IBM System z and the Cloud

Insight

Gordon Haff
April 28, 2009

To purists, the "cloud" is an abstraction somewhere out in the network that delivers needed software services without regard to physical considerations like bandwidth or network latency. What they have in mind are providers like Google and Amazon. If they think about the hardware underpinning this cloud at all, they think about vast racks of x86 servers humming away. This cloud is effectively a standardized utility and is metaphorically equated to other familiar utilities such as the electric power grid.

However, we can also consider cloud computing as something broader. Something that encompasses all of the ongoing transitions to how we do all computing rather than just a subset. Something that takes place at mega-service providers, yes—but also something that happens within an enterprise datacenter. To be sure, cloud can be very relevant within enterprise datacenters, especially as IT organizations seek to provide more flexible, dynamic infrastructures, to become more service-oriented, and to further exploit the opportunities that globally connected networks provide.

We can debate whether a term like "private cloud" or "internal cloud" is an appropriate stretching of the metaphor or not. But that's just niggling over terminology. What really matters is not what we call it but the underlying reality of a style of computing built on top of more flexible and dynamic infrastructures that deliver software in a more modular and service-oriented way. And, when it comes to enterprise applications, does so without sacrificing the performance and reliability characteristics that such applications have today. Whatever wording the industry eventually settles upon, private, secured, inside-the-enterprise manifestations of the cloud are inevitable.

Unsurprisingly, cloud computing in this inside-the-enterprise sense makes demands on the underlying platform infrastructure that often aren't well-met by generic rackmount servers. Established enterprise computing platforms therefore still play an important—even necessary—role. After all, they already have many of the characteristics that virtualization and other technologies are just now bringing to scale-out x86. System z offers a good illustration of this. Already reimagined to leverage modern software technologies like application servers and Linux, System z is staking out its territory within cloud computing—especially within enterprise clouds.

System z's Renaissance

In many ways, computing has come back to the mainframe.

"Workgroup" and "departmental" computing came about because users didn't want to be beholden to IT administrators and developers behind the glass wall of the computer room. Indeed, much of the history of the computer industry has been about moving from monolithic computers to more numerous and varied distributed ones.

However, the centralized style of computing that the mainframe pioneered is back in fashion. Cheap hardware, deployed willy-nilly, turned out to have unobvious costs. It tended to have very low utilization, was hard to manage and update, and security was a big problem. As a result, even though scale-out x86 hardware is still very much in vogue, it's increasingly administered as a centralized resource under the control of IT. Cloud computing in its various forms takes this recentralization to the next level.

While this pendulum was swinging back, the mainframe itself also evolved. It adopted modern practices and open interfaces at a rapid pace—everything from CMOS processors to Linux partitions, from Java app servers to Gigabit Ethernet networks. Add to this traditional mainframe attributes that Lintel, Wintel, and Unix servers have adopted, at least in part, over the years. The "volume computing" world now covets traditional mainframe virtues like dynamic workload management, high levels of reliability and availability, and the ability to handle vast I/O loads. As a result, today's System z mainframe looks not so much like a relic of the past but an exemplar of where much of computing is going.[1]

In conjunction with these technology changes, IBM also began to change how it talked about and positioned the mainframe. No longer would it be treated as a "legacy" line cocooned by its historical folkways and its impenetrable alphabet soup of CEC, DASD, CICS, zAAP, FICON, IRD, GDPS, IMS, and RACF. Instead, IBM started to talk about System z roles such as "Secure Vault," "Business Policy Manager," and "Flexible Business Integrator" that leveraged existing strengths in business resiliency, workload management, and business integration.[2]

In concert with its z/VM operating system, the mainframe has also evolved into one of the ultimate Linux consolidation platforms—and, thus, very much a part of the network-centric world. One of z/VM's big selling points as a host for large numbers of Linux guests is that the Linux instances don't need to communicate with each other over a standard Ethernet network. While applications continue to use standard network interfaces, HiperSockets allow data to travel the system interconnect, improving both available bandwidth and latency.

