From Data Processing to Data Primacy : The Agentic inversion of Work and IT Architecture

Andy Brown — Everpure Board Member, Chair of the Compensation and Risk Committees; CEO, Sand Hill East
Charles Giancarlo — Chairman and CEO, Everpure

How Sixty Years of Enterprise Computing Came Full Circle—and What It Means for Enterprise Architecture and Designing Work

***"95%** of enterprise generative AI pilot projects fail to deliver measurable financial returns or a positive impact on the profit and loss (P&L) statement"...MIT*

Niccolò Machiavelli : "There is nothing more difficult to take in hand, more perilous to conduct, or more uncertain in its success, than to take the lead in the introduction of a new order of things." from The Prince

Abstract

The vocabulary of enterprise computing is not incidental. Each era's dominant term—"data processing," "information technology," "information security"—reveals what organizations believed the machine was *for*. This paper traces that terminological evolution and argues that we are witnessing a structural cycle: computing began as a capability to process data, spent four decades organizing itself around applications that held data captive, and is now—under the gravitational pull of AI—turning to organizing around data itself.

The paper extends the same argument to work. Just as data is being liberated from applications, work is being liberated from roles. When agents and humans are designed into processes side by side—"human as a service"—process architecture can finally be specified in terms of inputs and outputs rather than in terms of which human, in which job, performs which step.

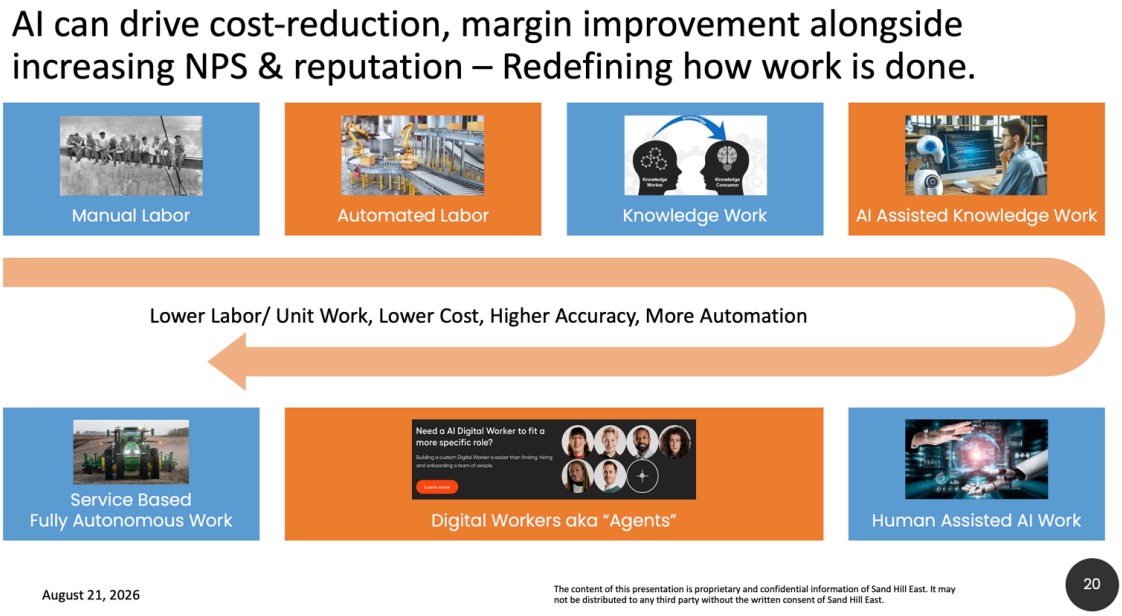
We tell both stories on one timeline because they were never two stories. In every era, the way an organization held its data determined the way it could organize its work, and the way it organized work determined what it did with its data. The cobbler, the clerk, the EDP department, the ERP user, and the agent are all points on the same line.

Just as Data Primacy has (re)emerged alongside process design as the core "hows," the move away from role-based to skill-based itself is the enabler for humans and agents to inhabit a hybrid future where, as agents and model capabilities continue to improve, how work is done can continue to evolve alongside.

Getting to Boston by 3pm

Thirty years ago, planning a trip meant gathering maps, schedules, reservations, and recommendations then stitching it all together yourself. You were the information system. Applications then digitized each piece of the journey — the airline, the hotel, the map, the expense report — but you still moved between them, carrying context from one to the next. You became the integration layer.

