C3B Consulting's

INSIGHT-SPECTRA

Applying common sense to using technology intelligently

www.insight-spectra.com

Volume 24 Report 3; May 2011

Have you decided which is your primary device?

Charles Brett, C3B Consulting Ltd.

Management introduction

The iPad has reinitiated — if not reignited — interest in tablet computing. With more than 20M iPads sold Apple continues to prosper. Running fast to catch up are a multitude of suppliers using a variety of hardware platforms (from ARM-based designs to Intel ones, and others) and various operating systems — varying from Android to Windows to QNX (RIM) and WebOS (HP).

Thus far, and in keeping with Apple's preference for a consumer focus, most interest in tablets comes from 'single users' — consumers — rather than organizations wishing to use tablets for commercial purposes. That restraint will not last. In any case, many of the vendors choosing Android, Windows, QNX or WebOS are looking for their prime selling opportunities to exist within commercial organizations at least as much if not more than in the Apple-dominated consumer environment.

Yet, throughout all the 'new excitement' about tablets (of whatever flavor), an old, old dilemma has started to reassert itself. It has, at its simplest, the issue of — what device has primacy? Or, put another way, which is the user's primary system and where is his or her primary data?

Consider what has happened

In the 1970s and arguably for most of the 1980s, the primary source of both processing and data was held on some form of centralized host. That host might be a mainframe or a mini-computer — with attached terminals. Whatever the system, data and related applications belonged in the center, and was available and backed up.

In the late 1980s and more particularly in the 1990s the mass adoption of personal computing (delivered on PCs) changed the focus. No longer was it quite so clear where 'master-data' was stored. But, because, the original PCs did not move around much (too big and clumsy), from an IT organizational viewpoint this did not present too much of a challenge: with near permanent LAN connections, data copying and backup was practical and the PC with data and applications existed where the user worked.

The arrival of inexpensive, portable laptops fundamentally changed this. More and more people (whether for themselves or for their employers) started to move around, taking their device with them. While some software — notably Microsoft's Exchange and IBM's Lotus Notes — provided mechanisms (often awkward in practice, like Notes' synchronizing) to ensure that the center also had copies of data, not all applications did. Control of data passed, gradually and de facto, into the hands of users who often did not possess the discipline to plan for 'what happens if my data, or system, is not available'.

Furthermore and complicating the picture, users started to install those applications they felt they 'needed', in addition to what organizational IT might select or try to dictate. This just added to the loss of control (and most attempts by IT to retake control simply did not work).

In effect the arrival of the readily move-able laptop (and then the netbook) saw the primacy of data being held at the center break down. The same applied with applications: people wanted to have the applications they needed with them, which meant that 'using' general purpose PCs as occasional terminals to access data became less and less useful. Even if you had your data with you, if you did not have your applications you could not do what you intended.

Despite IT creating ever more elaborate 'schemas' and 'solutions' (for example, organization-specific virtual machines running on devices increasingly chosen and bought by the user), these 'solutions' were and are cumbersome. Moreover, so long as an organization was not burned too badly or too often, an increasingly relaxed laissez-faire attitude took hold.

This operated pretty successfully into 2010. Yes, there were crises. Yes, there were disasters (think of both commercial and government laptops being stolen). But there were not too many of these.

Indeed, another reason this relaxed attitude worked was that many IT organizations were spectacularly ineffective about enforcing the very disciplines they claimed to promote. For example, one professional services organization bought reputable software to back up automatically, and in background mode, all its employees' laptops. But it never checked to ensure that, once that software was installed and running on each of its employees' laptops, it actually took the periodic copies. More than one employee found himself bereft of data and applications because there was no process to ensure that the necessary copies were actually made back to the center. This was definitely a case of optimism triumphing over intended practice.

Now think about what is happening

As 2010 turned into 2011 the environment has significantly changed again. It has become even more complex.

In the past, mobile or cellular phones were largely regarded as devices distinct from computing devices. Unless your organization lived in a Blackberry (RIM) world, then responsibility for (say) the employee's contact information on mobile phones was more often than not the responsibility of the user (who was also often the owner of the phone, because choosing a cellular phone was regarded as a personal rather than an organizational choice).

With the arrival of tablets that are essentially near-full scale computing devices plus the delivery of smart-phones having the power and capabilities of the iPhone/iOS or of Android 2 and 3 devices — with screens varying from 4" to 10" via a plethora of sizes in between — the laptop has lost much (but not all) of its primacy as 'the' device of choice.

