Impact, delivered

Ecosystem-Oriented Architecture in the Public Sector

How public sector agencies are building scalable, resilient, flexible, and future-ready cloud ecosystems for the Age of AI (and beyond)

codec

Contents

Introduction

Think back to the dawn of the consumer internet. An era that those of us who were around to experience it recalls memories of Netscape and the horrid buzzing of a dial-up modem. The year was 1996.

Then there was "Web 2.0", a phrase that sounds almost silly to say, now. This marked our transition from words and images on a "page" to pages with which we could more readily interact. The origins of the modern web app. The year was 2004.

The early to mid-aughts gave rise to the public cloud, to our long (and, unfortunately, still underway) transition to globally scalable platforms including Azure and AWS from the on-premises computing infrastructure upon which we had relied for decades. Let's call it 2014.

Platform-first technologies like Power Platform began to emerge in 2019 (we called them "application platform as a service", or "aPaaS", back then), followed by generative artificial intelligence in 2022. We spent 2023 gasping for air. Catching our breath. Wondering how we might set ourselves up for the next wave.
Notice the timeline.

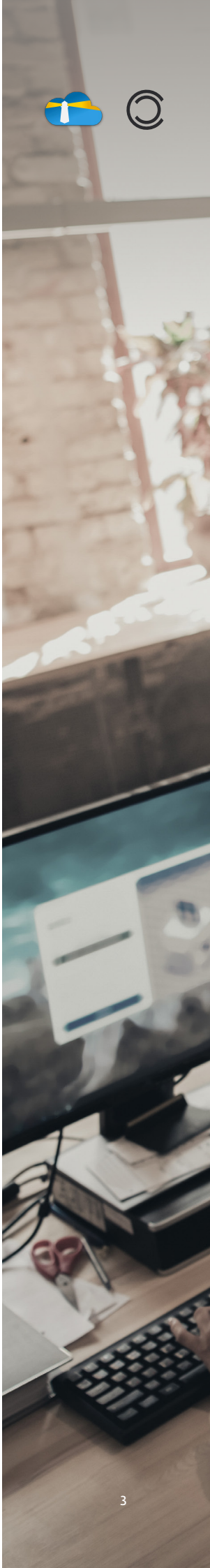
We were working with eight to ten years between major disruptions from the dawn of the consumer internet. But these "wave periods", that is, the time between the crest of two waves, have shortened to three to five years since the rise of the public cloud. It makes sense: As the evolution of computing technology and capacity picks up steam, it similarly accelerates.. Innovation begets innovation. Generative AI was only made possible by the incredible computing power and connectivity available in the cloud. Now, AI is further accelerating this pace of change, shortening the time we have available before new waves crash upon the shore.

The grace period for organizations to get their act together and position themselves for the next wave is growing much shorter; the margin for error is much narrower.

When I think about the non-technical barriers that so many organizations can't seem to get over when it comes to platform-first, I really wonder how many will miss out on the AI wave because they lack the wisdom or the willpower to make the most of it.

Accelerating wave periods present challenges to even the nimblest of private sector organizations. It will absolutely wreck traditional public sector models that have hitherto been anchored in careful, deliberative decision making, multi-year budget cycles, and lengthy software implementations.

The "cloud ecosystems" being developed by the organizations that are leaning into this transformation—which is to say, the organizations that will survive and thrive—rise to the occasion of these accelerative trends while solving significant problems faced by public sector organizations around the world.



- **Antiquated and disconnected systems** lead to poor employee satisfaction, even worse citizen and constituent outcomes, and a persistent inability of organizations to extract value from their data even as their technical debt make them more expensive to maintain over time.

- **Monolithic "point" solutions such as ERP, HRS, and CRM:** Costly, inflexible implementations that do not age well, nor can they be easily replaced because doing so risks toppling an agency's entire IT tower, which in turn further drives up the cost of the alternative.

- **Incredible levels of risk** incurred as agencies seek cheaper workarounds to their monolithic point solutions, often finding "solutions" such as the "SharePoint app" or the "Power App built with SharePoint as its data store" (because "the licenses ~~were~~ seemed free") whose ~~cost savings~~ apparent cost savings have been seductive, but which have ultimately led organizations to expose massive amounts of their data stored in unsecure locations.

I'm going to linger on that last thought, because I cannot make this point enough. Organizations that have overbuilt using SharePoint or Excel spreadsheets as a data store have exposed themselves to perilous risk. SharePoint and Excel hydrate an organization's Microsoft Graph with data. The Graph augments generative AI workloads such as Copilot. I implore you to not be the organization that learns this lesson the hard way.

Microsoft Graph is like a central hub that organizes and connects data from across Microsoft 365 services. It stores data from SharePoint lists, Excel files, Teams conversations, OneDrive, Outlook, and more, all in one place. Think of it as an interlinked productivity data network allowing you to pull data from different sources, like files, messages, or tasks, and use it seamlessly in your apps. It's all about making Microsoft 365 data easy to find, use, and manage in one unified place.

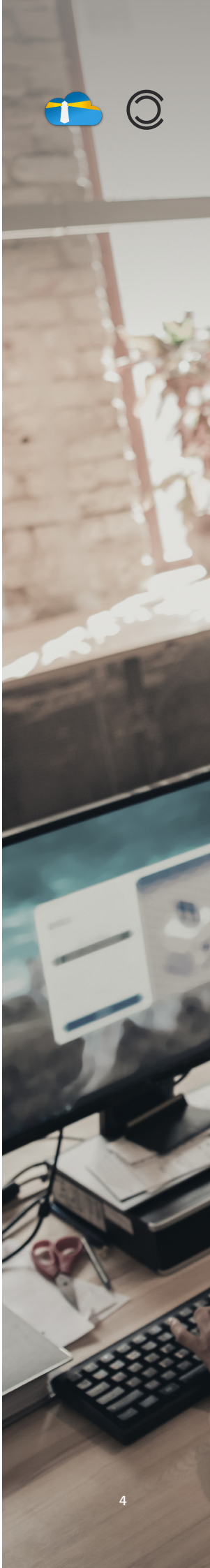
Ecosystem-oriented architecture (EOA) presents a future-ready path forward, allowing organizations that embrace it to more readily absorb successive waves of technological change in artificial intelligence, data, and beyond. EOA calls on these organizations to move their monolithic point solutions from the center of their architecture to its outskirts, place data at the core of their cloud ecosystem, and adopt composable development approaches such that when one workload is added or removed, the rest of the ecosystem continues to function and evolve.

This white paper explores the principles, application, and real-world examples of ecosystem-oriented architecture through a public sector lens. I could have chosen to write generically, applying these principles broadly across many sectors, but I believed that the uniqueness of the public sector environment called for its own exploration of EOA.

Wave periods between major innovation in the cloud are growing shorter. We no longer have the luxury of waiting it out, of adopting later. Cloud ecosystems built on strategic foundations create the conditions to absorb successive waves of change.

Cheers,

Andrew Welch, Author, CTO, Cloud Lighthouse
Follow on LinkedIn
Produced in partnership with Codec



What is Ecosystem-Oriented Architecture?

Understanding ecosystem-oriented architecture requires one to fundamentally shift their thinking away from legacy information technology and software implementation practices. It is a shift that forces decision makers to fight their urge to resort to scattershot methods to resolve seemingly urgent problems, and instead keep a holistic view, forcing even the smallest changes into a repeatable pattern.

Point solutions solve for a specific number of known things.

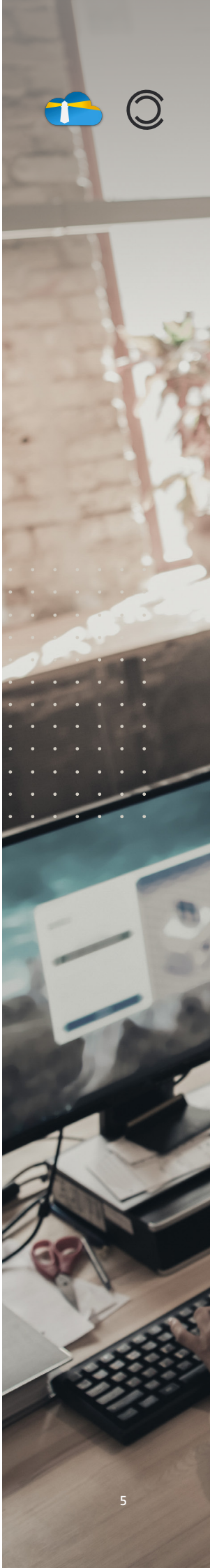
An ecosystem solves for an **infinite** number of **unknown** things.

So, ecosystem-oriented architecture is an architectural approach integrating technologies across an organization's entire cloud estate, prizing flexibility, the adoption of new tech, and thoughtful application of the principles of EOA that we will discuss next.

Principles

Let's begin with the core principles of EOA. Think of these principles as "lenses" through which ecosystem architects, technical leaders, and decision makers should consider their architectural choices and investments. In other words, the best ecosystem architects will consider thoughtfully and apply these principles consistently across the organization's entire cloud estate or ecosystem. This application will often require tough decisions, and while success is unlikely to be found in their uneven or incomplete application, we ought to acknowledge that budgetary, technological, organizational, and even political constraints are likely to prevent purity on most organization's journey transitioning to a complete EOA. But that's okay. As you will see, a cloud ecosystem evolves over time. It is not a big bang that happens the night you go live.

1. Principle of Platform First
2. Principle of Composability
3. Principle of Evolution
4. Principle of Restraint
5. Principle of Artistry
6. Principle of Following the Money



1. Principle of Platform First

The core platform elements must come first, even though it makes the first workload expensive.

