blue rabbit
sustainable product strategies

insight

The Nature of Data – Volume 2

abstract

The world of data has split into two completely separate pieces! When I say "data" in this sense, I mean the world of Big Data… The world where your bank account lives, the title to your property is, your credit rating is evaluated, your health records are stored, your spending habits are evaluated - and the world in which most Governments, Financial Institutions, Medical, Commerce and Business see the real "you". There is a serious difference between this type of data and the music you download from iTunes, movies you download, emails, photo's of your family that you transmit…

We now have a world of Transient (lightweight, can be replicated) data and Persistent (industrial strength, absolute and definitive source) data. And yet, we are treating them the same. They are NOT the same… they do not cost the same to maintain, or to build, or to protect, or to secure. If you are managing source data – then, by definition, it is Persistent… and you have to take that seriously.

Over the last year, we have been deluged by reports of hiccups, lost data, serious downtime and escalating project costs… it's seemingly epidemic. Not to mention, identity theft is now the highest profile white-collar crime that we all live in fear of. It's time to take the subject of "data" very seriously and appreciate there are clearly two different kinds of data.

We've taken the "lightweight" and "rapid", and "agile" and the "inexpensive" – way too far and, as an industry – we are, in our opinion, too cavalier when it comes to personal data. Transient data and Persistent data are completely different.

This paper is the second in the series and it is intended to explore the meaning of 'data' and our ability to keep it safe and respected. This paper is produced in the hope of creating dialog and some saner approaches to the safekeeping and processing of 'real data'.

the world of data, how it is approached and how it is stored

Over the last 10 to 15 years, we data processing professionals have watched the world split into two very separate markets – Transient Data which belongs on the desktop environment typically ruled and handled by Microsoft software – and Persistent Data which typically belongs on those massive back-office mainframes typically ruled and managed by IBM. I've named companies here but really, I'm referring to machine architectures and ideologies more than actual corporate entities. We in the industry may not have realized it – but there are two completely different kinds of 'data' in our world today.

Transient data is just that. It's transient, can readily be moved from machine to machine, can very simply be replicated and, in the event it's totally destroyed, it can readily be reproduced from source. We see it all the time – pictures, documents, spreadsheets, video, music and even data records like lists of people, lists of buyers, lists of interested email addresses, personal profiles, patient records – even copies of corporate and bank data records. It's a fluid environment with lightweight and agile tools and moving, shifting, shaping and reporting is done by 'magic'.

The people who work with this kind of transient data is an agile-thinker, uses lightweight tools beautifully designed and built by Microsoft and their partners. The information can be put across series of machines and multiples of desktop computers – and actually quite readily passed around as if it were tissue paper blowing on the wind. This is a huge strength of this environment. Internet tools allow me to search for it, copy it, replicate it, and pass it on to others. Development tools allow me to add to it, modify it, delete and change pieces of it, reshape it any way I wish. I can rebuild files in an instant, I can test it out and I can even launch it out to others, both inside and outside my network. In turn they can change it and pass it back or pass it on. If it's broken, or the receiver doesn't quite like it, I can change it again, in an instant – and re-launch it with an email saying, "Here, try this". Time-to-complete, time-to-create, time-to-modify and time-to-delete is measured in seconds.

Transient data - life is fast and the machines and tools are inexpensive and disposable. This environment is full of agile data for agile people, all working in agile companies. Brilliant! It's really fast, it's really inexpensive, it can move really rapidly and it's really cool.

The defining characteristic of Transient Data – its life cycle is measured in months, maybe a couple of years.

Persistent data is completely and utterly different. Much of this kind of data has been around for decades, is carefully stored, has fault-tolerant equipment and processes handling and storing it and it is the definitive source (typically) of everything important in our lives, places of work, places of Government and places of Policing. It is a completely different type of data to that outlined above. It is handled differently, it is modified differently and most times, we keep audits of who has modified the programming, which authenticated-person performed the change on the record-level data and when that change was done. It can even include images of the documents that authorized the change in the first place. This is the land of big-data and 'big-iron'. Persistent data is typically part of the Enterprise, is driven differently, is managed differently and we make some automatic assumptions that make it significantly more expensive. One of those assumptions –this kind of data will live forever.

