SMART CLOUD MIGRATION
A PROVEN, BUSINESS-FOCUSED APPROACH
“WE WOULD LIKE TO DISCUSS OUR CLOUD MIGRATION.”

It is a frequent topic of discussion these days. Most enterprises are contemplating or already migrating to the cloud. Companies deal with a sense of overwhelming complexity and risk. They worry about how to generate value and not just “migrate for the sake of migrating.” They wonder where to start and how to proceed.

This paper lays out a smart cloud-migration strategy. The first step is to establish clear business goals. The second step is to group applications into “clusters,” which leads to a 6-step sequential migration plan, treating different clusters in different ways. To execute, start migrating pre-production applications, and then the smallest and least sophisticated clusters. This is the “canary-based” approach to migration, which first migrates less complex apps and software and tests for any negative effects. (See Exhibit 1: “Smart Migration Strategy,” which outlines the processes described in this paper).

A cloud migration should directly embrace IT-process automation for repeatability and re-use, and prioritize pre-production environments. It is essential to keep an open mind about cloud innovation; just looking for cost reduction is short-sighted. Balance the need for cost-efficiency against the upside of disruptive cloud capabilities, such as machine learning or natural-language processing.

**EXHIBIT 1: SMART MIGRATION STRATEGY**

**BASELINE & GOALS**
- Cost vs. innovation vs. time to market priorities
- Scope
- Define KPIs and governance

**DEFINE APPLICATION CLUSTERS**
- High level baseline
- Define application clusters

**6 STEPS MIGRATION PLAN**
1. Application disposition
2. Choose target platforms
3. Define non-production strategy
4. Prioritize clusters
5. Enable forensics
6. Evaluate third-party applications

**EXECUTION**
- Lower level baseline
- Scripted migration factory where possible
- Canary model for migration
- Governance interlock for prioritization and impact management

**REVIEW & EXPAND**
- Continued application re-factorizing
- Capture lessons learned & business case
- Evolve migration automation to compliance checking
- Adjust operating model, train the broader team
THE STARTING POINT

Any migration project should start with a statement of objectives, along with their related baseline. What are the business goals and objectives for migration?

- **For savings**: Do we understand total cost of ownership (TCO) by application? Do we understand where costs are tied to committed investments and budgets (such as infrastructure-lease contracts)?
- **For time to market**: Is there a link and commitment to the Software Development Lifecycle (SDLC), and is the process part of the change program?
- **For innovation**: Is there an open-ended view that is aligned with the business strategy on where to unleash IT-enabled innovation? Is there a readiness to invest?
- **For any other KPIs** (cost and business metrics, percentage of public cloud migration), what are the baseline metrics and governance?

Enterprises will usually end up with a mix of the above objectives and refine them during baselining. Once these essential questions are answered, a more thorough baseline is required and configurations should be compiled by a team that holds the tacit knowledge of the targeted-application portfolio.

Often, a cloud migration offers a viable opportunity to consider the make vs. buy issue, and reconsider which elements of the value chain should be handled by external providers. Cloud computing allows enterprises to move from outsourced vertical-application silos to outsourced capability layers with well abstracted interfaces, while retaining innovation capabilities in-house. Of course, this requires an understanding of the team’s capabilities to master and lead the transformation and later own innovation.

Some companies tend to underestimate the relevance of innovation in migration projects. Managers tend to concentrate on cost/risk avoidance, and see migration as an internal IT activity. But using migration as a springboard to innovation – enabling new business capabilities and business models – will yield greater results.

DEFINE APPLICATION CLUSTERS

Cloud migration with a clear intent is complex, and requires a more abstract and suitable view of the application environment. Some enterprises have segmented their applications by size or cost. A better approach is to follow natural points of modularity and segment the application base into clusters along natural “fault lines” (see Table 1: “Applications Clusters and Migration Challenges”). This will also segregate different migration approaches.