Now imagine simply saying: *"Get me to Boston by 3 p.m., under $1,000, and ask me before making any expensive changes."* AI can work across your calendar, policies, preferences, schedules, prices and weather, regardless of where that data lives, and orchestrate the applications, agents, and humans needed to deliver the outcome. You did not name an application. You did not name a role. You named a destination, a budget and a control.



Enterprise computing has followed a remarkably similar journey: humans organized information, applications organized data and work, and now AI is allowing us to organize around data and outcomes. Even the language we have used along the way reveals what we believed the machine was for.

One Timeline, Two Questions

Any era of enterprise organization can be interrogated with two questions:

Where does the data live, and who owns it?

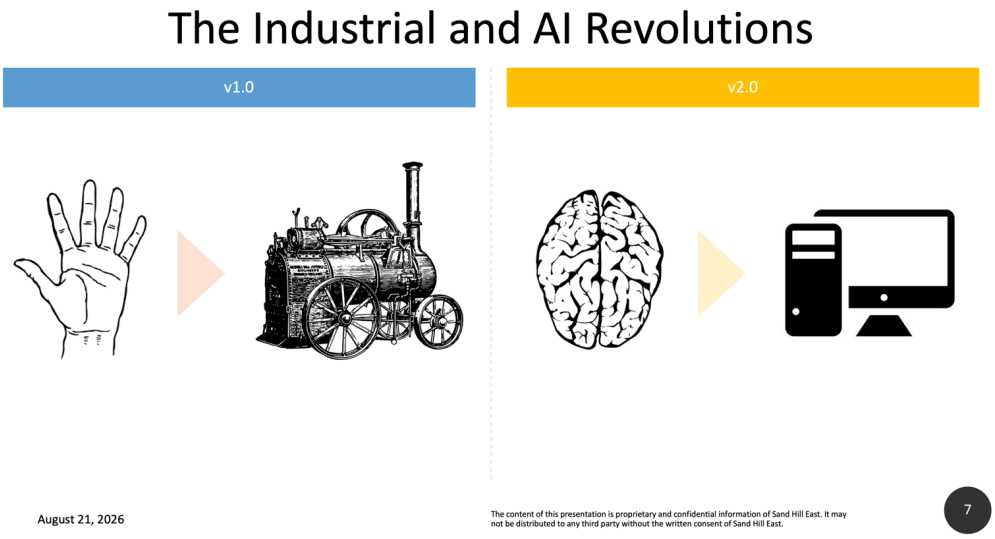
What is the unit of work, and who performs it?

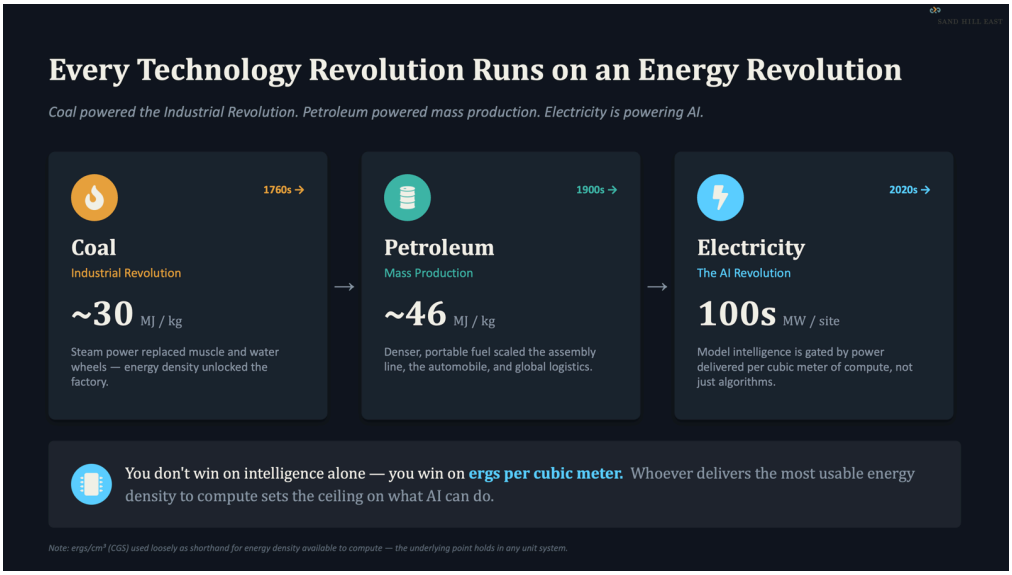
The answers have always been coupled, and they have moved together. When data lived in a craftsman's head, the unit of work was the finished shoe. When data lived in an application's schema, the unit of work was a role that operated that application. That coupling is why the two inversions now under

way—data escaping applications, and work escaping roles—are not coincidental.