Automatically, we assume this data is subject to Governance and to Regulatory Compliance and equally automatically, we assume it will be around either in its live form (available for machine processing) or in its archive form (locked up for safe archival storage and potential retrieval) for many decades, if not, the 'life' of the person (considered to be 75 years) – sometimes longer.

Similarly, the people who look after this data are long-term thinkers, use heavyweight industrial-strength tools, have many years of experience under their belt and spend their lives protecting the data against either malevolent or innocent acts of change. Any change is considered permanent – there is no "UNDO" button in this world, everything and every act against the data is captured prior to its being processed. New processes are built, meticulously tested in isolation on test-data sets, tested again against volume test-data sets and, prior to launch, tested in parallel to existing applications. Multiple groups of people are involved – systems people, programmers, people who wrote the new system, auditors, accounting groups and even end users. This is horrendously expensive to do; it can take many months to fully test prior and during implementation and is rugged in the extreme. It is not possible to build or modify these systems on the cheap – it just doesn't work!

And yes, it is a serious order of magnitude more expensive – for the creation and the on-going operational costs.

The defining characteristic of Persistent Data – its life cycle is measured in years, potentially decades.

troubles begin where ideologies and skill sets collide

Trouble begins when both communities are assigned inside a project – by management that has no fundamental understanding of what they are trying to achieve or of what skills are relevant to the job at hand.

Like the construction industry, the technology industry has areas of specialty by skill set and by expertise. Just as it wouldn't be prudent to have Electricians testing the structural integrity of brick walls – or Bricklayers testing the safety of your wiring… It is equally as imprudent to have Transient Data/Desktop (IT) people developing and testing your back-office Persistent Data (IS) applications and vice versa.

The technology industry, general data processing, is well over 50 years old. During that time, we've learnt how to handle and protect data, how to build applications so that they could be supported, upgraded and maintained and we've 'gelled' into series of skill sets so that our in-house expertise can readily be applied. Our industry is no longer 'general' – it's relatively specific with specialists working in areas where they have extreme knowledge in certain sets of tools and certain sets of hardware and even in certain sets of applications (Finance and Banking, Manufacturing, ERP, CRM and so on). We have hardware people, we have software people, we have applications people, we have telecommunications people, we have web people and we have a class of people widely referred to as "IT" who, usually certified by Microsoft, run around maintaining the Desktop software and networks and using and applying Microsoft tools to solve everyday problems inside spreadsheets, small databases with lists of data records, documents and other types of Transient Data. Just like any other industry, we have specific skills for specific needs.

When the wrong skill sets get involved with different applications, different databases and differing types of data, then it is a learning experience and, if carried to fruition and implemented, the fallout may or may not be manageable. When it comes to Persistent Data capable of self-checking itself, this can and often is, a show-stopper.

What happens when an IT person is responsible for testing Persistent Data applications? The result is lightweight and it is done in a time period that is measured in hours or days (rather than weeks or even months). They do a quick run-through; they may do a quick system test and then in the vein of "Here, try this…" release it to end users who assume it is complete and fully tested. If this is a changed page on a web site, or a new formula in a spreadsheet then this is more than adequate. If it is against Persistent Data back-office database and application, more than capable of self-checking itself – then everything stops and things begin to go horribly wrong!!

What happens if you have IS Persistent Data veterans build and test? Total overkill! It takes weeks, they build test jigs to run processes through, they understand regression testing, install testing and systems testing and it is done methodically and systematically. They look and protect against both malicious and inadvertent intent. Self-checking routines are inserted and duly tested. When it is finally released, it is rock solid, fault tolerant, capable of preventing run-through and cataclysmic cascading events and it is expected that this new application will run for a period of time in parallel to the existing one and that audits are to be done on resultant data. While this is absolutely necessary for applications that come out of persistent data and into, say, the general public – it is complete overkill if the application in question was housed and stored in a few spreadsheets.