Here’s a quick rundown of the application clusters. Web front-ends deliver the user experience – how the interaction feels, and how the data is presented. These front-ends are complemented by standard, horizontal, workplace applications, such as collaboration, video, chat, and email. Systems of integration create the insights and connections between multiple data sources, and perform all key custom operations. These “Mudball” apps (Systems of Integration) deserve special attention. Their cloud migration is iterative. Mudball apps contain a host of highly interconnected – and sometimes inadequately documented – functions that requires deeper analysis and modular decomposition. They are supported by grid or other homogeneous calculation tools very suitable for cloud migration. In turn, they connect to the core third-party applications, which are at the heart of almost all enterprise-IT systems (Oracle and SAP exemplify this category). These third-party...
applications rely on systems of record that hold the bulk of corporate data. Third-party applications require interaction with the vendor and a conscious decision on what to migrate, what to leave to the vendor’s roadmap, and what simply to keep and replace later. Third-party vendors need to explain their cloud plans in detail to prove value.

There are two supporting application clusters: Glue apps, highly customized or in-house interface applications that perform custom integration tasks; and system platforms for abstraction and automation, system and storage management, firewalls, and other security workloads. In the table above, the Migration Intent column calls out the main benefits for each cluster during a cloud migration.

Generally, a cloud migration can create a higher feature velocity when it is combined with modularization and the construction of microservices. A “lift-and-shift” of a workload onto a virtual infrastructure does not create feature velocity. Typical challenges found in a few clusters are data gravity (the need to migrate large amounts of data), and high availability and other infrastructure-related requirements.

Application clusters provide the abstraction level at which business leaders and executives should be involved to set priorities without being exposed to the full granular complexity. Application clusters should allow business units to discuss innovation versus cost focus, and development priorities. Introducing too much complexity at this level creates the pitfall of becoming solely cost/risk avoidance focused.

<table>
<thead>
<tr>
<th>APPLICATION CLUSTER</th>
<th>FUNCTIONALITY</th>
<th>TYPICAL COST %</th>
<th>MIGRATION CHALLENGES</th>
<th>MIGRATION INTENT (% - LEVEL OF COST FOCUS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Front-ends</td>
<td>User Interface and data presentation to internal and external users</td>
<td>10%</td>
<td>Easy to decouple; transactional data flow; stateless</td>
<td>Higher scalability; faster feature rollout; higher resilience (30%)</td>
</tr>
<tr>
<td>Workplace</td>
<td>Collaboration, email, voice, video</td>
<td>10%</td>
<td>Easy to migrate; watch for usage outside corporate firewall</td>
<td>Lower cost; policy-based collaboration with third parties; support mobile workforce (30%)</td>
</tr>
<tr>
<td>Systems of Integration (&quot;Mudballs&quot;)</td>
<td>Customized apps to support transactions in line with business process; middleware systems integrating and exposing logic to front-ends</td>
<td>25%</td>
<td>Complex interdependence between applications; migration modules embrace 80% application communication</td>
<td>Feature velocity; robustness; enable process transformation (50%)</td>
</tr>
<tr>
<td>Calculation/Grid</td>
<td>Perform complex model computation on large data sets</td>
<td>5%</td>
<td>Migration of large data sets: time consuming; data gravity</td>
<td>Scalability; cost management (50%)</td>
</tr>
<tr>
<td>Third-party Applications</td>
<td>Dedicated functionality delivered by external vendor, for example transaction processing, billing, and ERP…</td>
<td>7%</td>
<td>Dependent on vendor roadmap may require leapfrogging SW revisions - issues with backward compatibility</td>
<td>Lower cost (50%); feature velocity (50%)</td>
</tr>
<tr>
<td>Systems of Record</td>
<td>Billing, order, and account management; batch processing</td>
<td>28%</td>
<td>Security implications; data gravity implications</td>
<td>Lower cost (50%)</td>
</tr>
<tr>
<td>Glue Apps</td>
<td>Highly networked small-scale apps, often developed in house</td>
<td>5%</td>
<td>Customization, often developed in-house</td>
<td>Keep functionality and support overall architecture (10%)</td>
</tr>
<tr>
<td>System Platforms</td>
<td>DNS, authentication, monitoring, asset mgt, API gateways, development workflow, firewalls, load balancers</td>
<td>10%</td>
<td>Security implications; complex to move (HA always-on requirements)</td>
<td>Lower cost (20–40%); feature velocity</td>
</tr>
</tbody>
</table>
MIGRATION PLANNING

Once you have clarity on the business benefits and have grouped applications into clusters, you are in a position to create a master plan that details the sequencing for migration. Thorough preparation minimizes risk; resist the urge for immediate action. Execution without a clear end-goal guarantees failure. Migration planning involves the following:

1. **ASSESS APPLICATION DISPOSITION**

Conduct a top-down inventory of all applications, and require the application owners to provide a clear future disposition of that application:

   - Is the application scheduled for replacement/phase-out or to depreciate?
   - Is there a need to re-factor the application to enable velocity gains?
   - Will the application migrate in standard fashion and move to a container?