At the same time, these tablets and smart-phones have not occupied the primacy slot that the laptop had occupied. Instead a new interregnum has materialized, presenting new dilemmas for users:

- should one device deliberately be selected to have primacy (and what does this mean)?
- what should one do if you want 2 or more devices all to have equal primacy?
- should one take 1, 2 or 3 devices when traveling?
- where should one's master data and applications reside (and is this backed up/secure)?
- etc.

What has changed is that increasing numbers of users have 2 (more likely 3) devices which they use in different but complementary ways. What is also true is that the applications on the various devices do not complement each other well. Indeed, possibly the deepest argument (that with the greatest substance) in favor of Windows on a tablet is that it will be application compatible with ones on the laptop (think Office and/or OpenOffice). In contrast, most Office-type offerings on iOS, Android, QNX and WebOS are only partially (or not) compatible.

Thus the environment in 2011 is one of uncertainty about:

- which of multiple portable devices to use, when each can be important for doing one's job
- where master-data and its associated applications exist
- recoverability.

Where might solutions come from?

If you accept the assertion that 2011 marks when change has arrived — in the shape of more and more people carrying 3+ intelligent devices (smart phone, laptop and tablet) — then alternative solutions are required. The clumsy, if early, synchronization approach adopted in Lotus Notes is not sufficient. Nevertheless, something similar is needed that can work across all 8 main portable user platforms:

- Windows
- OS X
- Linux
- Android 2
- Android 3
- iOS
- QNX
- WebOS.

In this context there are two distinct and parallel requirements. The first concerns data. Users will want to know that data created/stored on one device can be accessed and used by another. Furthermore, this must be capable of occurring when connected (to the Internet) or disconnected — with the user deciding.

Second, and at first glance more difficult to solve, there must be the applications which can use this data (originated on multiple devices) in ways that are meaningful. In practice this may turn out to be easier

to address. The times that most users need applications that work with their data is when they are able to be connected (that is in the office, in a hotel, at home, etc.). The disconnected times (like in a plane, at least until affordable Internet connectivity arrives, or in a car or on a boat) are relatively short within the context of most working weeks (defined as those times when you wish to work).

So what might happen? One solution model is that some form of common space becomes defined that organizations and their users can exploit (this can be private or public or both). This may be like a form of 'Dropbox' which goes further.

Dropbox and its likes are already good for data. They could become the master-data storage point. They would become even more useful if and when they have a space where those applications that a user needs can run against that data in that common storage point if the key device is not present (like you have taken your tablet and smart-phone on the road but not your laptop but now you discover you must try to do something that needs your laptop…).

What needs to happen

For this to happen, major changes are going to be needed. Not only will the storage and sharing/synchronization need to improve but software licensing will have to change.

In addition, the way we think about applications and how we buy and use them will also need change. At present you buy for one platform, and that is that. But this no longer reflects how users want to work.

Perhaps a better model is similar, if much extended, to the one which Amazon is putting together that enables Android users to try out applications on a virtual Android platform running as an Amazon service. Take this a little further. Enable users to access to virtual execution environments where users can run Windows, OS X, Linux, Android, iOS, QNX or WebOS (and any others) and their particular application against their primary data and a different style of solution comes into being (for which one will pay by the service usage).

Technically, to do this with Linux is easy. Android is almost there. Elements of Windows applications (Office365, but running only on Microsoft platforms and with constrained functionality) are there. QNX and WebOS are probably not too difficult, which leaves OS X and iOS as being different (and Apple potentially exposed if it resists as it likely would). What is not so easy is the software licensing to enable this — which is what may inhibit progress.

Management conclusion

The point is that, for a user with multiple 'primary' devices, the need to choose primacy for only one of his or her devices would disappear. Instead a more practical world would emerge, one which also will make it easier to travel or to use what the individual circumstances make more convenient. Every user will know that all of both his, or her, data and applications are available and usable whenever one has connectivity.

This would be a wonderful step forwards. It would recognize today's reality, that no one device can do everything that a smart-phone, tablet and laptop each can do. Yet it would enable you to do that you wish with all of these — with users and organizations being able to make the choices that they prefer — in parallel and with the likelihood of data and applications 'going missing when needed' being pretty much eliminated.

Ultimately this would simplify the management dimension for IT, and make it more reliable. But the price will be acceptance of major changes by software vendors which history suggests will not be easy to obtain.

Meanwhile the 'primacy problem' will continue to stretch IT and ittitate users. That is the reality for 2011 and probably 2012.