Imagine an organization that has spent years in a hybrid cloud and on-premises situation. Perhaps there's an aging on-premise enterprise resource planning (ERP) solution in place to manage the finances, an even older human resources system (HRS) that bedevils the HR division and colleagues alike, and "customer relationship management" (CRM) functions handled solely in Excel spreadsheets owned by individual teams and compiled to produce reports when the boss asks for them. Now, an upcoming modernization program that you've been budgeting for two years calls for moving that ERP to a modern, cloud-based solution. The "old way of doing things" would be to stand up the minimal cloud infrastructure (could be IaaS, SaaS, PaaS; the particulars don't matter right now) required to support the new HRS, develop and deploy the workload, and call it a day.

But implementing a workload before you've built the core platform services on which the workload (and those that follow) relies is like buying an expensive living room set before you have a home to put it in. Yet, despite this obvious analogy, organization after organization—and consultancy after consultancy with which they partner—have a nasty habit of skipping the foundation, skipping the home construction, and going straight for the living room set.

Our Principle of Platform First calls on us to acknowledge that EOA often means higher up-front costs followed by dramatically reduced costs for future workloads. This is because the early workloads are often built in parallel to the cost of building the ecosystem that, once built, offers "composable" components that are reused again and again. EOA is designed to be open and transparent, allowing organizations to actively participate in its development. Its modular nature means it can be tailored to fit specific needs, and most importantly, it empowers organizations by giving them control over their technology, rather than having technology imposed on them.

Ensure that you allocate your "year one" budget accordingly, so that you don't spend so much on the sofa and lamp—your ERP, HRS, CRM, etc. workloads—that you neglect the core platform, integration, and data distribution services that drive the most value from your ecosystem in the long-run.

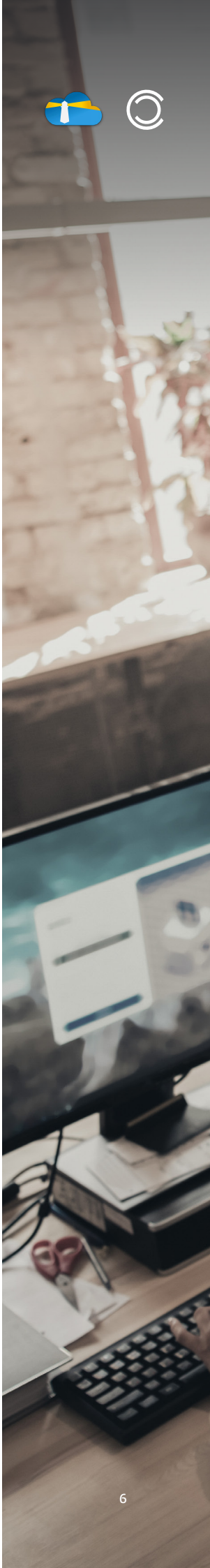
2. Principle of Composability

The whole ecosystem is greater than the sum of its parts. Modularize workloads to be added or removed with little impact to the whole.

Gartner popularized the idea of *composability*—that is, "creating an organization made from interchangeable building blocks"[1]—several years ago.

I wrote earlier how former eras of IT were filled with examples of architectures that placed mammoth ERP solutions in the center of the estate, and treated any further workload as an extension of that ERP system.

1. www.gartner.com/smarterwithgartner/gartner-keynote-the-future-of-business-is-composable



Composability calls on us to invert this thinking, beginning instead with core platform services, integration, and data distribution and placing those services at the center or our ecosystems. ERP then becomes a node, albeit an important node, just a collection of workloads alongside any other core business system.

I could have called this the **Principle of Keeping the Core Clean**, wherein we implement core business systems such as ERP with as little customization as possible, let it do its thing, certainly do not try to shoehorn in functions for which it was not designed, and then build composable solutions using mastered and integrated data to meet the organization's more unique needs.

EOA requires us to modularize individual workloads so that they are, individually, more flexible, more easily customized, and even retired when no longer needed, then assembled in such a way where the organization relies on the ecosystem as a whole to do its heavy lifting, rather than on a handful of monolithic apps.

3. Principle of Evolution

The ecosystem need not be fully functional on day one to be viable. Avoid lengthy "analysis", do what you must to get started, then go from there.

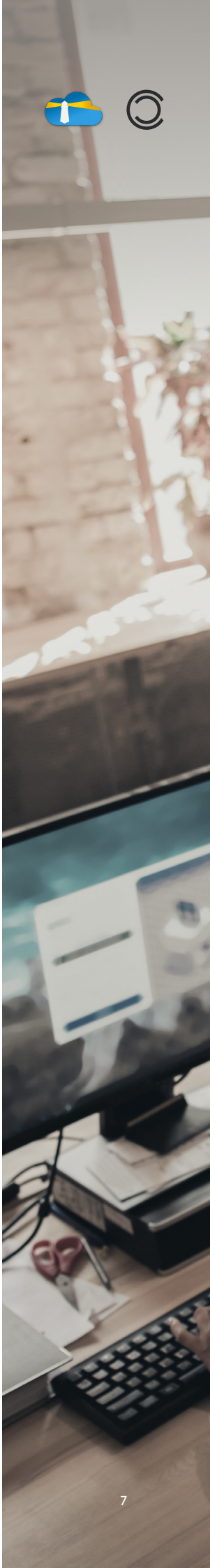
How many times have you seen a public sector agency spend years getting ready to build something—scoping out an immense technology initiative, gathering requirements, researching possible solutions, budgeting, requesting proposals, awarding contracts—only to realize that the thing everyone thought was needed three years ago is no longer fit for purpose?

The Principle of Evolution suggests that we allow the ecosystem to evolve with the organization's needs (and its budget), rather than mapping out every minute detail of many specific workloads. We're not advocating unchecked experimentation here, rather that we guide the ecosystem's evolution with principles and desired outcomes rather than years' worth of accumulated requirements.

This principle is like that of the "minimum viable product" (MVP), which has been floating around for many years. Swap "platform" in place of "product" and you have the idea. Too often architects, particularly those grounded in the art of implementing big point solutions, get hung up with the inner workings of core business systems and fail to see the forest for the trees. Months or even years are spent discovering, analyzing, designing every bit of minutiae functionality associated to specific workloads.

Evolution challenges us to first build the core platform services required across the ecosystem, deploying alongside this one or a handful of workloads that strike a good balance of providing value to the organization but without being so complex that they take years to deploy. Then go from there.

You may be tempted to think that this is a too big of a change for the Public Sector, but the secure, risk-reducing quality of this approach says otherwise.



4. Principle of Restraint

Not all data or application functionality is needed everywhere or in real-time. Be discerning between what you could do and what is needed.

If I have a dollar (or a euro, pound, franc, etc.) for every time someone has insisted to me that something simply had to be real-time...

Many will have worked with architects who lack this restraint. When given a choice between technologies, they'll always choose the one with the most features, the newest technology, the data insights produced in real-time, etc. This reflects point solution architecture and task-based thinking where the purview of the architect ends at the bounds of their own solution. But it is antithetical to EOA where the results of a system-based ecosystem working as one is far more important than the task-based outcome of any single workload.

Let's briefly consider that common scenario of real-time data analytics. Many organizations struggle with data latency in legacy analytical workloads, but when pulling back the curtain here, we find that we're talking about days between data collection and data visualization. Building a data platform that reduces this latency to hours is sufficient, simpler, and cheaper in many cases. In other words, if we can improve data latency from days to hours, is real-time truly necessary? It may be, but you are wise to take the decision thoughtfully before committing to often more costly real-time integration.

The principle of restraint challenges us to discern the difference between what we could do and what is actually needed in the context of the ecosystem and the business need.

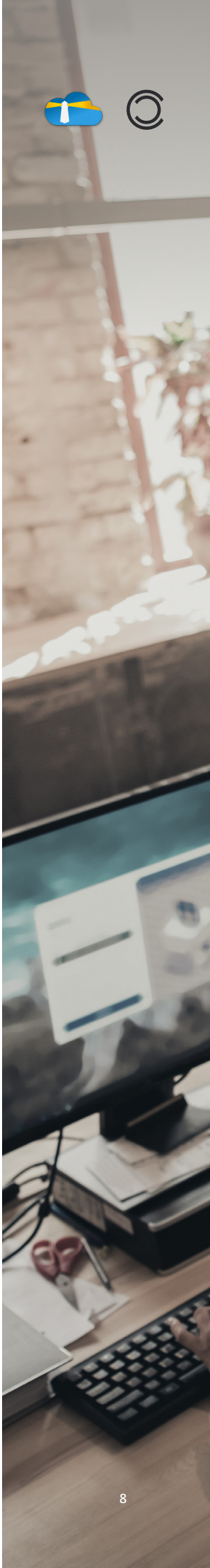
5. Principle of Artistry

Many ecosystems look alike in their common elements. Artistry is in how these are combined in practical terms for the organization and industry.

The principle of artistry was not obvious until after our notional "Reference Ecosystem", which we will discuss later, had been applied to various organizations across different industries. "Ecosystem maps", as we call them, looked very different in the early days of my work with EOA. Over time, as the Reference Ecosystem improved, however, I began to see ecosystems looking more and more like the composite.

I have found that cloud ecosystems across individual organizations share much in common. Nearly all of them are built with some selection of the same core platform services such as Microsoft Purview for data governance, lineage, cataloging, etc. Nearly all of them contain some combination of event, logic, and batch integration services. And most organizations rely on the same set of core business systems.