Incidentally, depending on the background of his manager, this Persistent Data person, instead of being valued for the intense skill and experience they bring to the back office systems and 'real data' storage devices – are usually vilified as being slow, paraded as paralytically inept and ridiculed for being old fashioned. So deep is the misunderstanding of the Nature of Data and its' need for persistence that this skill set is usually no longer considered of value to the new and more-agile organization. These people are being laid off or forced into retirement in unprecedented numbers – and the ultimate cost for this will be catastrophic.

The net bottom line; one is a fly-swat killing a T-Rex – the other is a cannon killing a Mosquito! And unfortunately, today's Systems Managers are typically highly articulate; politically correct Transient Data people who would likely see only a pest!

skill sets collide and budgets explode

The two modes of working are completely different and the split is along the nature of data and the need (or lack of need) for persistence. We need to understand this and we need to respect the skill sets involved and have a management team that can adequately assign human resources and intellectual capital to do the right job at the right time. Do it right the first time!

We also need to begin to cost the creation of these systems and the safeguarding of these data sets depending on the Nature of the Data. The greater the need for 'persistence', the larger the cost factor to modify, process and safely store. And it is a matter of scale here, many additional zeros on invoices.

Let's talk about scale - A number of years ago, a local consulting group invited me to a meeting to talk about the, then newly proposed, Government of Canada, Gun Registry System. They wanted to know if, for a budget of C$400,000 – the system could be prototyped. Once we'd got by what was meant by the word 'prototype' (build it as if it was real) – my answer was NO. I didn't think it could be done. Firstly, that amount of money is really only 4 person years, which implies that we'd have to use existing mainframe cycles somewhere (a rare beast at best) secondly, I wasn't sure that you could magic 'real data' processing using only 4 people to create the system. Finally, I suggested to them that, in order to get real costing numbers, we should add up the amount of money that was currently being spent on all the multi Provincial car registry systems (given that it was almost an exact processing replica) and you would have a guideline to at least the operational budget requirements on an annual basis. They told me – that I was completely "out-to-lunch" and that my feedback was entirely inappropriate! In fact, they were clearly quite offended that I should be such a discussion-spoiler.

A number of months later, the same company asked me back and told me that the project budget (for the build sequence) was around C$120 million. That roughly translated (outside needed processing equipment and software licenses) to 190 people to design, build and test the system and about 1000 people to enter virgin data, quite reasonable given the project would last for one calendar year. Still not enough… but it would get them through the early stages, potentially they had missed that the first years running costs had not been allowed for but no problem, I assumed we were talking capital budget and I really didn't want to "rain on the parade a second time".

In the world of Persistent Data – where data is secured over many decades, verified and has its integrity constantly checked, this was not a vast sum.

Again I used the Car Registry as a working example… again I was told that I was out in cloud-cuckoo-land – and that anything can be done for that kind of money. I wasn't asked back to any more meetings. All that to say – putting together systems of this magnitude costs a great deal of money, a vast amount of skill, many processing hours, big machines and serious tools that really care about minutia of sensitive persistent data (like 'record level locking' and multi-phase commit processes that guarantee the level of update and capture all the audit points). Not to mention – a single point of capture that can support upwards of 1000 or more concurrently running input/process sessions. And also not to mention – we'd need at least one "store-front" office for processing population centres in excess of 150,000 people, even given that a web page front-end would be produced. It is a matter of scale. It cannot be done in a Transient Data world that won't scale that fast and that big with that amount of persistence. Not only do we no longer have the ability to cost at this level – we seem to have lost the ability to manage and the ability to build. This was a new system with new people… and it blew the roof off the budget.

In terms of maintaining mature Persistent Data systems - it's easy to see what happens when these two Persistent and Transient Data communities collide – the mindset is completely different. During the Royal Bank of Canada and London Stock Exchange downtime situations, you don't have to be a rocket-scientist to make an intelligent guess at what happened. At enormous expense, they upgraded their systems and somewhere along the line something was missed during testing that, at best, must have been rudimentary. Persistent Data systems are geared to issue a 'halt' if the designed-in checks and balances are not correct (or reach unacceptable variances) – obviously, something issued a halt! It can safely be assumed that the "Here…try this!" approach to testing the new upgrade didn't work.