Charles Brett
C3B Consulting
www.c3bconsulting.com

IBM's 2011 System z: Windows support is big news

Wayne Kernochan, Infostructure Associates

Management introduction

This analysis is not meant to be a simple description of IBM's plans, nor do I intend to say what Windows support 'should' mean: IBM is certainly better qualified than I to enunciate that position. Rather, it is my take on what may happen at some time beyond IBM's recent roadmap and statement of direction about Windows and System z. I simply assert that, in my opinion, it will be important and in customers' best long-term interests if IBM later goes in the direction that I describe.

Indeed, as you will see, I urge customers to consider whether I am right. If you think I am, I suggest what you should to do about it. What is my position? I will argue that this Windows announcement may turn out to be highly significant to users and at least as important as all the other recent IBM announcements about the cloud, Integrated Service Management and extensions to analytics. How could this be?

The initial impression

At first glance, IBM's recent 'statement of direction' of support for Windows on its zBX platform during its recent mainframe analyst summit seems like small potatoes. For one, this support will not arrive until 4Q2011 and, secondly, it will involve only the blade side of zBX (Windows virtual machines running on specific System x-type blades) and the umbrella zBX virtual-server/storage/network admin tool — known as the Unified Resource Manager (URM). Full support of application-level administration and load balancing for Windows platforms is not yet on the IBM roadmap. As for full integration of Windows platforms with the mainframe in a private cloud, that is not yet on the roadmap either — and is, perhaps, a goal that may never be reached.

This does not mean that IBM's planned support for Windows on zEnterprise is incomplete. On the contrary, IBM has applied its usual combination of focus, strong technology and services to enable users to implement IBM System x blades running Windows that integrate seamlessly into an overall zEnterprise solution.

The state of Windows-mainframe integration

Of all of today's popular server platforms, Windows and the mainframe have proved the most difficult to combine. The reasons appear to lie partly in technical difficulties and partly in a sense by the vendors involved that users just do not care that much about such integration.

Surprisingly, however, much effective work has gone into integrating the two. The best way of thinking about this is to imagine an integration continuum:

- at one end, the two platforms are entirely separate
- moving gradually towards the further (other) end, first there is loose coordination; this is where the two platforms can exchange messages to carry out common admin. tasks
- next, there is 'umbrella' integration, in which a distributed application on both platforms communicates in a common language to enable the user to issue one command to both platforms (a multi-platform admin tool like IBM Tivoli is a good example)
- moving closer to full integration comes the ability to run multiple copies of an application on both platforms, with load balancing between the two delivered via an 'executive' mechanism
- nearly at full integration, there is 'static' integration, or the ability physically to move copies of an application between the two different platforms
- finally, there is 'dynamic' integration, in which the two platforms can act identically, by moving application copies from one platform to the other practically instantaneously, and by providing a 'veneer' that makes one platform seem the same as the other to an application or even a user.

Right now, in the world of Windows and mainframe platforms — to summarize drastically — users possess potential 'static' integration for many applications. But there is not much real-world translation of

applications from Windows to the mainframe — or vice versa. To most practical intents and purposes we are pretty much at the loose coordination stage. The basic underpinnings, however, for turning everything into 'static' integration (or even full 'dynamic' integration) are present::

- URM shows how to bridge z/OS and non-mainframe Linux at the infrastructure level
- Tivoli shows how to bridge z/OS and mainframe Linux at the application level
- together these do much of the job of handling 'umbrella' and 'executive' integration.

Indeed, fledgling third-party migration products — like Novell's SLES Mono Extension, Wine (open source), ReactOS, and Mantissa's z86VM — support Windows APIs and .NET. In the case of Mantissa there is promise in some cases of the ability to transmit object code that can run without change — 'dynamic' integration heaven. But without major user and IBM support, practically speaking, these products are unlikely to have a major impact.

Furthermore, it should be emphasized that, at present, neither these further steps in integration nor IBM implementation of the required code are on IBM's roadmap. Rather, for now, IBM is aiming at supporting an 'infrastructure as a service' basis across zBX Windows/Linux scale-out blades and scale-up Linux-and-z/OS mainframes. The main change in either URM implementation or IBM support in 4Q11 will apparently be simply to port a copy of URM's workload-aware KVM (Kernel-Based Virtual Machine) Linux hypervisor to Windows.