They are the same event, observed from two directions.

The steady march of technological progress is fueled by increasing the comfort and wealth of humans. To this end, technological progress has focused on efficiency of labor and the leveraging of science and engineering. Since the Industrial Revolution, it has focused on using external energy to substitute for human or animal labor and now human activity. Thus, each of these eras had to progress in one or more of these areas: time savings, labor savings, coordination improvements, or substituting external energy for human energy. This slide, which we also use a lot, explains how this worked in the Industrial Revolution and now in the AI revolution, with manufacturing automation post WW2 in the middle.





So, back to the plot: let's explore the evolution …

Era	Approx. dates	Where data lived	What the unit of work was	How it was secured
Craft, Ledger and Letter	Pre-1770	In the practitioner and the ledger	The finished outcome. Specific to Skill or Craft (e.g. Shoemaker)	Physical Security (who knew the information)
Industrial	1770–1910	In the form, the order, and the wire	The specialized task	Physical Security and Transport Security (including cyphers in letters during wars).
Corporation and Data Processing	1910–1970	In the memo, the file room, and the tape	The role	Physical Security (who had access to the computer)
Information	1970–1990	In the database and the terminal	The role, now systematized	Network Security (who could digitally access the information)
Application	1990–2010	Inside the application that produced it	The application owns the process definition, defines multiple roles, activities, and tasks managed within the platform	App/Network Security (Role based access controls and firewalls)
Analytics, SaaS and the Citizen App	2010–2025	In many places, none authoritative	Applications delivered as a service mostly from cloud or stand-alone	Cloud & App/Network Security (Role based access controls and firewalls). Zero Trust

Era	Approx. dates	Where data lived	What the unit of work was	How it was secured
			cloud. Lock-in on data even more rigid.	
Agentic	2025–Now	In governed sources of truth	Roles, process, activity, and task are the inputs; skills are the common language shared between humans and agents. This helps process designers allocate and orchestrate work	Data Security and Zero Trust Workflow Security

In each era, the human has become less specialized to an individual outcome. Think about the role of a shoemaker's apprentice in the 1800s versus the courses studied for an undergraduate degree in the liberal arts today. The apprentice, if you use Malcolm Gladwell's book ***Outliers*** as a basis, would require a minimum of 10,000 hours of training to become a shoemaker, whereas skills today even in AI code generation, can be learned online in a few short weeks.

The rest of this paper walks the timeline, and then outlines what needs to be done for hybrid work to be successfully performed with humans and agents working together.

Craft, Ledger and Letter (pre-1770)

Work was the unit. Before industrial scale, most work was organized around an outcome. The cobbler made the shoe. The merchant completed the transaction. The clerk maintained the ledger. There was specialization, of course, but the abstraction we now call a *job role* was not yet the dominant machine for decomposing enterprise work. The worker was close to the finished output and usually accountable for all of it.

Data in this era had no independent existence. It lived in the practitioner's memory, in the ledger under the counter, and in the letter carried by hand. There was exactly one copy; it was owned by the person doing the work, and no one asked which system of record was authoritative because there was only one, and you could see it from where you stood. Reconciliation was not an industry. It was a conversation.

The limit was scale. An organization could only be as large as the number of outcomes one accountable person could hold in view.

The Industrial Era (1770–1910)

Industrialization turned work into roles. Adam Smith's division of labor, then Taylor, then Ford: break an outcome into repeatable steps, specialize humans around those steps, and manage the interfaces. The breakthrough was not mechanization alone. It was that the human became a component in a process architecture. Once you do that, the "job" becomes a container for a bundle of repeatable tasks, and the enterprise begins to be designed around containers rather than around outcomes.

The moment you decompose an outcome, you create a new problem: the pieces have to be coordinated, and coordination runs on data. This is the era's undertold story, and Stephen Ambrose tells one version of it well in *Nothing Like It in the World*. The transcontinental railroad was not primarily an engineering achievement in the sense we usually mean. It was the first body of work too large for any one person to hold in their head — thousands of workers, two companies advancing from opposite coasts, supply chains that crossed a continent. What it produced, alongside the track, was the modern company: divisional structure, written operating rules, standardized reporting, and the telegraph used as a management instrument rather than a novelty. Standard time itself exists because railroads needed a shared clock to make their data mean the same thing in two places.