The "artistry", therefor, is in combining and prioritizing these technologies to support different aspirations and to achieve different outcomes. For example, a regulatory body may be tracking the organizations it regulates and any enforcement actions, a court system is preoccupied with managing its case backlog, military and defense



organizations are concerned with the people and supplies deployed to far-flung places, law enforcement will prioritize completing duty rosters without burning out their force, national security and intelligence is focused on communicating threats quickly and to the right parties, local government (or "councils" in some regions) need to be careful in planning and costing their work, healthcare institutions must generate and prioritize clinical care needs.

These are, of course, very different problems to solve. However, by rigorously applying the principles of EOA, focusing on data with composable applications built atop that data, and avoiding one-size-fits-all monolithic point solutions we can inject the "artistry" into combining various technologies to produce the fit-for-purpose solutions that are needed.

The principle of artistry challenges ecosystem architects to use similar paint colors (the underlying technical services) to paint ecosystems that match the mood of their organization's aspirations and desired outcomes.

6. Principle of Following the Money

Technology vendors like Microsoft focus their investments on what they believe are the technical capabilities of the future. Hitch your wagon to these.

We're not talking about broad technology categories such as "AI", "data platform", or "low-code" (a phrase that I abhor). No, the principle of following the money leads you to very specific technologies with which you should be building your cloud ecosystem.

EOA really began to ignite as a concept in 2023, which happens to be the year in which Microsoft announced its "Fabric" data platform capabilities. This proved a perfect example of the Principle of Following the Money, wherein one was able to place bets in the form of investment and upskilling on technologies where Microsoft was itself investing. At the time of writing, Microsoft technologies such as Fabric, AI Search, and Dataverse were the recipients of significant investment, which in turn leads to the conclusion that these are ideal technologies upon which ecosystem architects can rely for the foreseeable future.

Perhaps a better example from this era is Microsoft Purview, which, while lacking the marketing investment that Microsoft has made in technologies like Fabric and Power Platform, has nonetheless quietly become Microsoft's one-stop-shop for data governance.

In the public sector, robust governance, clear data lineage, and tools like Microsoft Purview are essential for ensuring transparency, compliance, and accountability. Governance frameworks help manage data access and usage, while data lineage provides a clear view of where data comes from, how it's transformed, and where it's used, ensuring data integrity and trust. Microsoft Purview adds an extra layer by offering comprehensive data cataloging and tracking capabilities, which are crucial for maintaining oversight and control in highly regulated environments, ultimately helping public sector organizations meet their compliance and security obligations.

Though it is impossible to guarantee that any technology will live forever—because no technology really will, at least not in its current form—the principle of following the money suggests that we are best served to learn and construct future-ready cloud ecosystems that rely on technologies where Microsoft (or your preferred cloud technology vendor) is investing heavily today.

Reference Ecosystem

I mentioned the "Reference Ecosystem" when earlier discussing the Principle of Artistry. We'll discuss this in more detail below. Let's nail down some key terms here so that we're clear on the difference between them:

- **An Ecosystem Map** is a high-level architectural diagram of an organization's cloud ecosystem, and something that every organization must create at the start and continuously evolve as they progress on their journey;

- **The Reference Ecosystem** is a specific ecosystem map that my team and I have created as a composite of many different cloud ecosystems across many different industries.

The "map" metaphor is instructive here. It is used to distinguish an ecosystem map from the various forms of architectural diagrams, nearly all of which tend to include more technical minutiae than a typical ecosystem map. Whereas an architectural diagram provides specific parameters for specific technical solutions, an ecosystem map presents a higher-level, more visionary view of an organization's cloud ecosystem.

To make an analogy to architecture in the physical world:

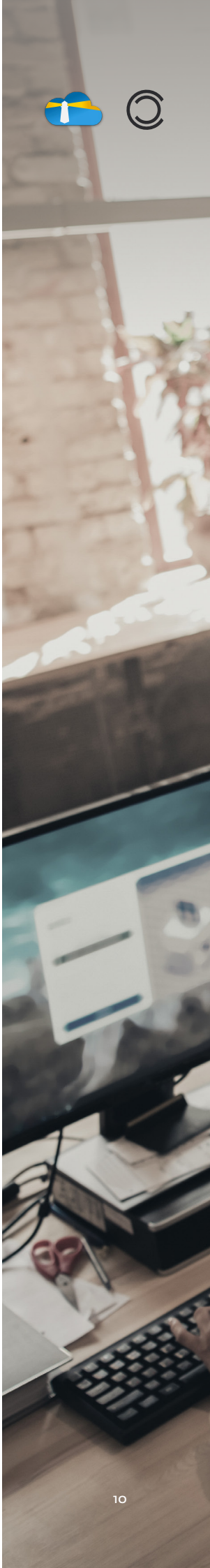
- **Solution architecture** provides schematics—floor plan, dimensions, electrical wiring, ventilation, plumbing—from which a building is constructed;

- **Enterprise architecture** provides plans for specific neighborhoods or systems such as a subway or electrical grid;

- **Ecosystem architecture** and, by extension, an ecosystem map shows us the entire city.

This analogy is fundamental to understanding and practicing EOA. Thinking of an organization's cloud ecosystem as a city, we then conceptualize the next-level-down component parts of the ecosystem as "neighborhoods" (we might have also called them "boroughs"). Cities the world over are pieced together this way: Downtown, Seaport, Southie, etc. in Boston; Greenwich, Soho, Canary Wharf, etc. in London (though you're forgiven if you thought I was talking about New York until you got to "Canary Wharf"); Palermo, Recoleta, Puerto Madero, etc. in Buenos Aires; Norrmalm, Gamla Stan, Kungsholmen, etc. in Stockholm. The list goes on.

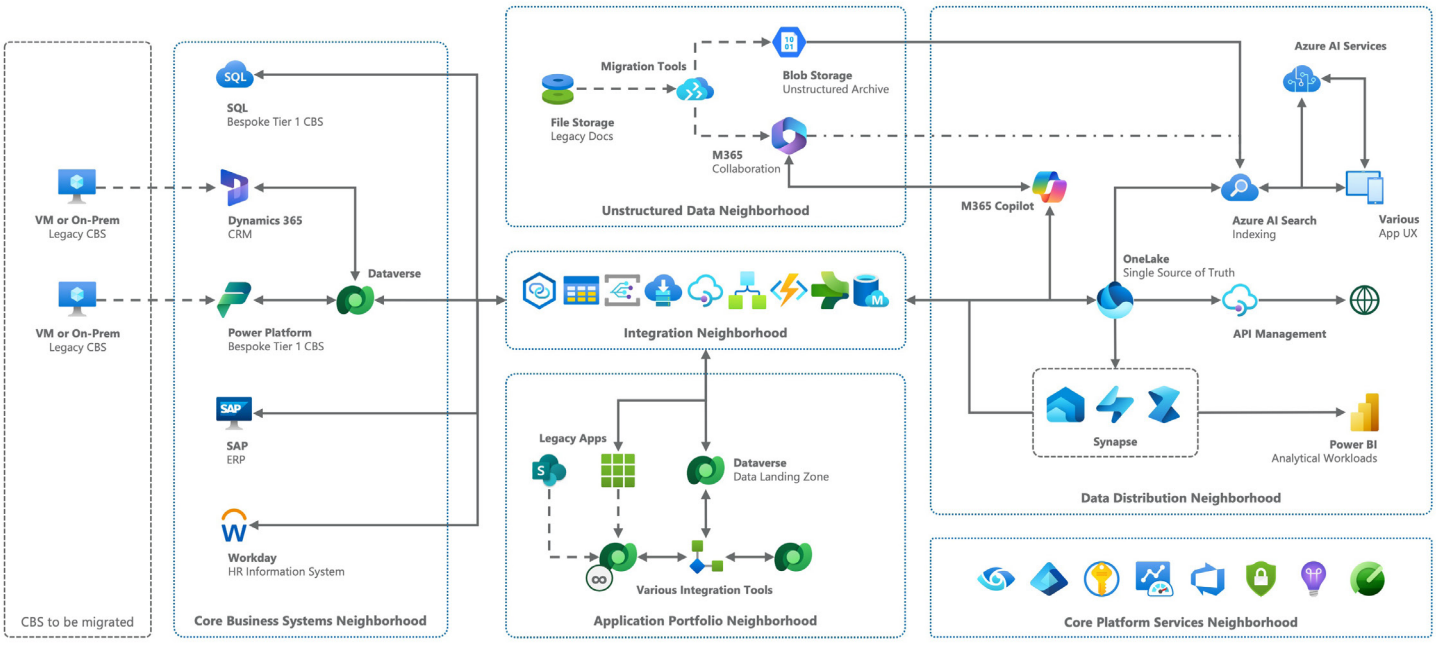
Each of these neighborhoods share the quality of dividing their city into smaller pieces, each often with their own distinct culture, aesthetic, or purpose.



My team applied this same concept to EOA, clustering technologies and workloads devoted to similar purposes into distinct neighborhoods, and tying them together with the flow of data, logic, and actions—our roads, subway lines, waterworks, electricity, etc.—to build coherent cities in the cloud.

To do this, we compared the cloud ecosystems across real-world organizations in different industries and geographies to compile their common features and best practices. We then produced the "Reference Ecosystem", that is to say, a notional cloud ecosystem that is a composite of the cases we considered. This reference ecosystem is the ideal standard, the prototypical notion of what a cloud ecosystem might look like.