The promise of Windows + Linux + z/OS support

Why might full Windows/Linux/z/OS integration — in contrast to IBM's roadmap of today — matter? First, Windows, Linux and z/OS (on the mainframe) occupy the lion's share of most large organizations' important platforms — with a majority of enterprises using all three. If we can imitate enough of Windows on Linux or on Linux-supporting blades we can logically move most Windows applications to scale-up servers when needed (UNIX/Linux and/or mainframe). In so doing we will have achieved:

- source-code compatibility from Windows to Linux and Java real-time portability from Linux to Windows
- source-code compatibility for most Windows applications from Windows to Linux on the mainframe, and Linux source-code compatibility and Java real-time portability from Linux to the mainframe and back.

It would also be nice to have portability from z/OS apps to Linux and Windows platforms. But neither large enterprises nor even cloud vendors really need this yet. The mainframe has too strong a TCO/ROI and energy-savings story for large-scale and numerous (say, morethan 20 applications) situations.

What is the result of all this? It is an almost complete 'static integration of all existing applications in the cloud or in an individual enterprise.

Second, I believe that Windows + z (in a mainframe 'hub' configuration that does not require any workloads to go to scale-up or scale-out) matters to users right now. There are clear and cogent reasons for users to want to integrate Windows workloads with the mainframe:

- one is Systems z's superior energy profile
- another is the ability of the mainframe to deliver long-term costs per application well below Wintel scale-out environments in cases where the enterprise moves 20 or more Windows applications to a single mainframe or zBX.

More urgently, the aim of today's cloud computing implementations is to cut large-scale costs — both of administration and of per-application hardware — while improving overall robustness and security. Mainframes excel at both. But, so long as the mainframe cannot support a large body of enterprise Windows applications, external cloud vendors and/or internal cloud creators typically have to decide between:

- a partial solution that replicates the existing Wintel/mainframe split (which does not save as much in the way of costs)
- movement towards Wintel (that increases long-term costs).

More realities

To repeat what I said above: IBM's promised Windows support on zBX, on the face of it, takes us only a little way down the path to the full Windows-z integration envisioned above. It applies only on a 'hybrid' archi-

tecture (scale-up z196 mainframe plus scale-out blades in a zBX enclosure) that is new and not yet widely used. In addition, the tools to treat a Windows virtual machine as just another system for purposes of backup/recovery, monitoring, load balancing and distributed-application creation within zBX will not be ready from third parties in 4Q11.

Still, even with these limitations, IBM's promised Windows support will be immediately valuable to many organizations. IBM's zBX is a story of scalability, and specifically of driving application performance and capacity higher where scale-up plus scale-out integration makes sense. Since Windows applications are predominantly scale-out, that means taking high-scaling Windows applications — 20 copies or more — and putting these on VMs on a zBX. Thus, email storage plus analytics involving text is one obvious workload candidate. Another is Web-site access to business-critical mainframe applications.

Moreover, not only will IT be able manually to begin to place Windows and the mainframe in the same administrative framework — with major benefits in important systems administration people costs — IT will also be able to compare and contrast more effectively the relative strengths and weaknesses of the two platforms. This will make even more effective the planning/allocation of resources for key new applications.

I argue, further, that the true value-add for users lies in the belief that customers should encourage IBM to undertake — or at least support third-party efforts to undertake — the rest of the journey toward integration. This should be similar to when customers led IBM to make this support commitment. (I can certainly cite times when such hopes did not pan out. Sun's never-wholehearted Windows support and the popular but never augmented IBM AS/400 Windows 'attached PC' are but the most obvious past examples.)

Today, however, I believe that customer requests for Windows support in mainframe-centric architectures will turn out to be the tip of an iceberg. There is indeed a pent-up demand for better Windows integration in major enterprise data centers. In parallel there is a case to be made to cloud vendors that now they can tap into mainframe scale-up's TCO, robustness and security advantages on a much greater scale (likely a significant driver given cloud's wide use of Wintel-style servers). If IBM responds to customers, the obvious path to user satisfaction is to move towards greater Windows-z integration. zBX is a great testbed, with little risk for any concerned.

Management conclusion

Do not just kick the tires: drive them. The type of rearrangement of users' present architectures involved in adding zBX-based Windows workloads using IBM's 4Q11 Windows support is extraordinarily low risk. Mainframe consolidation has already demonstrated low risk and low cost, fast migration of Linux applications. Here application recompilation is not even needed in many cases.

IBM has been managing Windows applications under the same general infrastructure framework as Linux ones for a while already. It has been integrating its management tools across all platforms for more than a decade. And, of course, once applications are moved, they begin to obtain the advantages of the mainframe's robustness and security technologies that have been baked into URM, and which are now being baked into IBM's Integrated Service Management (ISM).