   These dispositions require budget estimates and must have an approximate (quarterly) time-frame identified. This process step feeds directly into the target application list for migration and informs the priority.

2. **CHOOSE TARGET PLATFORMS**

As evidenced above, migrating each application cluster has its own motives and challenges. Therefore, choosing the appropriate target destinations for each cluster maximizes the overall migration ROI.

### TABLE 2: MIGRATION STRATEGY PER CLUSTER

<table>
<thead>
<tr>
<th>APPLICATION CLUSTER</th>
<th>TARGET PLATFORM</th>
<th>MIGRATION APPROACH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Front-ends</td>
<td>Public cloud (with appropriate private links back to on-premises DC)</td>
<td>Host the new URLs in the cloud, canary test it, and swing traffic over</td>
</tr>
<tr>
<td>Workplace</td>
<td>Commercial-off-the-shelf (COTS) offerings for cloud-based collaboration and messaging</td>
<td>Rebuild: Retire apps, migrate configurations and data, redefine security policy</td>
</tr>
<tr>
<td>Systems of Integration (“Mudballs”)</td>
<td>Cloud-based container solutions</td>
<td>Subsequent isolation and containerization of microservices, stepwise re-architecting of Mudballs. Containerizing makes them portable for on-premises or cloud. Move to cloud when majority of apps have swung over</td>
</tr>
<tr>
<td>Calculation/Grid</td>
<td>Public cloud or container platforms (e.g. Hosted Apache Spark) to manage distributed computing</td>
<td>Refactor code to new virtual infrastructure</td>
</tr>
<tr>
<td>Third-party Applications</td>
<td>Agree migration with vendor, avoid a lift and shift, which does not add new capabilities. Ideally also port on container environment</td>
<td>In agreement with third-party vendor, move to cloud only when NPV makes sense</td>
</tr>
<tr>
<td>Systems of Record</td>
<td>Re-platform and move to public-cloud data services only when ROI makes sense</td>
<td>Potential re-platforming, data migration strategies. Disaggregate SOR needs for new born-in-cloud applications and de-centralize away from on-premises legacy systems</td>
</tr>
<tr>
<td>Glue Apps</td>
<td>Cloud-based infrastructure as a service (IaaS) or container services</td>
<td>Lift and shift, rebuild only where it is cost effective. Use opportunity to consolidate business logic to frontline systems, and try to retire</td>
</tr>
<tr>
<td>System Platforms</td>
<td>Re-architect for cloud platform; for example, move from physical to virtual firewalls</td>
<td>Use forensics to discover detailed configurations, simplify, standardize, and port</td>
</tr>
</tbody>
</table>

Copyright © 2018 Oliver Wyman
3. **DEFINE NON-PRODUCTION STRATEGY**

Each of the above clusters has a non-production presence (also commonly referred to as staging or dev/test), built out to varying degrees depending on the stage that environment is in the development lifecycle. For example, application instances in early-stage environments may have dummy back-ends but are functionally a replica of the production stage. These dev/test environments are a great place to “prove out” cloud before moving production workloads.

It is, in fact, necessary to move these non-prod environment stages to cloud prior to the actual production instances. This ensures you identify and iron out any issues early, safely using the standard dev/test process. DevOps team sprints must be dedicated for cloud migration; resist every urge to introduce new software features during these sprints!

You can achieve non-trivial gains with non-production migrations, since there are typically many more assets in these environments than production. A major international bank had two-thirds of its data center assets tied into non-prod systems.