Variation surely exists amongst individual organizations and sizes, industries, geographies, and other considerations such as regulatory requirements. To compensate for this variation, we then spent nearly two years applying our reference ecosystem to real-world scenarios, testing the composite model against actual guiding principles, business objectives, and requirements. We incorporated lessons learned from these scenarios to improve and refine the Reference Ecosystem over time, and were pleased to see how durable the model really is. Indeed, over time more and more of the ecosystems we mapped began to converge on and look more like our best practice Reference Ecosystem. We interpret this as evidence of its durability and flexibility to meet the needs of various organizational profiles.



Click to view full page - Appendix 1

In general, we found that most ecosystems are home to six major neighborhoods.

- **Core Platform Services,** including many of the infrastructure, security, governance, management, and monitoring services used across the ecosystem. Largely synonymous with a "cloud landing zone", the Reference Ecosystem above shows (left to right) Purview, Entra ID (formerly Azure Active Directory), Key Vaults, Azure Monitor, Notification Hubs, Security Center, Application Insights, and Power Platform Managed Environments as examples.

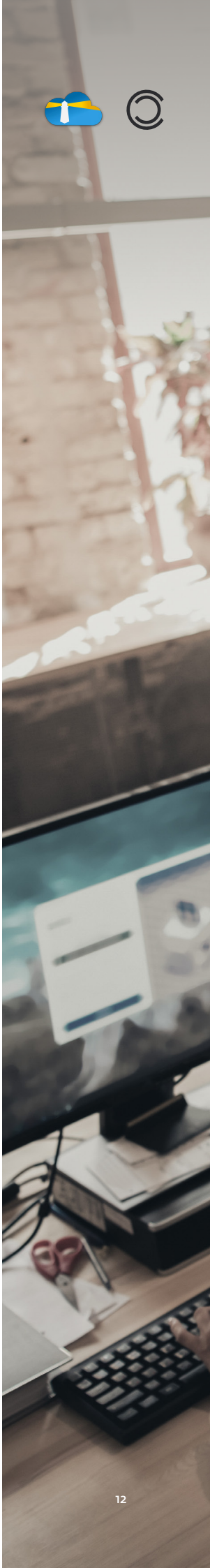
- **Integration,** including services for (left to right in the diagram) technology-specific integration services such as OneLake shortcuts in Microsoft Fabric and virtual tables in Microsoft Dataverse, event-driven integration services such as Event Grid and Service Bus, use of APIs via API Management, logic-driven integration such as Logic Apps and Azure Functions, batch integration relying on Azure Data Factory, and a generic master data management (MDM) solution.

- **Core Business Systems,** including the "Tier 1" business applications common in many organizations such as ERP, CRM, HRMS, etc. The Reference Ecosystem uses custom applications built atop Azure SQL and Power Platform, CRM in Dynamics 365, ERP in SAP, and HR in Workday as illustrative examples of the many solutions organizations rely on in their core business systems neighborhood.

- **Application Portfolio,** which may include "Tier 1" applications but often include solutions aimed at smaller audiences or more niche business processes of the Tier 2 ("business important) and Tier 3 ("productivity") variety. EOA in the Microsoft context ought to rely heavily on Power Platform in the Application Portfolio Neighborhood, though as you see above, we also highly favor Power Platform for core business systems.

- **Unstructured Data,** including documents, files, photos, videos, etc. often housed in legacy network file storage, SharePoint, or Azure Blob Storage. Many of the Microsoft 365 services will operate here, noting that we strongly recommend using SharePoint as a data service for the Tier 1 or Tier 2 workloads found in other neighborhoods.

- **Data Distribution,** which provides for data consolidation in (for example) OneLake, and for all manner of "downstream" data distribution such as search, APIs, data warehousing, analytical workloads, and use in AI-driven scenarios. We've also decided to place Microsoft 365 Copilot in the Data Distribution neighborhood as it—like many other scenario-specific Copilots that also reside there—relies on consuming, interpreting, and otherwise distributing data in response to user prompts. Note that M365 Copilot is hydrated with data from Microsoft 365 via the Microsoft Graph, and may be configured to consume data from elsewhere in the data estate, as well.



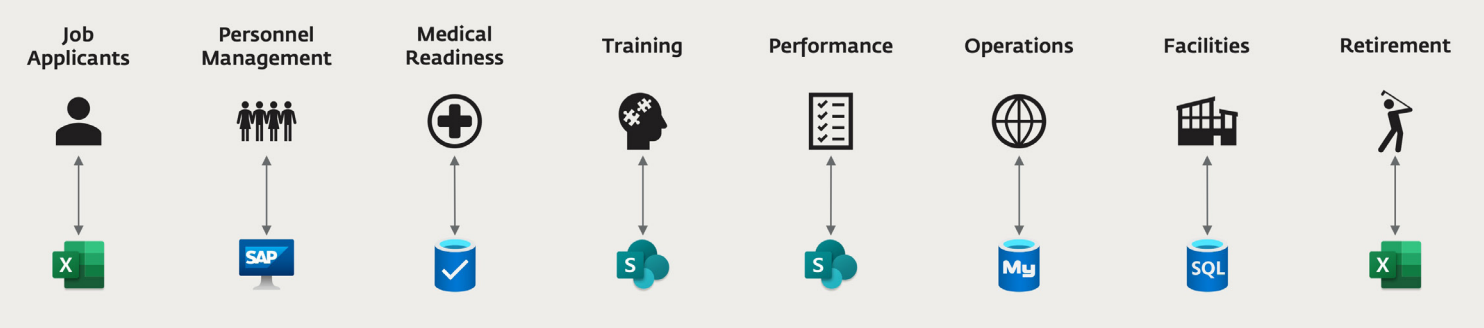
Applying EOA in Real-World Public Sector

We'll now make the concepts discussed in the previous section more real in context of public sector organizations. To do so, I want to spend some time speaking less about technology and begin describing workloads that incorporate functions and scenarios upon which a typical agency might rely on its cloud ecosystem to perform.

Workloads

I considered writing this in the form of agency-specific scenarios, in other words, a separate section devoted to what this might look like in various public sector domains including (based on my experience) civilian government, federal or central government (terminology varies between countries and regions), local and regional services, regulatory agencies, military and defense, law enforcement, and even academia.

Eventually, I concluded that doing so would turn this white paper into a book. So, instead, I've created another composite that we'll use to work through these concepts. You may find in reading that there are one or two specific functions that your agency isn't particularly concerned with, but on the whole, I think you'll easily see something of your own organization here whether you are in the business of administering services, running metro systems, regulating industries from gambling to telecommunications, administering agricultural policy, building roads, soldiering, policing, or teaching kids.



The above diagram shows (icons noted in black) an assortment of workloads common across many public sector organizations. These workloads have grown over time as point solutions implemented largely in isolation of one another. Their specific data storage technologies differ between organizations, but I've used a combination of Excel, SAP, proprietary databases, SharePoint lists, MySQL, and SQL Server to provide a representative sample.

Job Applicants

The place where it all starts for employees and volunteers eager to join the organization. Think about recruiting, candidate sourcing, interview and offer processes, pre-arrival preparation with recruits before their first day, personnel on-boarding, and all the related activities undertaken to progress someone from being unknown to the organization into their first day as a colleague.

Personnel Management

We could have easily called this "Human Resources". Examples here include personnel records, background checks, pay and benefits, awards, reasonable accommodations for colleagues' unique circumstances, and other HR-related processes.

Medical Readiness

Though not a requirement in many private sector settings, some public sector agencies find themselves managing the medical readiness and recovery of their workforces due to the organization's mission deploying personnel (be they sailors, scientists, or aid workers) quite far afield. Examples include core electronic medical records (EMR), vaccinations, psychological screening, clinic scheduling and management. Some agencies are conversely in the business of providing medical care directly to the citizens or constituents (e.g., disaster victims) they serve. I've personally built software solutions that ensure scientists in extreme latitudes are psychologically fit to spend many months in darkness, that soldiers deploying to far-flung regions are properly vaccinated, or that disaster victims are properly attended to once they've arrived in a safe haven.

Training

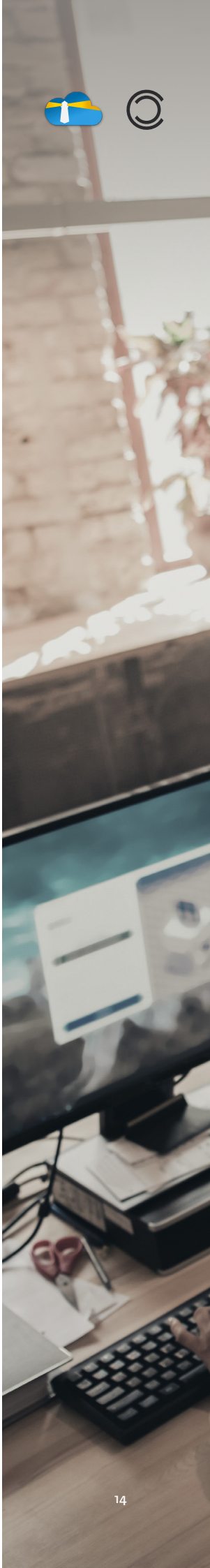
I have found that public sector training needs can range from the mundane tracking of mandatory training to more complex functions around competency or skills, languages, curriculum management, course delivery, and integration with a full-blown learning management system (LMS).

For example, public sector organizations—particularly in high-risk domains such as military and law-enforcement—are generally subject to much more rigorous management of curriculum, tracking of training status, and related activities. It is often insufficient to know that someone completed a required course, rather, it is necessary for the agency to know which version of the curriculum, even down to the level of an individual lesson, a training graduate completed. This capability is not a common feature in your run-of-the-mill learning platforms, so composable architectures are very welcome here.