The bottom line for users, given IBM's slow and careful approach, is that they should feel entirely comfortable in shortening the testing/benchmarking phase of implementing 'Windows support' and moving quickly to full implementation of major Windows functionality. This will serve two purposes; it will shorten the payback period and enable users, if they are so inclined, to keep the pressure on IBM to move ahead, as implementations identify further areas for Windows-z integration.

The chorus of inquiries from analysts and users that IBM reports shows that enterprise customers see the potential of the company's existing roadmap. Even next year, since Windows support will only arrive in 4Q11, the real-world benefits of any further Windows-z integration will still be in question. Yet this is good news for savvy users: they will be out in front of the pack, at little cost and with low risk — while other IT shops and vendors alike slowly realize there is more here than once they had thought or even imagined.

Wayne Kernochian
Infostructure Associates
www.home.comcast.net/~kernochanw70

Continued reflections on software rusting: Part II

Larry Fulton, Enterprise Architect

Management introduction

*In the first part of this analysis (***INSIGHT-SPECTRA, January 2011***) Larry Fulton discussed the factors that make software-based business solutions degrade over time, exhibiting all the signs of 'wearing out' that we normally associate with physical systems. The causes included changes in utilization, fragmentation and other cumulative impacts from within the system itself, plus the ongoing refocusing of resources to deliver new solutions over time.*

From a holistic, or 'whole system', perspective this observed tendency results from the complex interaction of many components. These include the business itself, and the somewhat independent evolutions of the many constituent components.

In this concluding analysis Mr. Fulton:

- *examines common approaches to solving this problem*
- *offers his own perspective on the 'next practices' that organizations should use to drive further improvement.*

To recap

A brief recap of Part I of my original argument is in order. The 'rusting' of business solutions occurs because of the various performance and operational challenges that plagues all oftware as it ages. This is caused by many factors, typically related to the independent evolution of the business and its technical environments. The consequence is an increasing mismatch between the solution and the components that each solution relies upon.

In Part I, I talked about these kinds of symptoms:

- 'young' systems, which largely enjoy the consequence of investment in working out initial implementation and functional evolution issues
- 'older' systems, which tend to be functionally stable but are more at risk from changes in the surrounding environment.

In effect, newer systems have the full attention and resources of the business as they seek to reach and maintain operational effectiveness. In contrast older systems do not enjoy this focus. Plus, the challenges facing older systems include the lack of business functionality evolution as well as a surfeit of these evolutions and changes in their non-solution-specific components (the software infrastructure).

Existing approaches

The implication is that resources need to be focused on maintaining stability and predictability in the overall collection of non-solution-specific components. Or, as I stated in Part I: "how do we establish and maintain a stable, high-performing collection of non-solution-specific components as a hedge against foreseeable late-stage degradation of our business solutions?"

Existing approaches to this problem tend to fall into two categories:

- build the cost of this maintenance into individual solutions' long-term maintenance
- leverage non-solution-specific components across as many solutions as possible to minimize the per-solution cost of required maintenance.

In the first of these approaches you literally build in the cost, expecting the business regularly to continue to invest in keeping the solution up-to-date. Some organizations implement a 'tax' on new development (often in the neighborhood of 10%) and reserve the funds specifically for this kind of maintenance. There are similar variations.

The problem with this approach is that it is often difficult to sell to stakeholders and even more difficult to sustain over the long term. For organizations that can pull it off, it is workable.

Another common way to build in the maintenance cost is to buy off-the-shelf solutions and hold the vendor accountable for the maintenance. One trap here

is failing to justify the investments required to keep the solution up-to-date. Then there are always problems that fall outside of the vendor's responsibility. One result has been the rise and acceptance of Software-as-a-Service models.

However successful you may be in adopting this second general approach, you can not help but have a situation where each solution incurs its own costs, at least to some extent. In contrast, in the first approach, efforts focus on establishing standard application stacks. The theory, which has proven itself in practice, is that by building many solutions (off the shelf, custom or hybrid) on the same set of supported software and hardware components, an organization can afford to keep that stack up-to-date, ensuring that patches and upgrades to individual components receive the appropriate testing and attention.

Sharing integration infrastructure serves the purpose of leveraging maintenance investments across solutions. (Unrelated to the current discussion, it also separates integration work from the solutions.) Using multiple commercial applications from the same vendor provides similar benefits. This is especially true when vendors employ common elements across applications.

There are many other examples, most of which are common practice — in particular using standard:

- operating system versions
- server configurations
- database configurations
- shared application and web server farms

to name just a few.