IBM, often in partnership with SUSE (now part of Novell), has also made a variety of Linux modifications to better coordinate the way that memory gets used within z/VM virtual machines. Two such techniques are collaborative memory management assist (CMMA) and discontinuous shared segments (DCSS). CMMA extends the coordination of paging and other memory use between Linux and z/VM to the level of individual pages. This lets the z/VM host and the Linux guests better optimize their use of memory. With DCSS, portions of memory can be shared among virtual machines. Executable binaries that will be used by many or all Linux VMs are placed in a DCSS; all of the virtual machines can then refer to this single range of addresses.

In short, System z was a hub of sorts. It wasn't where all computing happened—processing power was typically cheaper, even much cheaper, on other platforms. But it could tie that other computing

1 See our The Mainframe Reloaded for a more detailed overview of the mainframe evolution up to the z990 generation of 2003.

2 See our The Mainframe Comes Down From the Mountaintop.

together. Such coordination only grows in importance with new forms of service oriented architecture (SOA)—which is a big part of what cloud computing ultimately is after all.

Enterprise Fabrics and Clouds

Cloud computing isn't literally the equivalent of SOA; it's both broader and less specific. However, especially in the context of enterprise computing, it speaks to many of the same IT requirements and takes some of the same approaches.

Those requirements include application and infrastructure attributes such as being event-driven, real-time, flexible, and secure—and having to deal with an increasing quantity of increasingly rich data. These are at least conceptually similar to some of the requirements associated with public clouds—such as Amazon Web Services. Modularity, Web-based access, self-service provisioning, and responsiveness to changing workload demands tend to characterize well-designed modern applications regardless of whether they run inside or outside an enterprise's firewall.[3]

However, conceptually similar does not mean identical. Applications and services will continue to run both inside enterprise firewalls and in the cloud for reasons of technology, switching costs, and control.

On the technical front, many of today's applications were written with a tightly-coupled system architecture in mind—for example, high performance Fibre Channel disk connected to large SMP servers—and can't simply be moved to a more loosely-coupled cloud environment. Latency still matters. Data often has to live close to the programs that operate on it and application components often need to interact across optimized networks and interconnects.[4] While a Software as a Service (SaaS) provider can optimize infrastructure to deliver a specific software service, now it's the software that has to be relatively standard. In the public cloud, you can't run software that's specific to, or highly customized for, your business and simultaneously expect to have a hardware environment optimized for it.

For existing ("legacy") applications, there's also the switching cost and time to move wholesale to a new software model. In fact, one of the big arguments for standardized, outsourced IT—that it allows companies to focus on their competitive differentiators—also argues against making investments to change functional systems (and their associated business processes) if the financial benefits are long-term and somewhat amorphous.

Security, compliance, and a generalized sense of control are also critical factors that will keep applications internal to organizations. Public cloud proponents argue that we're talking here about perceptions rather than reality; it's not like enterprises have exactly proven themselves immune to data breaches. However, for one thing, like it or not perceptions drive reality. For another, companies in many industries have to conform to a wide range of regulations that include the way they store and process customer data—regulations that we should expect to become more rigorous rather than less. And such regulations are largely predicated on a traditional enterprise computing model. It will take, at a minimum, time to adapt and expand regulatory regimes to different styles of IT. System z has long been at the pinnacle of secured computing. Among other things, it benefits from decades of security hardening, sophisticated access control systems (e.g. RACF), specialized hardware for accelerating encryption, and decades of in-the-field success in areas such as banking exchange networks where vast sums of money are processed and transferred daily. If one needs a secure hub for high-volume, high-value business process, System z is automatically on the selection short-list.

In short, this coming wave of computing incorporates many things that are genuinely new and deals with computing problems of a different type and scale from traditional IT. But, at the same time, neither existing enterprise requirements nor

3 I use "firewall" here as convenient shorthand. However, the relevant distinction is often a more complex trust boundary of some sort as opposed to a physical barrier.

4 See our Latency (Still) Matters.

the software created to address them are going away—nor should they. Therefore an enterprise cloud computing strategy isn't about wholesale rip and replace, but about integrating new applications and operational practices with existing ones and orchestrating them in a coherent way. Integration and orchestration are two areas where System z has a track record.

System z Today

System z pioneered many of the techniques and attributes that are now being widely discussed in the context of distributed systems.