So the first great expansion of organizational scale was data-centric. The form, the order, the manifest, and the letter, wire or post became the interface between specialized humans. Data stopped being memory and became infrastructure — the connective tissue that let decomposed work be reassembled into an outcome. Here, the data was still treated as “exhaust”: once exchanged, it was history, and generally did not persist once the message was passed.

The Corporation and Data Processing Era (1910–1970)

The corporation then hardened roles into hierarchy. Once work is divided among specialized humans, you need coordination, supervision, information aggregation, exception handling and approvals. Middle management is not an accidental corporate phenomenon; it is largely the coordination cost created by the decomposition of work. This was originally created and perfected by the creation of large armies and empires, as was perfected by the Romans.

Before enterprise software, the corporation's information system was made of people. The memo — typed, carbon-copied, physically routed — was the packet protocol of the organization. Typing pools were serialization layers. Filing clerks were storage and retrieval. Middle management existed in large part as a *human message-passing and aggregation network*: collecting information from below, summarizing upward, disseminating decisions downward. The hierarchy was not merely a control structure; it was an information architecture. (The "cc:" field in every email today is a fossil of the carbon copy — the memo era's vocabulary outlived its medium.) Persisting the data allowed the hierarchy to operate and scale.

Into this world, the first commercial computers arrived under an honest name: **electronic data processing**. The EDP department, typically buried inside the finance function, existed to do exactly what the label said — ingest structured records (payroll, ledgers, inventory) and process them in batch. The data was the point, at both the input and the output. The machine was a faster tabulator, a successor to the punch-card operation, and the organizational unit was named for the activity performed on data, not for any technology or application.

Note what that name concedes. In 1965, data was primary and processing was the service. The computer was hired to do a job to the data, the way the typing pool was hired to do a job to the memo. Nobody would have said the machine *owned* the payroll file.

Security in this era was correspondingly literal: it was called **computer security**, and it largely meant physical control. Lock the machine room, control who could submit a job, protect the tapes. The asset being protected was the machine and its media because the data had nowhere else to live and could not be accessed by any means other than physical ones.

The Information Era (1970–1990)

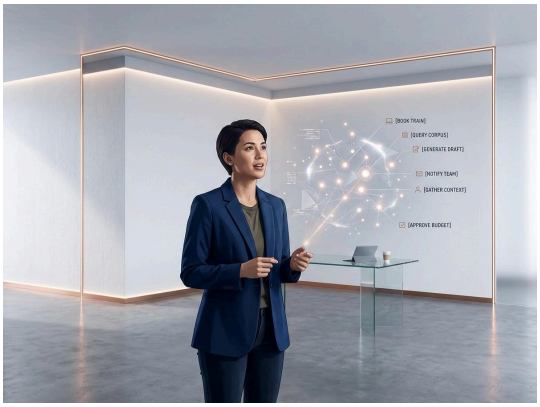
As terminals, databases, and networks arrived, the vocabulary shifted. "Data processing" gave way to **management information systems** and then to **information technology** — a term coined as early as 1958 in the *Harvard Business Review* but became dominant only by the late 1980s. The shift from "data" to "information" was aspirational: raw data, processed and contextualized, would inform decisions. The shift from "processing" to "technology" was structural: the function was no longer defined by an activity but by an ever-expanding portfolio of systems, networks and,critically, **applications**.

Work changed in parallel, though more quietly. The role survived intact; what changed was that the role acquired a system. The clerk became the user of a system. The supervisor became the approver in a system. The hierarchy that had been an information architecture began to be re-implemented as one, which is a subtle but consequential difference: once the hierarchy is encoded, it is much harder to change than when it was made of people and paper.

Security followed the same arc. "Computer security" became **information security**: the recognition that the asset was the information itself, in whatever form it took — on disk, in transit, on paper, in someone's head. The CIA triad (confidentiality, integrity, availability) formalized this abstraction. Later, as networks became the dominant attack surface, the term mutated again into **cybersecurity** — a telling shift because it named the *domain of conflict* rather than the asset. We stopped naming the thing we protect and started naming the battlefield.

"In the beginning, work was the unit. The craft was the process."

"Then, we built roles and apps. We trapped data inside silos."