Future practices

Organizations, looking to keep business solutions performing well and functionally close to evolving business needs, must exploit these philosophies, seeking both to:

- build ongoing evolution strategies and costs into their solution planning
- leverage the investment across as many solutions as possible.

What will these future practices look like? How can organizations begin to take advantage of them?

The use of locally standard technology stacks and integration facilities, effectively leveraging non-solution-specific components across many solutions, is only a beginning. The real questions are: how far can this idea go and what are the practical limits? Before I consider these, let me first look at the general structure of a platform from an IT perspective (Figure 1).

Essentially, the red line in Figure 1 this divides 'solution-specific functionality' (representing traditional project-based investments) from the IT 'solution plat-

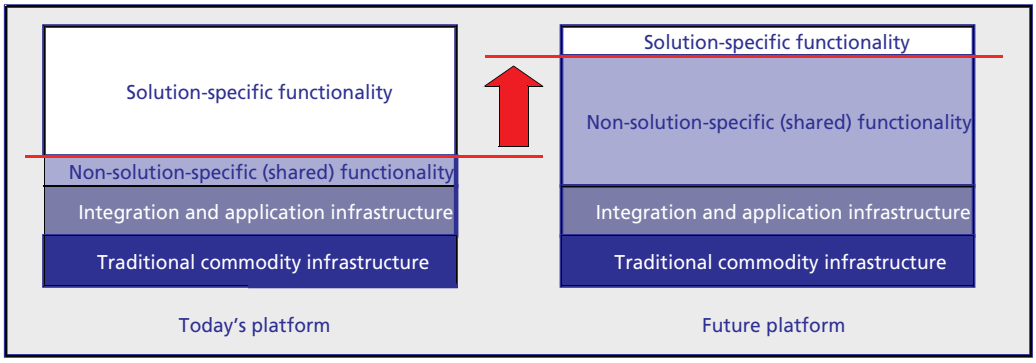

Figure 1: Today's and Future Platforms contrasted

form'. Such a platform represents the long term investment supporting individual solutions. Or, put another way, the focus is on what can be leveraged across many solutions. The basic objective is to move as much as possible into the platform over time, effectively enlarging it and thereby reducing the cost and effort associated with individual projects.

Sounds simple, right? There are of course challenges in practice. Let me take a closer look through a series of three relevant questions.

How is this different than what we are doing, today?

Conceptually, this is similar to today's practices. The basic concept is to share as much as possible across solutions, reducing the per-solution cost of maintaining the environment.

This alone does not, however, address the challenges of keeping the solution-specific functionality aligned with current business needs. More is needed.

What investment priorities are needed to make this work?

This is really the most important question of all. On the surface, we are reducing the per-solution cost of delivery and maintenance. But we then have a growing platform full of technical and business functionality to support, and that platform can and will rust and age just like any of today's software solutions.

The bottom line is that you cannot ever stop investing in the platform — for the simple reason that the business and technical environment is always in motion. However, by reducing the per-solution costs dramatically, you should be able to attain a position which justifies the ongoing cost of maintaining and evolving the underlying platform, and then enjoys the benefits of its high leveragability.

To be successful, the platform and its constant evolution, including decommissioning of obsolete functionality and ongoing enhancement to meet future needs, must be accepted by management as a funding priority. In practice, it is easy to imagine this funding argument becoming weaker over time. It is important, therefore, that stakeholders understand the value of the platform as a strategic asset, year after year after year.

What proofs exist to demonstrate this can really work?

Service-oriented architecture (SOA) is the best one. Many organizations have been successful in creating broad business service portfolios, driven by the desire to have one implementation of critical business logic. This has produced exactly the kind of non-solution-specific shared business functionality that is critical to this platform approach.

It is more than 'just' SOA, however. Because the goal is ruthlessly to pursue this approach across the board, success only happens when you drive common functionality and infrastructure into the platform *wherever and whenever* possible.

Management summary

Mr. Fulton started these analyses discussing the phenomenon of software business solutions degrading ('rusting') over time. He explained that this was a natural result when technology — as well as the business — moves forwards around a solution that over time receives less and less care or attention as newer challenges take center stage.

He then examined some of the root causes as well as some of the approaches organizations have taken to solve these problems. Commoditization of infrastructure across the industry, standardization of local technology stacks and Service Oriented Architectures all have the characteristic of enlarging the proportion of IT assets shared across multiple business solutions.