Perhaps most obviously, what we now call server virtualization first appeared commercially on an IBM mainframe (the System/370 in 1972). And the combination of System z, z/VM, and Linux remains a particularly sophisticated example of it, in part because the multiple levels of the hardware and software stack are integrated with each other and work in concert.[5] This in turns leads to the use of hardware in a way that's both efficient and tightly managed. And that means large numbers of workloads can run simultaneously and still be kept out of each other's way.

More broadly, the efficient management of mixed workloads has long been a System z theme. This mindset originally was born of necessity; when hardware was as expensive as it used to be, you just couldn't afford to waste any of it. But even in a world in which hardware isn't quite so precious, workload management remains an important attribute of servers and datacenter architectures to ensure predictable performance and to avoid application conflicts.

And, thus, System z is very oriented toward juggling multiple workloads—all the while keeping them out of each other's way, assigning them appropriate resources, protecting them against hardware and software failures, maintaining response times and other service levels through varying load, and ensuring that they're protected against failures. Ensuring a reliable, secure, and performant platform for these many workloads has meant devoting more circuitry to things like channelized I/O, cryptographic acceleration, RAS (Reliability, Availability, and Serviceability) features such as instruction retry, and larger caches to better handle larger memory working sets. We're starting to see more of these types of features on other architectures as they mature—but they were first on the mainframe and, for the most part, they are still most refined there.

And this view of the mainframe as an integrator of heterogeneous workloads clearly feeds into IBM's view of enterprise cloud computing which also has management as a central tenet.

A Managed, Enterprise-Focused Cloud

"Web 2.0" clouds tend to be based on relatively homogeneous scale-out infrastructures running mostly open source platform software.[6] The applications they run—which is to say the familiar Web sites like Facebook used by millions—were constructed from the ground up for these environments. They have both fundamentally different starting points and design centers from the typical large enterprise. For example, handling rapid growth (as best they can) at the right cost points tends to trump attributes like rock-solid stability. That's not to belittle the challenges of the massive scale that successful Web 2.0 companies have to deal with—or the importance of protecting their customers' data. But, at the very least, they view attributes like manageability, scalability, flexibility, reliability, and so forth through very different lenses from their enterprise counterparts.

IBM, on the other hand, approaches cloud in an enterprise context as essentially an evolution of enterprise software and processes into the cloud, as opposed to an evolution of Web 2.0 into the enterprise.[7] This view of the cloud is through the top-down lens of an enterprise architect in contrast

5 See our z/VM: Teddy Bears and Penguins.

6 See our Open Source in the Next Computing Wave.

7 This is not to suggest that all IBM cloud efforts have a traditional enterprise flavor. Its System x iDataPlex, for example, is specifically designed for large scale-out datacenters in the Web 2.0 vein.

to the bottom-up, tactical, always-in-beta methodology that's been most associated with the consumer cloud.

Consider the vice president leading IBM's overall cloud strategy, Erich Clementi. In a prior posting, he ran the System z business. It's also notable that he's currently general manager of Enterprise Initiatives. At IBM, cloud computing is an enterprise initiative—it's not just about "Web 2.0" or "services for startups," but rather about building and running network services that enterprises can take advantage of.

It therefore makes sense that manageability is central to IBM's view of cloud computing. This includes many well-accepted enterprise management concepts like workload management, service automation, compliance, and orchestration. Tivoli, which owns many of IBM's management software assets within the broader IBM Software Group, plays a key role. Tivoli software in categories such as asset management, business application management, security management, server, network, and device management, service management, and service provider solutions—all these come to bear in IBM's cloud.

In short, IBM's approach to cloud architectures is enterprise-oriented. It's naturally connected to, and supported by, IBM's portfolio of enterprise management software. And this is the context in which System z plays a major role.[8]

Where System z is Headed

System z's future—and its role in cloud computing—is in some respects an extension to the enterprise hub concept that IBM introduced in the context of its On Demand strategy. The conceptual similarities are there in attributes like application-level availability and resiliency, policy-based automation, high levels of security, and end-to-end systems management. What is new (or at least taken to the next level) is an emphasis on a heterogeneous dynamic infrastructure of which System z is just a part, albeit the part in the middle of things. This is the next step in a System z evolution from legacy and insular to relevant and integrated.