I'd elaborate with more examples from different public sector domains, but again, I am trying to stay in white paper territory here and not turn this into a book.

Performance

"Performance Management" is one of those domains that means different things to many people, but to paint a picture let's think of performance in this context as the management of employee performance along the lines of career progression, goal setting, employee appraisal, annual review, and disciplinary actions.



Operations

Many of these agencies are tasked with a set of operational missions or functions diverse as arresting criminals, transporting supplies, fighting wars, caring for patients, and collecting trash (or "rubbish", depending on which country's public sector in which you find yourself). These functions are diverse so as to lend themselves poorly to monolithic point solutions. They're often highly inter-related so as to require data integration with other systems, and usually highly specific to the organization so as to make them great candidates for building your own solutions rather than shoe-horning yourself into the process that a vendor built for you.

Depending on the organization's mandate, "Operations" might also include what we'll call "constituent services", activities serving, regulating, or otherwise interacting with their organization's public—citizens, regulated industries, etc. The private sector might call this "customer relationship management" (CRM), but in the public sector context we're talking about IT functions to enable efficient management of citizen casework, regulatory infractions, refugee resettlement, and—gosh—this list would grow incredibly lengthy were I to not cut it off here.

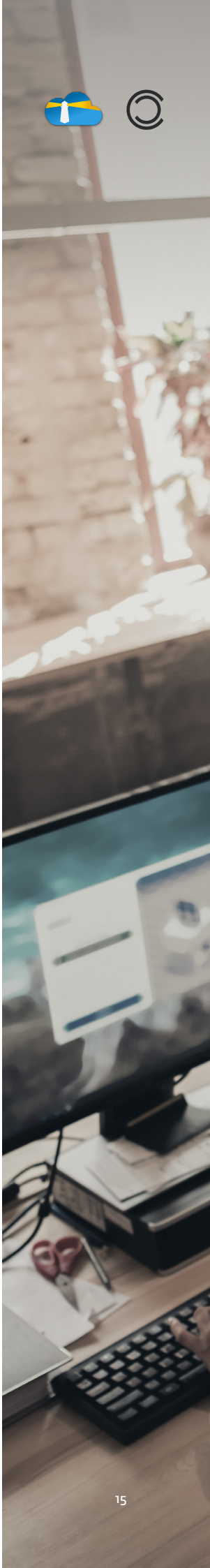
Facilities

The pieces begin to fall into place. Once we're entrusting our cloud ecosystem and the composable solutions with such rich personnel data, a natural extension is to further leverage the technology investment to manage the facilities that this workforce inhabits. I'm thinking here of the real estate footprint, e.g., buildings, campuses, and other physical locations, requiring day-to-day management and long-term upkeep, and security functions such as who goes where, and with which badge.

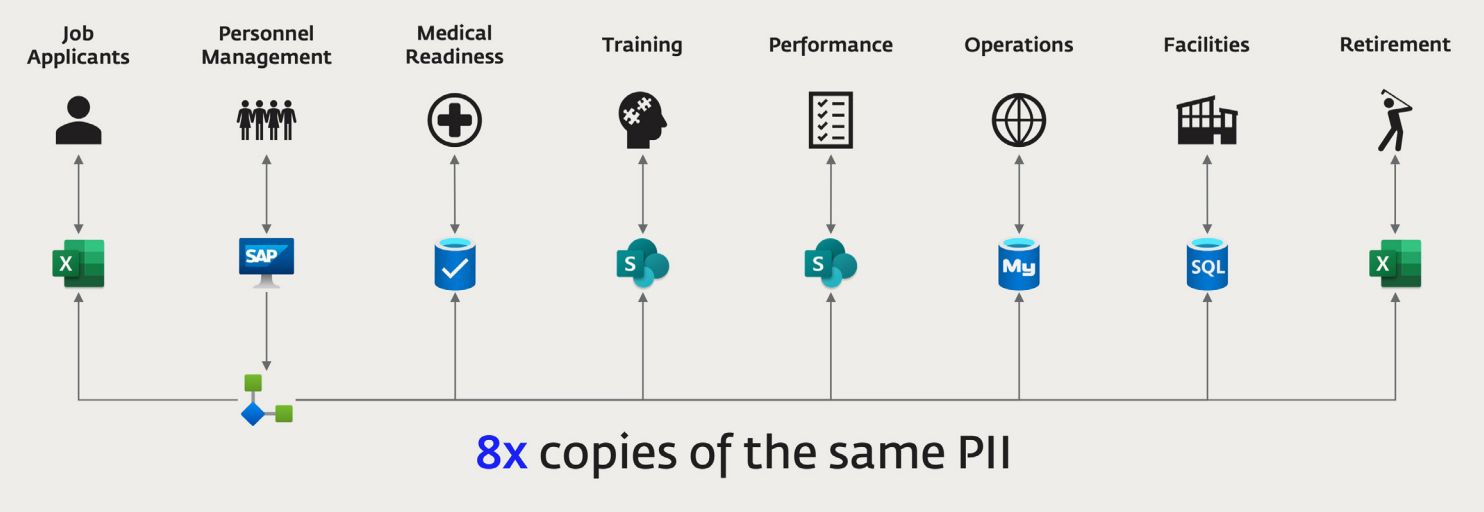
Retirement

We've come full circle with "job applicants" above onto workloads I've broadly grouped into the "Retirement" category. These could include projected departures and the workforce planning that this necessitates: employee offboarding, administration of retirement benefits (a reality for many, but not all public sector agencies), case management for departed colleagues, and the cultivation of the organization's alumni network.

Now, in nearly every organization I've worked with—prior to their transition to an ecosystem-oriented architecture, of course—individual applications in their fragmented collection of point solutions require significant amounts of common data. Personnel data offers a great example, because each of the workloads discussed above require some degree of data or knowledge about the people working in the organization. So, IT organizations build "spaghetti web" point-to-point integrations between data stores using a variety of tools including Power Automate, scattershot use of actual integration tools (event, logic, or batch integration), Excel, and even what we used to jokingly call "sneaker net", in other words, manually moving data from one system to the next via physical media.

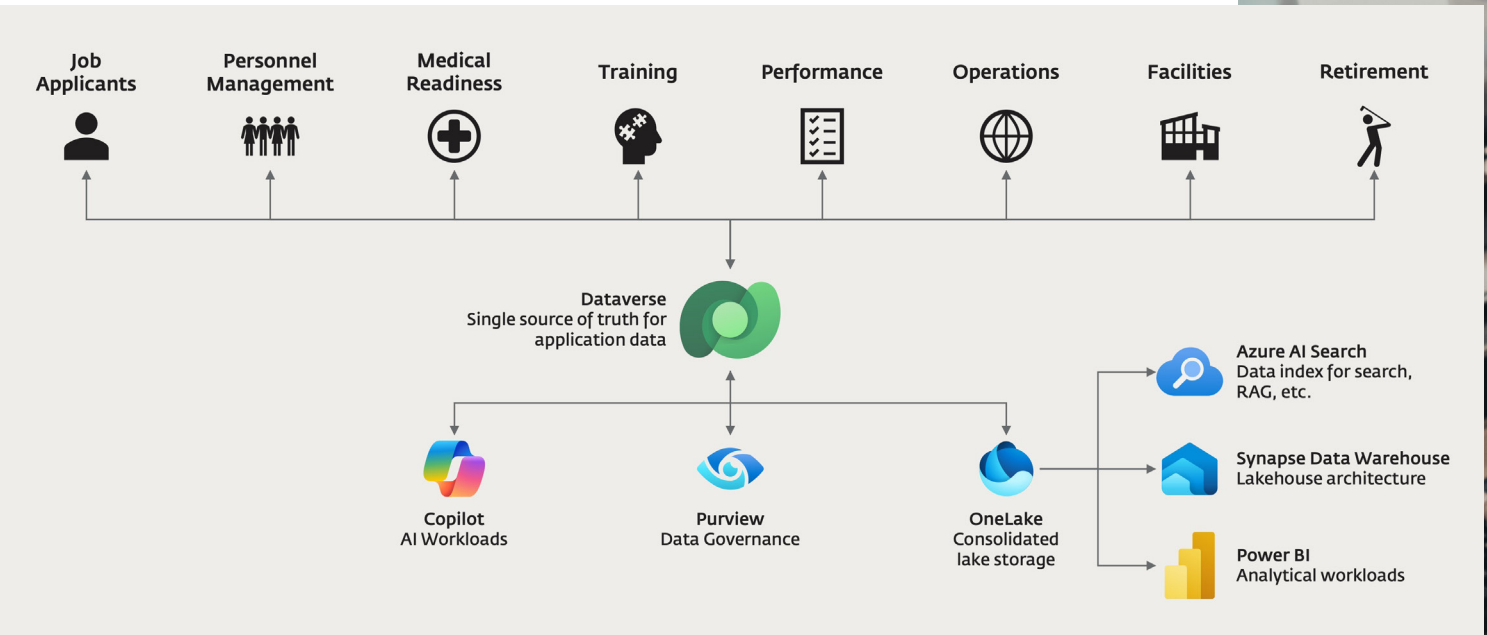


This copying of data—scratch that, this making copies of copies of data—often results in a catastrophic proliferation of (among other things) personally identifiable information (PII). Indeed, our scenario below has resulted in 8x copies of the same PII.