Finally, he looked at taking this concept as far as possible — recognizing, then investing in and even enlarging the overall 'platform' to encompass everything that can be shared on behalf of a minimal set of solution-specific components. Making this work requires a sustained focus on the platform and making clear its value to the business. This effectively drives ongoing investment in order to keep the platform operating efficiently, satisfying new needs and maintaining its leanness and trimness by decommissioning out-of-date features on a regular basis.

In practice, this approach may be something to which to aspire. But, as Mr. Fulton argues, it is an objective that you will prove over time.

Larry Fulton
Enterprise Architect

Selected common sense Dos and Don'ts — before entering the cloud

Charles Brett, C3B Consulting

Management introduction

Talking about the attractions of cloud computing is all the rage (and not dissimilar to that when Grid Computing first appeared). One can understand why: the notion of an electricity-style 'usage plan that you turn on when you want and switch off when you no longer need' is attractive.

Entering a cloud in ways that are 'business beneficial' is not as simple as watching a plane passing through a cloud in the sky above — but becomes similar when one considers the work that needs to be done before that plane can pass through that cloud. Planes do not just materialize.

Flying is the fruit of much preparation before a plane even takes off, never mind climbs towards the clouds. In this analysis Charles Brett briefly discusses a selection of 'Dos and Don'ts' that management should bear in mind before expecting the benefits of cloud computing to materialize for their organizations.

Dos and Don'ts — a starting list

Common sense is wonderfully easy to describe, and much harder to practise. In this brief analysis the following Dos and Don'ts may sound trite as simple statements — but that does not mean they should be underestimated:

- know what you have before you start
- decide what you can improve (and what you cannot)
- understand what your savings objectives are and what they entail
- create a plan likely to deliver
- measure what you are delivering and review what happens
- do not permit the 'candy-store effect'
- do not forget to switch off what you replace
- do not think cloud migration is a one-off.

Each of these is discussed below in further detail. As so often, it is the detail that can seem to make a nonsense of the simplicity of common sense. But the detail is mandatory.

Know what you have before you start

The attractions of a utility-type cloud for computing are pretty self-evident. The talk is easy. That said, one element frequently overlooked is that cloud computing is based on clever services that are rooted in and based on virtualization — the concept that organizations (or individuals) can have multiple logical 'machines' running on one physical machine. When common server efficiency can be as low as 10-15% (or less), that means 85-90% of the capability and costs of a server is not being fully used. When you have 100 servers of equivalent inefficiency, then this adds up to a large amount of unused capacity — the equivalent of some 85-90 physical machines that you might not have needed to buy (in practice, it is better to use — say — a 50-60% savings factor, to be sure you can cater for occasional peak loads). Add the additional infrastructure (memory, disk, backup, communications, etc. — and people) and an even greater degree of potential wastage becomes apparent.

However, despite a simple basis, it is one of the enduring idiocies of IT that IT does not always know what it possesses (even if there are good, reasonable though not excusable explanations for this). One example involved a major management consultancy that discovered, when it finally looked at itself, that it possessed 30% more servers in each of its N American, European and Asia-Pacific data centers than it thought (never mind what it had elsewhere).

The key lies in knowing what you possess before you start. If you do not know what you possess, you cannot plan accurately.

With the complexity and multitude of today's systems, automated discovery of what you have is the required starting point. Once you know what you have, only then can you start to think forward. Until then you are speculating or living in hope — even if your destination is as nebulous as a cloud.

Decide what you can improve (and what you cannot)

Once you know what you possess then it is time to

decide what you can improve — which should be an iterative process requiring analysis of what you have and then modeling of what you could have. This is as much about management's and an organization's business priorities as it is about technology. Yes, virtualization and cloud technologies can save you an enormous amount (payback on investment is commonly between 6-9 months), but it will only work if there is a business-derived envelope within which cloud computing execution can be defined.

Cloud computing depends on virtualization to deliver its cost, systems and management efficiencies. Before you can contemplate moving to a cloud it is necessary to establish those areas where you want to produce improvements, and those where you do not.

Understand what your savings objectives are and what they entail

For example, having found that you have only an average of 15% server usage (this efficiency measure is most often of CPU usage rather than overall usage, which is still frequently lower in cost-efficiency terms), set a target. This might be as low as 25% (because you wish to be cautious) or as high as 60% (if you are daring). But such targets assist in defining and then deciding what can be virtualized and what should not be — as preparation for a cloud implementation.

Frequent iterations of pre-virtualization modeling enable coherent and comparable RoIs to be prepared for different cloud scenarios. These are essential for decision making.