A critical strategy is to introduce load balancers (or even API gateways) in front of each application end-point and reconfigure the neighboring applications to point to these load-balancers. Once legacy and cloud instance are both connected to the load balancer, it is easy to test and complete migration. Cutting over to (and back from) the cloud instance of that application is greatly simplified, and this greatly increases flexibility in testing the migration.

Finally, run a catalog of reliable automated tests on the legacy environment, so that application business flows can be confirmed after the migration. Without this set of automated tests, the migration program becomes heavily dependent on having QA teams run manual testing after each migration. Manual testing can add major costs and delays in a migration.

---

**EXHIBIT 2: MIGRATION OF PREPRODUCTION ENVIRONMENTS**

**Migrate Preproduction First**

<table>
<thead>
<tr>
<th>Migration</th>
<th>Design</th>
<th>Test</th>
<th>Stage</th>
<th>Preproduction</th>
<th>Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

Progressive Automation, Automated Testing, Cloud Migration
4. PRIORITIZE CLUSTERS

Typically, when prioritizing which cluster to start with, there are three popular approaches:

- **By size:** This simplistic approach attempts to gain ROI early in the migration initiative. However, the largest applications are often the hardest to migrate, and offer the least cost and agility benefit, as they are already optimized on-premises and are running steady with constant load.

- **By cost:** An application with high cost can look like an attractive first target. However, the most expensive applications often have the most customization, and those customizations don’t fundamentally change when this application is migrated to the cloud, unless one fully re-factors or re-architects the application.

- **By anticipated cloud benefit:** Often the optimum target applications within this list are those with low customizations, high variability in capacity use, and/or rapidly growing footprint.

The recommended approach is to segment the migration relying on typical enterprise-application clusters, identify quick wins (such as systems of engagement), and then move to more complex segments, such as systems of integration, high computing grids, and glue apps. Systems of record are typically in the last migration wave, and some assets within this cluster category never migrate to the cloud. Often, regulatory and privacy concerns are quoted. (The authors are of the view that public clouds can be as secure as private IT). Also, as mentioned earlier, some clusters (such as systems of integration) require re-factoring prior to migration for optimum returns.

5. ENABLE FORENSICS

ROI from cloud migrations often assumes infrastructure service-level agreements (SLAs) would be equal or better than hosting on-premises. However, owing to “spaghetti” application architectures, inefficient database calls or various other reasons, application throughput can sometimes take a nosedive post-migration! This destroys the business case. It is important, therefore, to clearly enable transaction-performance measures across the application cluster fault lines, allowing for quick diagnosis and isolation, and remediation at the atomic level. Front-end interfaces typically have plenty of telemetry set up; it is the middleware layer downward where forensics must be set up. One popular method of improving visibility is to introduce an API gateway layer in front of the downstream application.

6. EVALUATE THIRD-PARTY APPLICATIONS

Several organizations rely heavily on third-party applications for core business functions, and slow-moving vendors or poorly shaped contracts can severely hinder modernization efforts. Statement of Work (SOW) Contracts with these vendors can be crafted (or modified) to push them to deliver on DevOps-style increments, and to require modernization and visibility into their software-development lifecycle (SDLC) practices (such as continuous integration, automated testing, and containerization). Abstraction from the infrastructure and environment automation is a critical requirement within these contracts. If these third-party applications are developed, tested, and deployed on-premises using modern SDLC principles, moving them to the cloud is typically an inexpensive,
and often a welcome, proposition for the vendor. If not, it’s better not to migrate the applications. Some vendors promise a lift-and-shift to their own proprietary cloud. This provides short-term gains, but makes it even harder to evolve in the long term.

EXECUTING MIGRATION

In this paper, we take a high-level look at migration. A migration will evolve over time, so the use of a migration-planning tool is helpful. These tools can capture dependencies that will rebuild the integrated plan after changes have been made. The tools will also accommodate for an ever finer granular baseline, reflecting configuration and dependency changes that will occur over the lifetime of the migration.

- To execute migration, we mostly use the “canary” (taken from “canary in a coal mine”) model (see Exhibit 3: “Canary-Based Migration”)
- Gradually move traffic from on-premises to the cloud, dividing traffic between on- and off-premises; use “canary” where possible
- Measure performance metrics carefully, with special attention on those expensive calls to SOR (systems of record)
- Switch over the entire user base once the canary test passes and performance metrics are satisfactory

To migrate standard clusters, build a scripted and automated migration factory and pre-stage migration using factory stages.