"The Inversion: Work escapes roles. Data escapes apps."

"Data Primacy. The destination is the goal; the executor is fungible."

The Application Era (1990–2010)

The IT era's defining and largely unquestioned architectural assumption was that **applications owned data**. ERP, CRM, HRIS, core banking, trading systems — each application defined its own schema, optimized for its own workflows, and treated data as its own bespoke repository rather than as an enterprise resource. The application collected data from the human teams or machines that interacted with it and then organized that data in the interest of the application architecture itself, at the whim of the application's developers. The phrase "system of record" captured the power relationship precisely: the application was the system; the data was merely the record it kept.

It is worth naming what an application actually is, because the term has been doing too much work for too long. **An application is a group of processes defined by workflows** — a set of steps, sequences, controls and states that someone once decided to encode. The schema is downstream of the workflow. Which means that when an application owns your data, what really owns your data is a set of process decisions made by a software vendor, for a generic customer, at some point in the past.

The consequences defined thirty years of enterprise architecture and expansion. Each enterprise application defined and contained its own data and its own version of truth. Data silos proliferated. An entire middleware industry — ETL, EAI, enterprise service buses and eventually API management — arose solely to move and rationalize data between the applications that imprisoned it. Master data management existed because no one could agree which application's copy of the customer was true. In most large enterprises today there is no single source of truth and, simultaneously, many sources of truth that do not agree — producing enormous volumes of reconciliation, data marshalling, stewardship and lineage tracing. Data warehouses were built as archaeological digs: extract the data from the applications after the fact, reconstruct what happened, and report on it.

Data was the exhaust. Applications were the engine.

Humans, in turn, were organized around applications. Enterprise software encoded the role model rather than replacing it. The application era changed the human role from *processor of information* to *operator of systems*. Job descriptions specified which systems a person operated. Work was decomposed into roles, roles were mapped to applications, and process design became an exercise in choreographing which role performs which step in which system. "Swivel-chair integration" — a human re-keying data from one application into another — became the connective tissue of the enterprise. RACI matrices, BPM notation and workflow tools all share one assumption so deep it was never stated: **the executing unit of work is a human occupying a role.** The org chart and the application portfolio co-evolved until they mirrored each other. For example, the same exact finance platform implementation on Oracle vs SAP would have had different job titles like SAP Finance specialist or Oracle Finance specialist.

The Analytics, SaaS and Citizen-App Era (2010–2025)

Three things happened at once. First, data volume and variety outgrew the relational assumption: Hadoop, then object stores, then lakehouses and in-memory engines, absorbed structured and unstructured data at a scale the application era never contemplated. Second, Software as a Service let individual departments choose their own applications for their own workflows — a genuine gain in speed with a side effect that took a decade to bite: it fragmented corporate data, and often made it very difficult to use that data as part of the enterprise data infrastructure at all. Third, low-code tools and, latterly, AI-assisted development let small groups and even individuals create their own applications, cementing non-enterprise data structures into place with no architectural review at all.

The result is data and data-structure arbitrage across departments, languages and geographies. It is worth being precise about why this is corrosive. The problem is not that there is no source of truth. The problem is the *proliferation of local data*, which fragments the context of the data and creates conflicting sources of truth — which is as bad as, or worse than, having none. A single wrong number can be corrected. A dozen defensible numbers, each with a plausible owner and lineage, cannot be.

It operates as if every department kept its own address book. Each may contain a different version of the same customer, and there is no authoritative way to reconcile which version is current and correct. Everyone is confident. Nobody is right. At a company like Merrill Lynch, one of the earliest scale adopters of SalesForce, the ownership of customer data was itself a negotiation tactic in Wealth Management practices recruitment into large wire houses. When the data was trapped in ACT and Goldmine it was an asset owned by the FA Practice; as soon as it was in SalesForce it became a corporate data asset.

Work fragmented the same way, and knowledge work made the fragmentation less visible rather than less real. The assembly line is easy to see because the workpiece physically moves from station to station. In the modern corporation the "workpiece" is a contract, a customer issue, a forecast, a purchase order, a configuration, a hiring decision. It moves through email, meetings, spreadsheets, SaaS applications and approval chains. Because we cannot see the line, we stopped measuring the work piece and started measuring the stations — headcount, tickets, utilization, story points. We managed the containers and lost sight of the work.

By 2025 it was already clear, ahead of AI, that the existing IT edifice was inefficient and that a new model was required.