As important is exploring the implications of the savings objectives. Try to do too much too soon — and a recipe for disaster may be the result. Try to achieve too little and it may not be worth the effort or expense. A business-driven balance is what matters. The technological consequences flow much better when this has been described (and it may be that such a description envisions a number of stages, as progress is made and each stage is seen to happen).

Create a plan that will deliver

In one sense this is the most facile of these Dos and Don'ts. Who would think of moving to cloud computing without a plan?

It is a sad fact that organizations do exist (and will exist) that prepare meticulous analyses and models with exhaustive RoI statements — and then think the savings have been 'achieved'. This is not so much an IT fault as a management one. Because the savings have been identified does not mean that they have been realized.

No RoI should be proposed without, at a minimum, an outline plan describing what is needed to achieve the identified savings. That it is outline does not matter, so long as it can be reviewed for coherence and then expanded if accepted.

Another common fault is to see management authorize the savings shown in the RoI but not make the investment required to make the savings real. By associating the plan with the RoI statement organizations avoid this sort of 'semi-intentional oversight'. The investment required, the savings that will flow and the plan needed to deliver should all be one whole. Be suspicious if they are not.

Measure what you are delivering and review what happens

You can argue that this is no more than an extension of 'creating the plan that will deliver'. If you think this way, fine.

Too many do not set measurable targets that identify if progress and the intended results are occurring. Measure, measure and measure — after you have defined what you need to measure. Reviewing what actually happens against what was intended is simple common sense and just as applicable to a move to cloud computing as to anything else.

Do not permit the 'candy-store effect'

The 'candy-store effect' is a byproduct of virtualization. It is, normally, an IT fault.

Too many in IT appear to think that being able to enable (set up and tear down) new virtual machines (VMs) at will is fun and easy, cheap and without consequences. Without adequate IT management disciplines and tools — which are implemented from the start of entering the cloud — cloud computing can become very expensive if a candy-store mentality ('why not just have another VM') is allowed to take root.

In one organization there were more server VMs than people. Worse, most were not backed up nor was the

data and applications within each VM accounted for properly. This is how to waste what could have been saved. It is easy to avoid (and there are many good tools around to manage and prevent it).

Do not forget to switch off what you replace

In some ways you may think that mentioning this is facile. Nevertheless, it is quite feasible (and many organizations have unwittingly 'accomplished' it) to introduce cloud computing — and forget to remove you have virtualized. Doing this negates your savings.

How does it happen? Quite simply — by not turning off and dispensing with what you possessed previously. Thus servers may remain in data centers. Some may even be left switched on. This is ridiculous.

Put in place an explicit measure objective which shows three things:

- the number of devices turned off
- the number of devices removed
- the ongoing server efficiency %.

If all these figures are not improving, investigate. Otherwise you will be burning rather than saving resources.

Do not think cloud migration is a one-off

One classic error made by both IT and management is to think that virtualization, whether in-house or in a public cloud, is a once-off exercise. It is not. Constant and continuous review of what has been placed in the cloud is necessary to ensure that:

- changing business circumstances are addressed
- further savings are identified
- alternatives are assessed for relevance.

Clouds, and even virtualization, are fashionable. New technologies may, however, appear. Adopting a 'full life-cycle' approach is a necessity if an organization is to continue to reap the cost benefits. This is no different to checking your house annually to see that all is well. It does not have to eat up major resources, but should be periodically scheduled.

One way to ensure this happens is to appoint a cloud advocate plus a devil's advocate. The first should argue, with both financial and technology justifications, the case for continuing with cloud computing. The second should argue the opposite. This establishes a regular process that enables open thinking about both now and the future.

Management conclusion

Cloud computing can deliver savings. It may deliver even more savings than virtualization undertaken in-house. The rationale for either is primarily (but not exclusively — there are plenty of additional justifications — about reducing costs, releasing investment capacity and being more efficient.

To realize these requires discipline. Unfortunately cloud computing, and virtualization, can encourage the opposite. Adopt Mr. Brett's selected 'Dos and Don'ts' and you will avoid reducing what you could have saved.

Charles Brett
C3B Consulting
www.c3bconsulting.com

INSIGHT-SPECTRA

is published and distributed worldwide by:

C3B Consulting Ltd.
19 St. Michael's Road
Winchester SO23 9JE
UK

Telephone:
+44 787 233 4000
+34 686 116 993

Email:
insight-spectra@
insight-spectra.com
or
charles.brett
@c3bconsulting.com

World Wide Web:
www.insight-spectra.com