An Ecosystem-Oriented Approach

A more ecosystem-oriented approach is shown below, wherein we've adopted Dataverse as the single source of truth for transactional application data, using it as a data orchestration layer that makes data securely available for use in Copilot, data governance via Microsoft Purview, and virtualized storage in OneLake where the organization's data can then be used in onward scenarios such as indexing for enterprise search or RAG, warehousing, and analytical workloads that will perform far better than Power BI hitting the application data store directly.



I'll add a note here that traditional architecture might be tempted to argue that "ERP can do all of this, so why not use SAP shown in our first diagram across more workloads?"

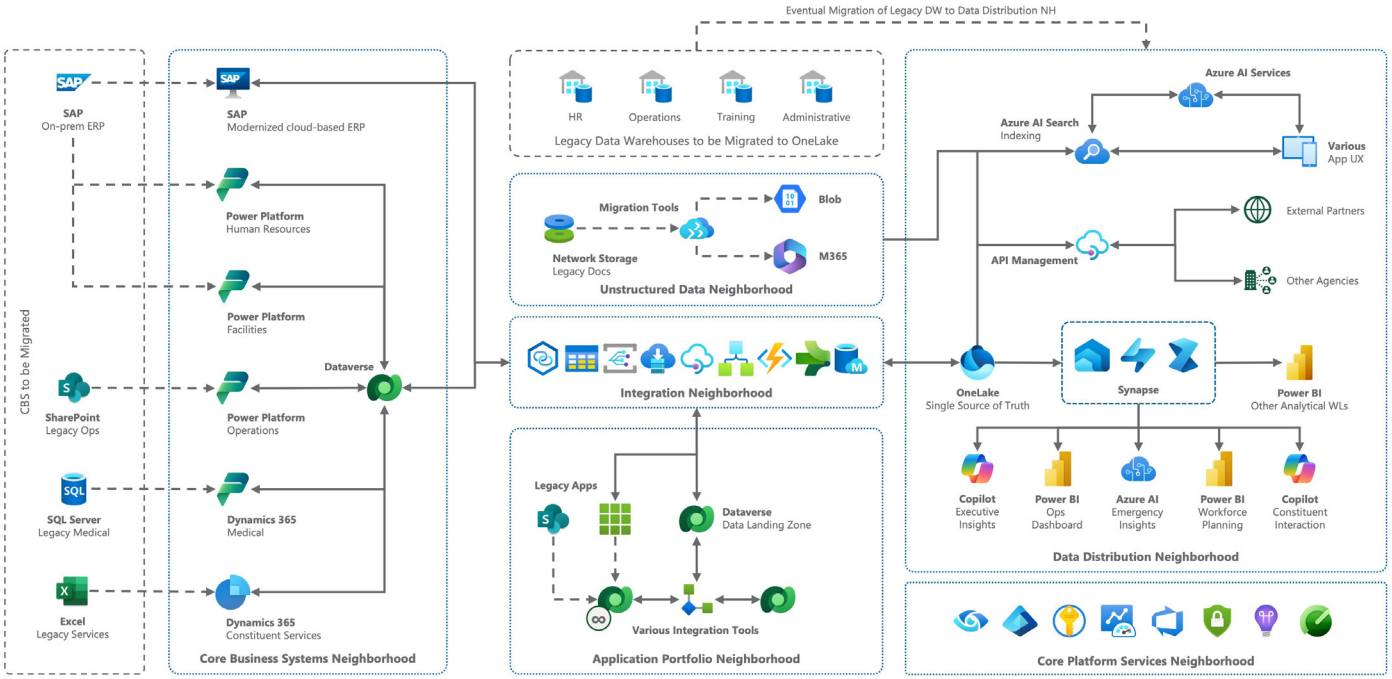
This argument is seductive, but wrong.

First, many organizations have become trapped by overengineered ERP solutions. They've so extensively customized their ERP over time that replacing it proves to be monumentally expensive down the line.. Second, this monolithic app approach is antithetical to ecosystem-oriented architecture because it fundamentally ignores the "Principle of Composability": The whole ecosystem is greater than the sum of its parts. Modularize workloads to be added or removed with little impact to the whole. Rather, our cloud ecosystem must:

- **Disaggregate these functions,** many of which would be considered "core business systems" in their respective organizations, from one another so that significant change in one workload does not have a domino effect in the others;
- **Provide operational insight,** analytics, and reporting to decision makers at a cadence that is appropriate to the task, i.e., reporting on training completion can probably wait an hour whereas knowing who is coming and going from your campus may require a real-time solution;
- **Share data with one another,** ideally from mastered data sources, so as to reduce copying data (or copying copies of data) from one point solution to the other; remember, data is the lifeblood of our cloud ecosystem;
- **Extract value from data,** the organization's collective knowledge, in the form of AI-infused workloads or predictive analytics, e.g., "where shall we stage supplies for the upcoming flood season, based on historical flood patterns and our own supply chain?"

Reference Ecosystem for the Public Sector

Think back to the Reference Ecosystem shared earlier as we combine this with the workload map of our notional public sector organization discussed above. We'll link them in the ecosystem map below that I've built around these common public sector scenarios.



Click to view full page - Appendix 2

This ecosystem map, like the Reference Ecosystem, is a composite of many organizations. Also, remember the Principle of Artistry, and keep in mind that I could have taken this in any number of other directions. Let's hit the highlights (moving generally left to right on the map) with those caveats in mind.

- We're migrating several legacy core business systems into a modern ecosystem-oriented architecture. Legacy ERP has been disaggregated back down to its financial management core, with capabilities for HR and facilities moved out of ERP and into their own composable solutions. Note that we've also moved sensitive operational data out of SharePoint—making that data far more secure—retired on-premise SQL Server that would have been difficult to weave into our modern data platform, and have moved constitution services out of Excel, which would have been difficult to manage at scale.

- We've not eliminated SAP entirely, but we have moved the functions that were logical to keep in SAP into a modern cloud-based solution. Otherwise, we have standardized on Power Platform for the development of many core business systems. Our key thinking here is that getting data out of its legacy stores and into Dataverse allows us to more easily integrate that data to our Data Distribution Neighborhood and its onwards distributive workloads (more on that later). Finally, we opted to move constituent services into Dynamics 365 because (in the case of this organization) we felt that D365's out of the box features were a good fit for these specific constituent services needs. We may have gone in a different direction with a different organization; this ecosystem map is illustrative, not authoritative. **The key takeaway here is that we are leveraging Dataverse as the data orchestration and storage service for our core business systems wherever feasible.**

- This organization had previously allowed data warehouses to sprout up in a rather haphazard way, resulting in four main warehouses each using a different cocktail of technologies. They were hydrated by legacy core business systems that will eventually be sunset, so we will similarly direct the modernized core business systems not to the legacy data warehouses, but to OneLake (more on this in a moment), thus fully integrating their data into the Data Distribution Neighborhood.

- Unstructured data is migrated from legacy network file storage—which is still shockingly common—into a combination of Microsoft 365 (SharePoint and OneDrive) and Azure Blob Storage, which will in turn be indexed and made available for onwards distribution.

- Our Tier 2 ("important") and Tier 3 ("productivity") apps reside within the Application Portfolio Neighborhood, where we will standardize to the greatest extent possible with Dataverse as our primary transactional data service, though expect many personal productivity apps to store their data in a combination of SharePoint and Excel. That's just fine as long as we are rigorously governing that neighborhood to prevent data leakage. There's no need to migrate every legacy app in our portfolio, rather, we can integrate them via the Integration Neighborhood as required, modernizing them when the need (and budget) arises.

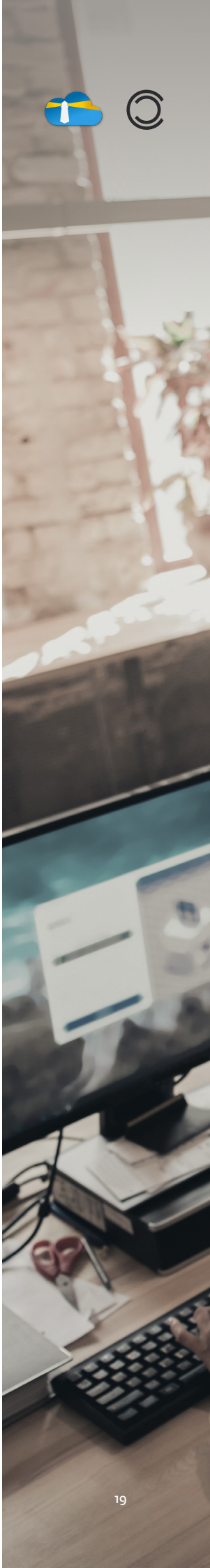
- Data from our core business systems and application portfolio alike is integrated by the services found in our Integration Neighborhood. The specific, neighborhood-level architecture here will be different in every organization, but I've left it alone in this ecosystem map.