The Agentic Era (2025–present)

Agents break the role abstraction. This is the inversion. An agent does not need a "job." It needs a task, context, permissions, an input and a required output. Once you introduce that executor into the enterprise, it exposes something uncomfortable: many human jobs were themselves just bundles of heterogeneous tasks, held together for no better reason than that one person had to perform them.

The way work has been organized has broken. Capable agents can now execute cognitive tasks — drafting, reconciling, configuring, validating, triaging — so process design confronts, for the first time, a genuine choice of executor. The honest way to design under that condition is to put agents and humans on the same plane and ask of each step only: *what does it consume, what must it produce, and to what standard?*

We are, predictably, measuring the wrong thing in the meantime. Today people are counting tokens. A token is not a unit of work; neither is a task. Both sit below the unit of work, the way keystrokes sat below the memo. We expect the pendulum to swing back toward measuring and organizing actual work — the completed unit, the outcome — which is where this timeline started, in the cobbler's shop, before scale forced us to substitute the role for the result.

Three arcs, one shape:

work/outcome → roles/apps → work/outcome

data → applications → data

physical data security -> app/network security -> zero trust & data security that travels with the data

The AI trend has forced us to re-examine technology yet again around organizing principles. For data, we are entering a Data Primacy era. For work, we expect conversations about the *completed unit of work* to displace conversations about jobs and roles — which will change the fundamental organizational construct of the enterprise.

The Reversal: Why Data Wins This Time

AI inverts the ownership relationship. A foundation model is, in a meaningful sense, *compressed data* — its capabilities are a function of what it was trained on and what it can retrieve. In an AI-native architecture, the durable, differentiating asset is the corpus: the enterprise's documents, transactions, telemetry, conversations and institutional knowledge. The application layer, by contrast, is becoming thin, generative and increasingly disposable — an interface synthesized on demand over data plus models.

If the application era ran on the premise that data was exhaust and applications were the engine, the premise now reverses: **data is the engine, and AI — like applications before it — is the fuel.** Fuel is consumed and replaced. Engines are what you invest in.

The signs of this reversal are already institutionalized. Data lakes evolved into lakehouses; "data mesh" and "data as a product" reframed datasets as first-class deliverables with owners, contracts and SLAs; semantic layers and knowledge graphs emerged to make meaning (data plus context) machine-consumable independent of any application. Vector stores and retrieval pipelines are, in effect,

the new middleware — but where the old middleware moved data *between* applications, the new middleware converts data *into* actionable intelligence.

These are, for now, workarounds outside the core applications. They attempt to solve for the fact that no single application has the answer to most company activities, and the problem worsens with every application instance added — including every new AI agent, each of which creates its own data set and fragments the corporate data asset further.

The alternative is **Data Primacy**: enterprises optimizing their IT around governed sources of truth defined by their own business model, rather than around any individual application. The best analogy is a passport. A passport is one trusted, governed identity that many systems and processes accept, precisely because none of them is allowed to issue its own version. That is what a governed source of truth does for a customer, a product, a position or a policy — it ends the address-book problem by making one record authoritative rather than by making all the copies agree.

The sixty-year arc, then, is a palindrome: data → applications → data. But the second data era differs from the first in one decisive respect. In 1965, data was primary because it was all the machine could handle. In 2026, data is primary because it is all that endures — models improve, applications regenerate, but the corpus compounds. Data governance, lineage, authenticity and accuracy play a decisive role in the trustworthiness of any answer an AI gives from that data.

Governance, and the Verity of Core Data

There is a governance problem hiding inside the good news, and it needs to be said plainly.

SaaS allowed individual departments to choose their own applications for their own workflows, with the side effect of fragmenting corporate data. AI is making this worse by allowing small groups, or even individuals, to create their own applications and their own agents. The easy proliferation of agents by individual employees looks set to exacerbate the SaaS problem rather than solve it: as agents proliferate, so does the number of data sets that contain *some* version of the truth.

Just as there needed to be centralized governance around application architecture, there is an even greater need for centralized governance around AI agent architecture. And the reason is not primarily the proliferation of the agents. It is to maintain the verity of the core data.