This is not to suggest that IBM has been a proponent of a single architectural approach for every situation. The variegated nature of IBM's System products would put the lie to any such notion. However, the notion of different types of servers working together (and being sold) in an integrated way—as opposed to just appealing to different types of organizations or different application sets within a given organization—is relatively recent at IBM.[9]

IBM talks about this heterogeneity using a concept called "ensembles." Essentially, an ensemble is a grouping of a particular type of system—such as System z or System x. Each ensemble has its own management tools to do things like configure and control hardware, autonomically allocate resources, manage energy consumption, and so forth. These ensembles then expose management interfaces to higher-level service lifecycle management tools that compose services, determine the optimal placement of workloads across ensembles, create policies, and provide monitoring dashboards.

The System z ensemble—which you can also think of as the System z "cloud"—has three major components.

The first is **the mainframe itself** that, as we've discussed, already incorporates many of the attributes that distributed, scale-out architectures are working to implement. Stability and effective workload management are traditional mainframe attributes and mainframe VMs offer the ultimate flexibility in divvying up a single large system into smaller pieces, which can be quickly created, resized, or destroyed. Beyond partitioning and virtual machines, the list of techniques the rest of the world learned from mainframes goes on at some length: policy-based workload management,

8 Though System z will usually be deployed within enterprise boundaries, it could be used for roles in other cloud infrastructures, for example handling high-value financial transactions or solving performance bottlenecks common to distributed applications (see e.g. *MMORGs on Mainframes*).

9 See our IBM Continues to Submerge its Product Brands.

multi-system workload distribution, hierarchical storage management, intelligent I/O channels, large memory support, capacity upgrade on demand, role-based security, intrinsic cryptographic acceleration. Distributed-systems managers have also gradually adopted many of the attitudes and techniques of mainframers, such as today's focus on repeatable datacenter procedures, or the design philosophy that calls for everything to be instrumented and monitored in order to prevent or correct errors at the most granular level.

The second is **accelerators**. While System z continues to advance on the processor performance front, the reality is that its "MIPS" are still relatively expensive. IBM has dealt with this to some degree with what amounts to pricing breaks for hardware used to run workloads that aren't traditional mainframe workloads. Thus, you can buy a System z processor dedicated to running Linux (an "IFL," Integrated Facility for Linux) for less money than if you use it to run CICS. zIIPs and zAAPs are other examples of "specialty engines," as IBM calls them.[10]

However, pricing magic only takes you so far. IBM is also on the path to various forms of hardware acceleration. An early example is the decimal floating point accelerator introduced with the current z10 generation. Its ultimate direction is to have application serving blades that logically integrate System z resources with other types of processors[11] for applications with a close affinity to mainframe data.

The third, and arguably most important, component is **management software** and firmware. The System z Integrated Systems Management Firmware will manage resources, workloads, availability, images, and energy in a federation way across not only System z's own z/VM and z/OS operating systems but also PowerVM and KVM hypervisors running on integrated non-z hardware.

Today's zCloud

To be sure, full heterogeneous integration is in the realm of "zFuture." However, even today's System z lends itself well to running and managing the sort of inside-the-firewall, composite, service-oriented workloads that are often the subject of "private cloud" or "enterprise cloud" discussions.

These workloads have a foot in two worlds. On the one hand, they have requirements akin to "traditional" enterprise applications—things like customization, optimization, security, privacy, availability, and so forth. On the other, they're more modular, are built with some services orientation, and, in general, have at least some of the self-service, Web-y characteristics that people associate with cloud applications.

The platforms on which they run must likewise have a foot in both worlds. They need to have the performance and reliability attributes traditionally associated with enterprise "Big Iron" while simultaneously being able to adapt to the flexibility and management requirements of applications that are more modular and dynamic and often assembled using different technologies from what has been the norm for enterprise applications.

System z has a track record of dealing with this sort of dualism. There's no commercial computing platform that is more associated with traditional enterprise transaction processing and computing than the IBM mainframe. And the mainframe's renaissance has, to no small degree, been about its embrace on new workloads and software technologies from Linux to Java to relational databases. Of which cloudy apps are just the latest.

System z is ready to stake out its role in cloud computing—especially for high-volume, high-value, must-be-available, must-be-secure roles within enterprise clouds.

---

10 See our z10 EC: The Mainframe Bulks Up.

11 IBM has specifically mentioned x86, Cell, and POWER.