- Finally, we arrive in the Data Distribution Neighborhood. There's a lot going on here. Most of our data will eventually flow downstream to land in OneLake. This technology is part of Microsoft Fabric and is built atop Azure Data Lake Storage (ADLS) Gen 2. It's a fantastic technology that also offers a good example of how we can apply the Principle of Following the Money. Microsoft is investing heavily here. OneLake has a 1:1 relationship with the Azure tenant, in other words, there is one OneLake per tenant. There are not zero. There are not two. One. The idea is that organizations manage and govern their lake data here using a strong security model, rather than copying data from data lake to data lake, a long-standing process that has tended to create copies of copies of data (and so on). From there, data will be distributed to "data products" that the organization has decided to build, among them:

1. *Indexing the data estate to support both search of information across the organization and to use in more advanced AI scenarios such as augmenting a LLM (large language model) via a RAG (retrieval augmented generation) pattern, which is a story for another time;*

2. *Distribution to external partners and other agencies via API;*

3. *Consumption by analytical and AI-infused workloads, of which I've offered several examples. These include a Copilot that provides insights and talking points for executive leadership, an operational dashboard through which the agency acquires insights as to the performance of its ongoing operational missions, a "born in AI" workload providing predictive insights to colleagues planning for emergency response contingencies, a workforce planning dashboard used by HR, and a ChatGPT-like chatbot that can be embedded on the agency's website to provide a more rich experience for constituents interacting with the organization.*



- Lastly, our Core Platform Services Neighborhood remains unchanged from the Reference Ecosystem, though I do want to call out the importance of Microsoft Purview in getting a handle on data governance across the ecosystem. This is an essential piece of technology for a future-ready cloud ecosystem.

But why Dataverse as the data orchestration and storage service for our core business systems?

The answer is in the word orchestration, for Dataverse isn't a database in the traditional sense, per say. Rather, Dataverse is a technical service that quite smartly orchestrates much of the storage, logic, integration, security, and other needs for the applications built atop it. In particular, Dataverse is the bridge between an app and the downstream data distribution of the app's data. It's difficult to show in a diagram, but implicit to our Integration Neighborhood is the use of what Microsoft calls a "shortcut" to surface data from Dataverse into OneLake. Think of it like a shortcut to a file in OneDrive or on your desktop. We're not outright copying data into OneLake, which has historically contributed to the copies-of-copies-of-data problem, but rather virtualizing and caching that data in the lake such that we can perform lake-like operations on it without actually duplicating it from its source. This capability to easily use transactional application data from source within the data platform is both critical to the modern cloud ecosystem and unparalleled in alternative technologies.

We'll explore this more in the next section tackling EOA and artificial intelligence.

Let's finally assess this architecture through the lenses offered by our principles of ecosystem-oriented architecture.

1. Principle of Platform First

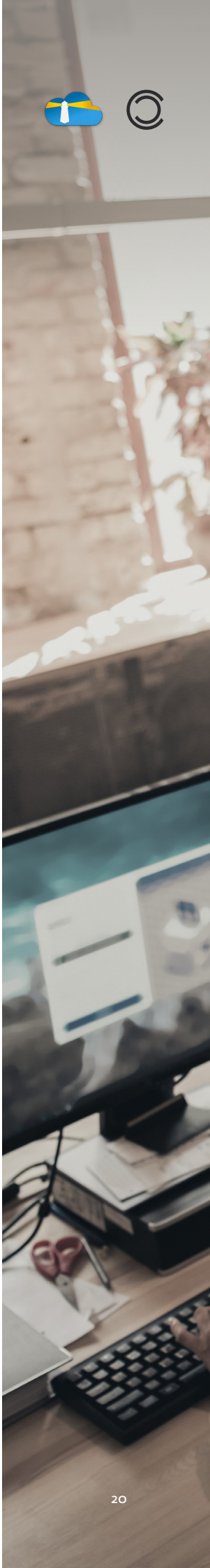
This architecture employs platform-level technologies at every turn, in our Core Platform Services Neighborhood and beyond. We've relied on the Data Distribution Neighborhood to consolidate data, in turn allowing us to retire legacy data warehouses. We've integrated data using the repeatable, reusable patterns and components in the Integration Neighborhood, and have standardized on key technologies like Dataverse throughout the architecture.

2. Principle of Composability

Disaggregating capabilities and data away from monolithic ERP means that workloads can be customized, developed anew, or even removed from the ecosystem all together with minimal impact on the whole.

3. Principle of Evolution

You can't see it in the diagram, but the transition to this EOA need not occur all at once. We could even leave our legacy SAP and data warehouses in place for some time, preferring instead to focus on building a minimum viable Data Distribution Neighborhood into which we gradually—evolutionarily—pipe data from our newly modernized operations, medical, constituent services, and application portfolio workloads.



4. Principle of Restraint

There's a lot that we could have done here, but did not. We could have developed an entirely custom ERP, but modern SAP in the cloud was just fine for our financial workloads. We could have heavily customized a Dynamics application to meet the needs of our operational workloads, but we concluded that a custom app was better. The inverse applies to our employ of Dynamics 365 for the constituent services workloads. We could have gone real-time on the analytical workloads, but instead have blended several of the Synapse technologies to meet a range of insight needs. We made these choices by practicing the...

5. Principle of Artistry

...which suggests that, though the ecosystem map we've drawn shares many elements in common with the Reference Ecosystem, it is the product of the artistry in combining technologies to suit the organization's needs.

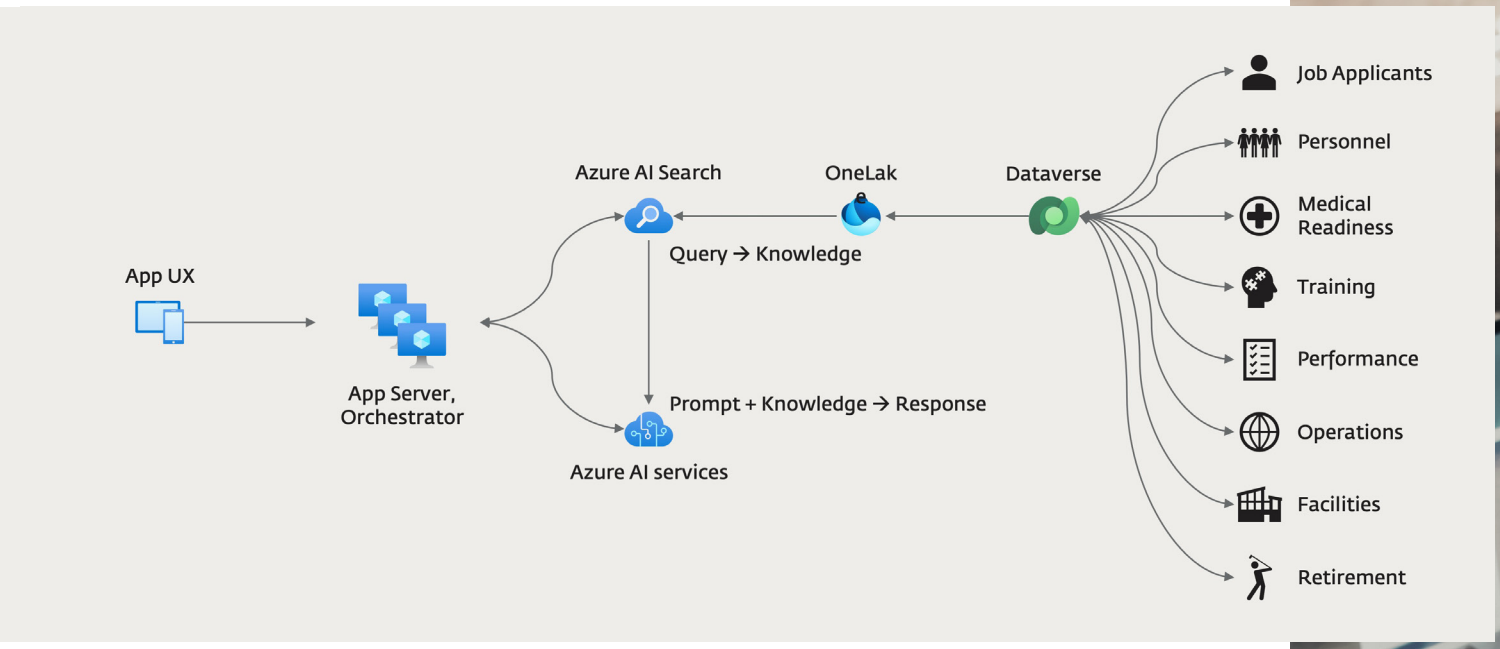
6. Principle of Following the Money

The cloud ecosystem shown in this map relies heavily on some of Microsoft's biggest investments including a Data Distribution Neighborhood built atop Fabric technologies, the widespread use of Dataverse, and the use of Purview to govern data across the ecosystem.

A Future-Ready Public Sector

We'll conclude by exploring how EOA equips organizations to leverage artificial intelligence, in particular, AI acting on an agency's own data.

The diagram below reshuffles the various workloads from our earlier example. On the right we have the collection of Tier 1 "core business systems" discussed at length above, all atop Dataverse as their application data service.



Dataverse then shortcuts that data into OneLake.

Meanwhile, on the far left we see an application user experience (UX)—e.g., a mobile, tablet, or web app, built with Power Apps, in this case—that provides an end user the ability to interact with our AI workload.

The application sitting beneath the UX queries the knowledge contained in Azure AI Search's index (as derived from the data sources on the right). It then passes that prompt and knowledge to Azure AI services to generate an appropriate response to be fed back to the user.