This is, oddly, why Wikipedia matters more today than it did before. Not because it is always right, but because it is a single, openly governed, contested-in-the-open system of record that *is* the reference. An enterprise needs the same thing: an authoritative system of record, highly governed, centrally and corporately owned, that does not become fragmented by individual or departmental AI development. Agents and metadata/metamodel aggregation layers must be able to deal with the data ambiguity that already exists while presenting as coherent an enterprise view as possible. That gap — the distance between what the enterprise believes it knows and what its data can actually support — is now

extremely visible to any agent asked to collate data horizontally or vertically. Agents are, among other things, the best data-quality audit an enterprise has ever been subjected to.

Security Follows the Data

Security is being pulled through the same inversion. Network, perimeter and application security assumed you could protect data by protecting the systems around it. When data flows continuously into training pipelines, retrieval systems and agent contexts, the durable control point is the data itself: classification, lineage, entitlements and policy that travel *with* the data — and with the output of the workflow that consumed it, which needs to be governed by entitlements as well. Zero trust anticipated this — trust nothing by network location, verify everything by identity and context — and data-centric security completes it.

It would be comfortable to conclude that AI simply relocates the control point. It does more than that: it changes the adversary's economics. An AI can string together several medium-severity vulnerabilities into a single major one, at a speed and scale no human red team can match — turning findings that were rationally deferred into findings that are not. And models can be attacked through their own interface: carefully constructed prompts can induce a model to divulge exactly what it was engineered not to divulge. Neither of these is a network problem, and neither is solved by protecting the systems around the data.

The next mutation of the vocabulary is already visible: we increasingly speak of **AI security** and **data governance** in the same breath, because governing what the model can see *is* the security model.

What an agent is allowed to know is not what it is allowed to tell you

This deserves to be pulled out on its own, because it is the least-solved problem in the entire architecture.

In an agentic enterprise, three separate questions must be answered for every interaction, and they have historically been treated as one:

- **Who — or which model — is allowed to access which data?**
- **What entitlements travel with that data?**
- **Who is allowed to see the result?**

An agent may be legitimately authorized to access information that the person asking the question is not authorized to see. If a piece of data is off-limits to me, should the model be permitted to use it as an input to logic whose *result* I am allowed to see? The answer, of course, is "it depends" — on whether the result discloses the input, on materiality, on regulation and on jurisdiction. But "it depends" is not an architecture, and today most enterprises have no mechanism to make that determination at all, let alone consistently and auditably.

This is inference leakage, and it is what makes Data Primacy more than a plumbing exercise. Making enterprise data available to AI is the easy half. The consequential half is keeping the data governed *after* it leaves the application boundary and becomes context for machine reasoning. Put simply:

What an agent is allowed to know is not necessarily what it is allowed to tell you.

The New Organizing Concept: Human as a Service

That the assumption has broken is now clear. What replaces it is less obvious, and this is where the deliberately provocative phrase **"human as a service"** earns its keep.

In an agentic workflow, the human is invoked through a defined interface — an approval request, an exception escalation, a judgment call, a client conversation — with a specified input, an expected output and a latency budget. This is not a demotion of the human; it is a *clarification* of the human. It forces the process designer to articulate precisely what the human contributes that the agent cannot: accountability that must rest with a person, judgment under genuine ambiguity, ethical and fiduciary responsibility, relationship and trust, and the authority to accept risk. Everything else is negotiable as to executor.

Or, more compactly: **execution can be fungible. Accountability cannot.**

“Purpose” is human. Goals are delegated.

There is one more difference, and it is the important one. The human has, or defines, a *purpose*. The human then gives the AI a *goal*. AI has only the goals set for it by a human who holds a purpose. Agents may take unexpected and even unwanted actions in pursuit of a goal, but they hold no purpose beyond it.

Today's AI is, in this narrow sense, amoral rather than immoral. It has no intent of its own; it operates toward someone else's goal. Clarity of purpose remains uniquely human, and it is the thing that makes declarative process design possible in the first place — you cannot specify inputs, outputs and standards for a process whose purpose nobody has articulated.

Two practical consequences follow. First, the gap between a stated goal and an unstated purpose is now an attack surface. An agent that derives *derivative* goals in service of the one it was given — spinning up sub-tasks, acquiring data, invoking tools — is behaving exactly as designed and may still end up somewhere no one intended. Zero trust must therefore be applied to the agent itself, not merely to the network it sits on: verify intent continuously, scope authority narrowly, bound the derivative goals an agent may set for itself, and monitor behavior against the purpose rather than against the goal.

Second, and more bluntly: if agents ever begin operating with *purpose* rather than a human-defined goal, we are all in trouble. That is a design line worth naming explicitly while we still have the option of drawing it.