A governance board that links migration plans back to the business, tracks KPIs, and works around any business constraints is also advisable. Finally, set up 24x7 vendor support during migrations to accelerate the process and hedge risks, especially when the window for migration is tight.

EXHIBIT 3: CANARY-BASED MIGRATION
SUMMARY: SMART MIGRATION STRATEGY

Here’s a quick review of the process described in this paper.

Break applications into clusters along natural fault lines. Choose a tailored migration approach per cluster. Only migrate where there is business value; third-party applications and systems of record sometimes do not offer business value for cloud migration. A cloud migration never stands alone; it usually interlinks with many other aspects of the enterprise-IT architecture. Enterprises should use the momentum of a cloud migration to further IT automation, staff training, and evolution of their operating model.

Start with easy, low-dependency clusters: Web and Dev/Test. Transform complex apps by using microservices, then migrate them. Move to SaaS as much as possible; IT no longer has to run everything. Modularize “Mudballs,” meshing microservices. And say no to lift-and-shift migrations that rarely add value.

It is essential to keep the cloud migration in line with business goals, and focus on generating top-line impact along expected cost reductions. Top-line impact results from greater development speed and easier addition of new features.

EXHIBIT 4: BENEFIT AND EASE OF MIGRATION BY CLUSTER

Start migration process with less complex applications to accelerate transformation.

Impact

Ease of Migration

Systems of Integration (“Mudballs”) 25%
Systems of Record 28%
Keep Legacy – No Cloud Migration

Start migration process with less complex applications to accelerate transformation.

Ease of Migration

Impact

Keep Legacy – No Cloud Migration

Start migration process with less complex applications to accelerate transformation.

Ease of Migration

Impact

Keep Legacy – No Cloud Migration

Start migration process with less complex applications to accelerate transformation.

Ease of Migration

Copyright © 2018 Oliver Wyman
Omer Ansari is a Director at T-Mobile, where he is responsible for the digital transformation of all T-Mobile’s U.S revenue-generating applications – from legacy-monolith patterns towards cloud-based, distributed micro-services. This utilizes Continuous Delivery, automated testing, service virtualization, data forensics, and advanced data-management techniques. Prior to T-Mobile, Omer held multiple roles at Cisco related to building, designing, and operating cloud services.

Uwe Lambrette is a Partner in Oliver Wyman’s Digital Acceleration and Transformation, as well as the CMT Practice. Before Oliver Wyman, he built his experience for the high-tech industry. After his PhD in communications design, he worked in a major consulting firm, focusing on technology and operating model transformation. He spent 10 years at Cisco in strategic business development for cloud, and delivering and implementing cloud- and network-software solutions.
Oliver Wyman is a global leader in management consulting that combines deep industry knowledge with specialised expertise in strategy, operations, risk management, and organisation transformation.

For more information please contact the marketing department by email at info-FS@oliverwyman.com or by phone at one of the following locations:

EMEA  
+44 20 7333 8333

AMERICAS  
+1 212 541 8100

ASIA PACIFIC  
+65 6510 9700

www.oliverwyman.com

Copyright © 2018 Oliver Wyman
All rights reserved. This report may not be reproduced or redistributed, in whole or in part, without the written permission of Oliver Wyman and Oliver Wyman accepts no liability whatsoever for the actions of third parties in this respect.

The information and opinions in this report were prepared by Oliver Wyman. This report is not investment advice and should not be relied on for such advice or as a substitute for consultation with professional accountants, tax, legal or financial advisors. Oliver Wyman has made every effort to use reliable, up-to-date and comprehensive information and analysis, but all information is provided without warranty of any kind, express or implied. Oliver Wyman disclaims any responsibility to update the information or conclusions in this report. Oliver Wyman accepts no liability for any loss arising from any action taken or refrained from as a result of information contained in this report or any reports or sources of information referred to herein, or for any consequential, special or similar damages even if advised of the possibility of such damages. The report is not an offer to buy or sell securities or a solicitation of an offer to buy or sell securities.

This report may not be sold without the written consent of Oliver Wyman.