More than a skill set though, we have a growing misunderstanding on this split inside our own industry. IT people think and act completely differently to IS people and vice versa. It's a question of scale. And it's also not helpful that we now call the old MIS departments by the new IT name!

troubles also begin when persistent data becomes transient

How do you get persistent data into transient form? You produce a report! And then snap those print images to disk and wow… it will go anywhere! Including downloading to a flash-key.

Time was when this kind of reporting (Extract, Transfer and Load – ETL) was a heavily restricted activity and you had to sign for it. Nowadays, most report and query tools will allow you to issue report requests direct against the data and, on authorization, allow the query to proceed – to anywhere. Once persistent data is in a transient form – it can very simply be allowed to escape. Once the "report" is on a Desktop – you've completely lost control, or at least, you have potentially put the data in the hands of a person who is in a position of extreme trust (who may, or may not realize it). There will be more to come on this subject in later volumes.

Persistent data can and often does escape. The minute this data is released, it moves into a world where its rigid security controls are no longer in place. This can be overt and maliciously moved (by someone with knowledge) or inadvertently moved (by someone with no knowledge).

Another example; create a private web page with password enabled access. Accessing data through a browser is the modern way to build applications and put data out into the public domain for update. You can rest assured that privacy is assured through your developers putting a full password protection in place and only those authorized will have access – right? Wrong! If you subscribe to the Google search engine for your corporate web sites and you put the Google toolbar on the top of your browser (at the Desktop browser level) – then you have opened this information up to a Google search (from their equipment, not yours) and now inadvertently made the contents publicly available. Once in the public domain – no amount of protest is going to get it back – there is no DELETE key once stuff is on the Internet. Worse yet – you may have made your sensitive information subject to Google's Clause 11 in the Terms of Service Agreement. You can only hope that your IT staff truly understands these repercussions.

> **Google's Clause 11**
>
> **11. Content licence from you**
>
> 11.1 You retain copyright and any other rights you already hold in Content which you submit, post or display on or through, the Services. By submitting, posting or displaying the content you give Google a perpetual, irrevocable, worldwide, royalty-free, and non-exclusive licence to reproduce, adapt, modify, translate, publish, publicly perform, publicly display and distribute any Content which you submit, post or display on or through, the Services. This licence is for the sole purpose of enabling Google to display, distribute and promote the Services and may be revoked for certain Services as defined in the Additional Terms of those Services.
>
> 11.2 You agree that this licence includes a right for Google to make such Content available to other companies, organizations or individuals with whom Google has relationships for the provision of syndicated services, and to use such Content in connection with the provision of those services.
>
> 11.3 You understand that Google, in performing the required technical steps to provide the Services to our users, may (a) transmit or distribute your Content over various public networks and in various media; and (b) make such changes to your Content as are necessary to conform and adapt that Content to the technical requirements of connecting networks, devices, services or media. You agree that this licence shall permit Google to take these actions.
>
> 11.4 You confirm and warrant to Google that you have all the rights, power and authority necessary to grant the above licence.

An interesting quandary! Great strengths – great weaknesses and even inadvertent irresponsibility carries an enormous penalty.

In conclusion:

1. In our clamber to collect everything meaningful in digital form, we may have forgotten just how sensitive the vast majority of that digitized data can be. And once digitized – how fragile we have made it.
2. We no longer appear to understand or even respect the world of big (persistent) data. Our thinking has been clouded by the nimble and agile Transient Data thinkers who can deliver "solutions", via the Desktop for pennies within hours.
3. Our next-generation IT managers appear to have not noticed that IT and IS skills have evolved. It also appears that speed-of-completion is now more valued than data protection, safety, security and integrity.
4. Once persistent data is transient – exposure control over it is limited at best.
5. Persistent data can become transient in an instant… once transient, it's without controls or respect.

Produced by: Sue Hardman, Principal, Blue Rabbit Consulting Inc.

Blue Rabbit is a consulting company in Ottawa, Canada dedicated to creating winning strategies for technology-based companies.
Telephone: (613) 692-3868 or shardman@bluerabbit.ca