Process design becomes declarative

The deeper consequence is that **process design becomes declarative rather than role-based**. The legacy question — "which role in which department executes this step?" — dissolves into a cleaner specification: define the inputs, define the required outputs and their quality gates, define the controls and audit evidence required, and let an orchestration layer bind each step to an executor (agent, human, or human-verified agent) based on skills, capability, risk, cost and regulatory constraint.

The binding can even be dynamic: routine instances flow straight through agents; anomalous instances invoke human-as-a-service. Work 1.0 designed processes around the org chart. Work 2.0 designs the process first and derives the org — and the agent fleet — from it.

Whether we organize agents using human constructs is an open question worth taking seriously. If agents and humans must collaborate, keeping agents organized around personas is at least a logical starting point, and a good deal of governance and safe-API design will keep humans in the loop for some time. But the persona is scaffolding, not the destination. The destination is Work Primacy: the movement from application and workflow centricity to the completed unit of work as the organizing object.

Two governance corollaries follow, and both are security questions as much as organizational ones. First, **non-human identity becomes a first-class, but untrusted, citizen**: agents need credentials, entitlements, least-privilege scoping and behavioral/intent monitoring even more than employees do, and the identity fabric must treat them as an insider threat; outcome driven, amoral and unpredictable.

Second, **accountability must remain human even where execution is not**: the control framework shifts from supervising *how* work is done to attesting to *what* was produced — output-based controls for output-based process design.

> Employees interact with other employees all the time (both in and out of the office). Humans have a sense, trained by our DNA and years of socialization, of whether another human is to be trusted, and can pick up on these cues generally quickly. Humans have precious few ways to do the same with AI agents, who operate more like fully remote employees who do not interact with their human colleagues unless instructed to do so.
>
> Another way to think of it, is that the Agent needs to be treated as a capable but suspect "contractor". An employee is someone that has been interviewed, vetted, reference checked, tested, and passed behavior checks with countless colleagues. AI agents have none of these things - more like a freshly hired contractor with no references, but they passed a coding test - and could easily be a secret N. Korean remote software engineer.

Conclusion

The vocabulary told us the truth all along. "Data processing" described a world where data was primary and processing was the service. "Information technology" described a world where the technology estate — above all, applications — became the organizing principle, and data became its captive. AI is now dissolving the application as the center of gravity, exposing the problems with the model we built around it, and restoring data to primacy, with security following data wherever it flows.

Work is undergoing the same inversion, one level up. The craft era made the outcome the unit. Industrialization broke the outcome into tasks and put a human in each one. The corporation organized humans as the information system. The application era organized humans around the systems. The agentic era organizes the *process* around inputs and outputs, and treats humans and agents alike as invocable services within it.

Three arcs, one shape:

Data → From Data Processing to Data Primacy

Work → From Work Processing to Work Primacy

Security -> from Physical to "Attached"

Organizations that grasp all inversions together — data liberated from applications, work liberated from roles, security aligned with zero trust principles — will design their next operating model from first principles. Those that grasp only one will automate an org chart and processes that no longer needs to exist.

The Everpure Thesis

Everpure (NYSE:P) believes that data — and process — is the primary axis for enterprises. Our acquisition of 1touch.io, now Everpure Data Intelligence, reinforces that conviction by extending our ability to discover, classify and map knowledge across all enterprise data, including the mainframe. We can then apply consistent policy to the data with that contextual understanding.

As agentic solutions are more broadly deployed, the ways enterprises turn inputs into outcomes will continue to evolve. Applications will remain an important part of that equation, but increasingly alongside agents, models and humans. Everpure operates across this changing landscape: in cloud and on-premise, AI and traditional compute, application-centric and data-centric architectures, and multiple models and agents. The opportunity is to reduce the complexity across these environments while allowing data and process to remain the common threads connecting them.

Sixty years later, the lesson may be surprisingly simple: start with the destination, understand what you need to get there, and choose the best way to make the journey.

So the real question is: how do you architect an organization, work and data for today's reality?

Andy Brown — Everpure Board Member, Chair of the Compensation and Risk Committees; CEO, Sand Hill East

Charles Giancarlo — Chairman and CEO, Everpure

Ideas are our own.

Everpure helpers: Prakash Darji, Chadd Kenney, Penny Bruce, and John Gallagher.
AI helpers: Claude.