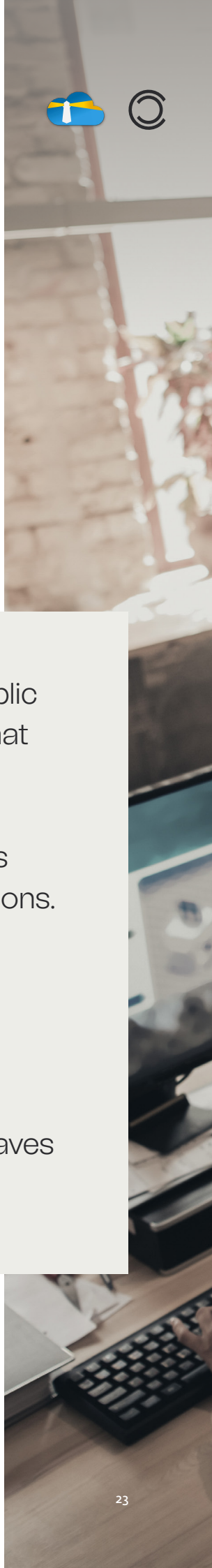
This is what is called a RAG pattern, which stands for "retrieval augmented generation". The precise methods of causing AI to act upon an organization's data are both varied and constantly evolving, but whether RAG or some other technique, similar principles apply: Consolidate the organization's data into storage technologies that are accessible to AI, index that data, and pass that data to AI services that are able to reason over that data and generate a response.

If you're interested in further exploring the topic of organizational maturity for artificial intelligence, I encourage you to read my September 2024 white paper, Crafting your Future-Ready Enterprise AI Strategy, Second Edition.

The key is that ecosystem-oriented architecture creates the conditions for these types of patterns to succeed, and has made the agency future-ready in its flexibility.

Organizations across the economy and around the world have spent decades kicking the proverbial data can down the road, opportunistically storing data in whichever service was cheapest—or simply present—implementing workloads that are disconnected or integrated as point-to-point rather than via platform-level integration services, and paying little to no mind of data governance.

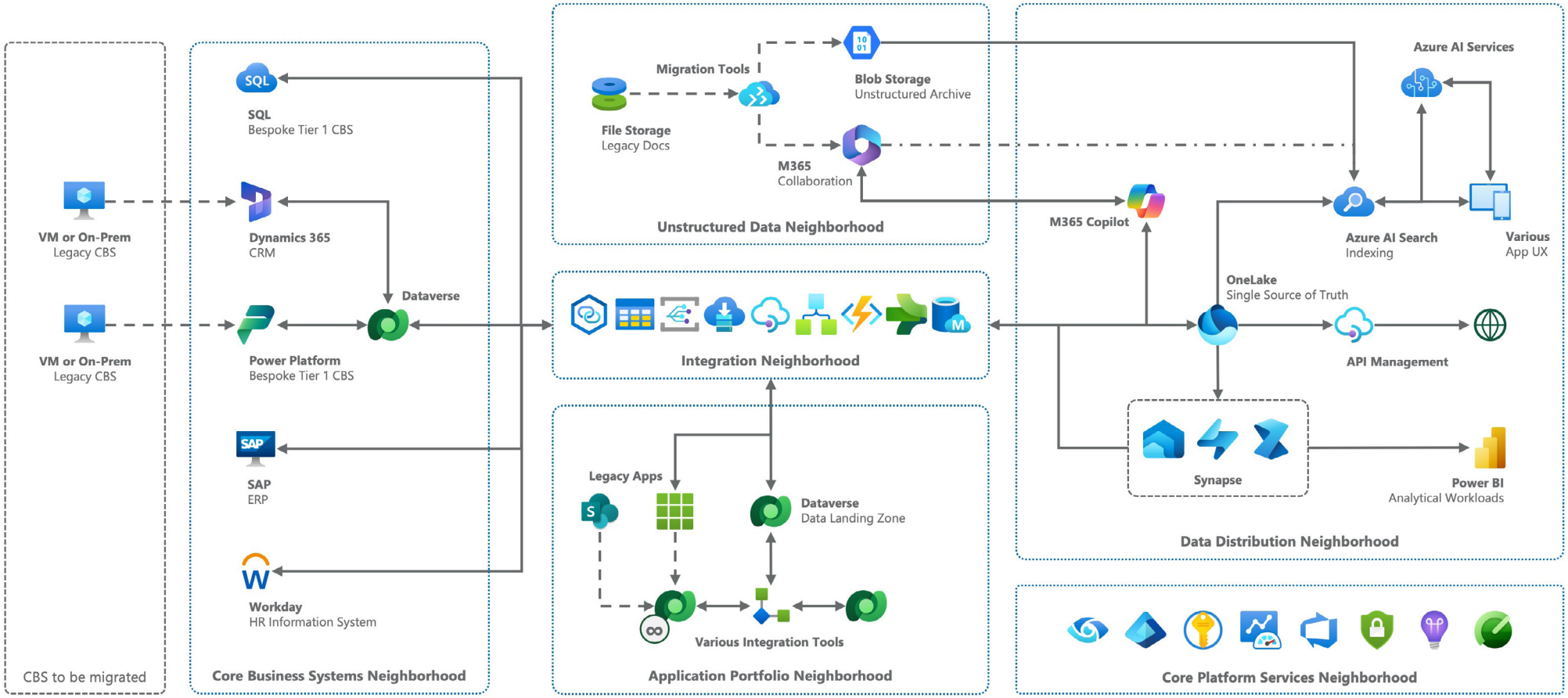
This white paper could have easily been four times longer, the basis for a semester-long class, at minimum. So, I'll leave you with this.



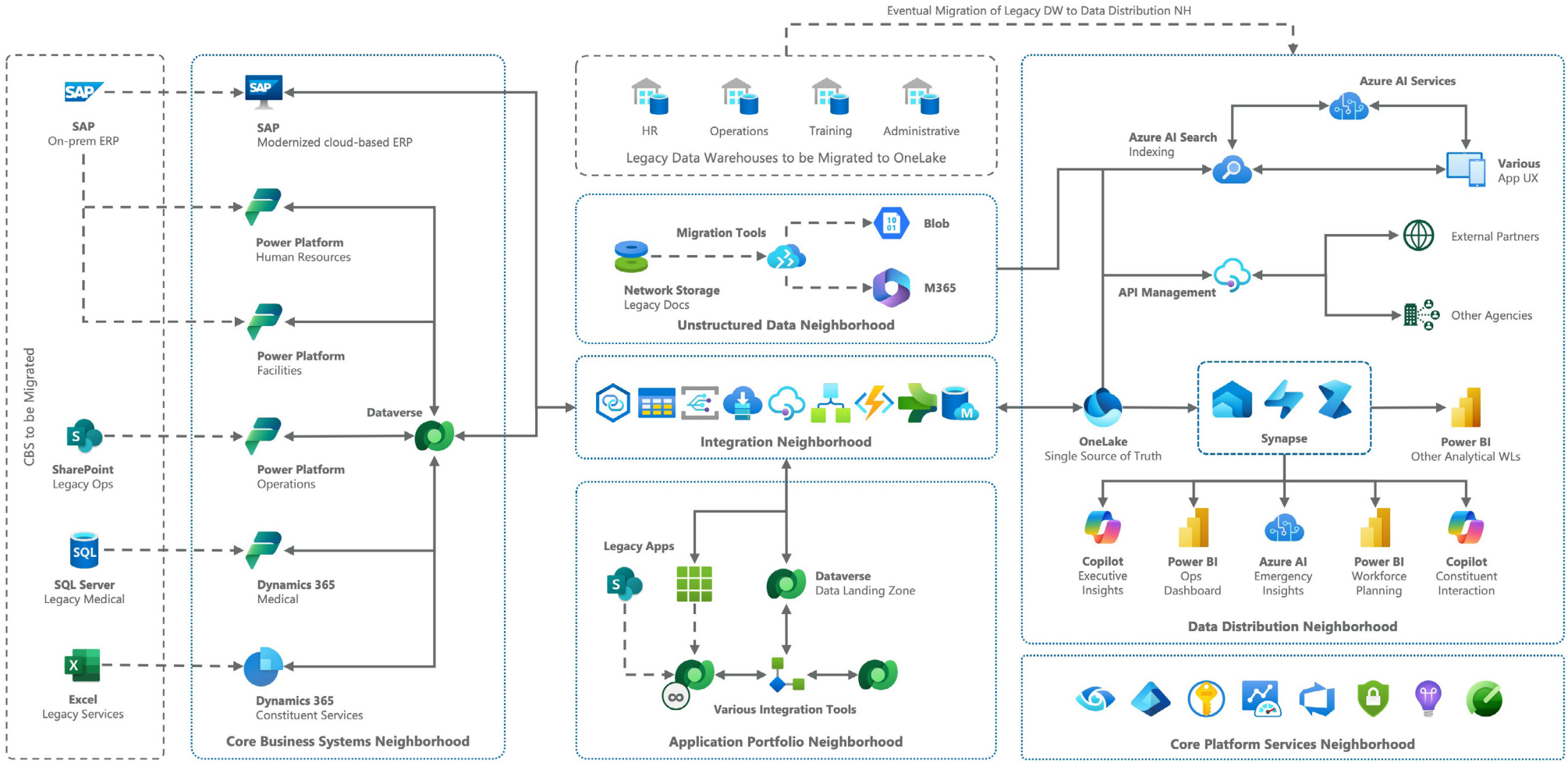
The transition to EOA feels difficult at first, particularly in public sector organizations with lengthy budget planning cycles that lack a tradition of viewing technology leaders as strategic leaders. But as wave periods between major innovation become shorter, so too does the need for IT leaders across government to become strategic leaders of their organizations. After all...

Wave periods between major innovation in the cloud are growing shorter. We no longer have the luxury of waiting it out, of adopting later. Cloud ecosystems built on strategic foundations create the conditions to absorb successive waves of change.

Appendix 1: Reference Ecosystem Map



Appendix 2: Reference Ecosystem for the Public Sector Map



Want to find out more about our solutions and the impact we can deliver for your business?

Contact us today

www.codecsystems.com