them and it is not possible to consider workload portability in a vacuum that ignores all the interconnected items that surround it

This means that "data availability" is not just about having access to information (the "What"), but also having it at the right time (the "When"), and in the right way (the "How") which may mean for speed, quality of service and reliability. Because no 2 workloads are the same, not all will have the same requirements for data availability but knowing this information is crucial in Cloud environments because, as explained later, Nested Resource Schedulers may put data availability in danger.

Enterprise workloads are typically managed and any portable workload must be properly instrumented to provide the metrics and KPIs to the control plane. But Instrumentation and Control Planes may mean different things when addressing Legacy, Modern or Transient/Hybrid workloads. They could also vary across different Cloud Service Providers and therefore represent a potential new roadblock workload portability.

All this information enriches dependency graphs with details about hard, medium or weak data-related dependencies and the graph must be updated regularly since data and data flows are the most liquid asset in the technology landscape.

Service Levels

Moving workloads around between Cloud Service Providers triggers a significant amount of uncertainty. This is especially true when a workload hasn't run in the infrastructure that a specific provider has. So, even if a workload is portable "in theory" there may still be resistance to moving it. How a potential relocation would impact end-user experience, performance and, ultimately, Service Level Agreements needs to be established.

The Service Owner of any workload will always have responsibility for delivering the required Service Levels to users and business stakeholders. Therefore, when considering portability in cloud environments Capacity / Demand Management and all of the Service Support processes (Incidents, Problems, Changes, Releases etc.) must work seamlessly regardless of the hosting facility being used.

The underlying tooling to support these processes may be different for Legacy, Modern or Transient/Hybrid workloads but should be standardized as part of the transformation for any workload considered to be portable so that its management does not change, regardless of its location.

One of the most important aspects to consider is the SLAs offered by each Cloud Service Provider and how well they align with those committed to. A Cloud Service Provider might commit on just Virtual Machine availability while transaction response times are the SLA measure; different tools and metrics may be used to evaluate certain KPIs and trigger penalties; Cloud Service Providers change Terms & Conditions over time; The list can go on. Portability can be considered a technical challenge but many for factors need to be addressed.

Organization

Some organizations manage their workloads and service under the principle of segregation of roles and responsibilities. Many outsource Application Management to one service provider and the Infrastructure Management to another. It is not uncommon to see different outsourcing decisions for different Business Units or geographies. A company might also simultaneously apply multiple outsourcing criteria for its services (functional, organizational, regional, etc.)[40].

This organizational reality will have direct implications when pursuing the portability dream in cloud environments. The more stakeholders and contracts being managed, the more coordination, processes, and formalities needed.

Portability can also stress the organization itself to diverse degrees: Legal must deal with new and changing terms and conditions, back to back agreements, etc. Human Resources must find new kinds of recruits and address talent scarcity; Procurement must address new costing models, payment methods, and approval processes.

These transitions can be planned ahead of time, but organizational resistance to change will still have to be addressed.

39. In fact, network optimization techniques could also be applied to minimize latencies and avoid or minimize these transfers.
40. "Service Integration and Management: Motivation, Challenges and Best Practices". Atos Scientific Community, 2015.

About Atos

Atos is a global leader in digital transformation with approximately 100,000 employees in 73 countries and annual revenue of around € 12 billion. The European number one in Big Data, Cybersecurity, High Performance Computing and Digital Workplace, The Group provides Cloud services, Infrastructure & Data Management, Business & Platform solutions, as well as transactional services through Worldline, the European leader in the payment industry. With its cutting-edge technologies, digital expertise and industry knowledge, Atos supports the digital transformation of its clients across various business sectors: Defense, Financial Services, Health, Manufacturing, Media, Energy & Utilities, Public sector, Retail, Telecommunications and Transportation. The Group is the Worldwide Information Technology Partner for the Olympic & Paralympic Games and operates under the brands Atos, Atos Consulting, Atos Worldgrid, Bull, Canopy, Unify and Worldline. Atos SE (Societas Europaea) is listed on the CAC40 Paris stock index.

Find out more about us
atos.net

Let's start a discussion together

For more information: **atos.net/contact-us**

CT_J644_180209_RY_WP_THEROADTO