Heads in the Cloud: A look at the business proposition of Cloud Computing

Gregory Travis
Persistent Systems, Ltd.
greg@littlebear.com

PERSISTENT
Partners in Innovation

What is Information Technology?

PERSISTENT

- What do we mean by “Information Technology”?
  - Generally that technology that helps us organize, digest, correlate, analyze, and present information in ways beneficial to human decision making.
  - Automation is implied by the term

Why Information Technology?

PERSISTENT

- To increase human productivity, thereby generating economic growth through increased leisure
  - We don't have to spend as much time thinking because the machines do that for us
    - The definition of productivity increase
  - What do we want? HAL – Heuristically programmed Algorithmic computer)
  - What do we settle for? IBM – International *Business Machines*

Information Technology in the Enterprise

PERSISTENT

- Companies make investments in capital equipment and operational costs only when those investments are reasonably likely to generate returns substantially in excess of the investment amount.
- Examples: Modern facilities, modern equipment, production processes, etc.
- Information Technology is nothing more than a type of capital & operational investment designed to produce returns

Example: WalMart

PERSISTENT

- I used to work at a company that produced a massively parallel relational database for the purposes of decision support.
- At the time, WalMart was a heavy user of Point-of-sale driven decision support tools
  - Product selection for individual stores according to demographic differences
    - Cheeze-doodles and pork cracklin's in the South
    - Potatoes and Beer in the Midwest
  - Product placement in individual stores
    - Beer and diapers

IT and ROI

PERSISTENT

- When we went to sell our system to WalMart, we discovered something extremely interesting
  - WalMart's minimum internal rate of return for any IT capital investment was 100% per annum!
    - Meaning they would not buy any IT system if it was not reasonable to expect that the system would *pay for itself* within the first year of operation.
    - For us, it meant that we had to show a $1.5 million dollar benefit in the first year through decreased costs, increased sales, increased efficiency, etc.
    - It was a real eye-opener

The bottom line

- Corporations view any investment in IT no differently than they view any other internal investment. If the investment in IT can't be expected to return at least their baseline internal rate floor, they won't do it.

Cloud Computing

PERSISTENT

- What is it?
- The coalition and maturation of a number of technologies at a certain time that allows new methods of solving business objectives at a higher rate of internal return.
- It is not a “technology play.” It is a “business play.”

Enabling technologies

PERSISTENT

- Ubiquitous high-speed, reliable, pervasive, and secure interconnection of heterogeneous electronic devices
  - “The Internet”
  - BlueTooth
  - WiFi
  - Near Field Communications (RFID, etc.)
- Consumer electronics (tablets, smartphones,eWallets, etc.)

Enabling Technologies

PERSISTENT

- Distributed storage and replication mechanisms
- Payment processing technologies (PayPal, etc.)
- Processor Aggregation Technologies (MPC, etc.)
- Management Technologies (Windows Azure, OpenStack, etc.)
- Legal frameworks for international issues (emerging)

Technologies

PERSISTENT

- Most of these technologies were developed pre-cloud and to solve specific domain issues.
- But once a sufficient “critical mass” developed, entrepreneurial forces recognized that they could be leveraged together for a new way of looking at and approaching problems.
  - This is what “Cloud Computing” is. Not the technology, but the use of the technology to find new and better ways to solve problems. Meaning increase productivity.

Typical problem domains

PERSISTENT

- Software Delivery (e.g. Software as a Service (SaaS))
  - Simplifies deployment, update, and maintenance
  - Allows licensing on a per-use basis, not per-head
  - Frees in-house IT resources to focus on higher-return tasks
  - Promises platform independence (tablet, PC, smartphone, etc.)

Typical problem domains

PERSISTENT

- Redundancy and resiliency
  - For mission-critical applications, allows an architecture that is both redundant and resilient
  - Move away from the “data center” model to a distributed cloud model. No longer any such thing as a single point of failure or critical asset. “Self healing” is a prime attribute

Typical problem domains

PERSISTENT

- Load-based allocation
  - Enterprises traditionally had to invest to cover the peak usage. I.e. in-house IT systems had to be large enough to handle “worst-case” scenario.
    - Think of a Supermarket parking lot. It’s not sized for everyday, but for holiday crowds at Thanksgiving, Christmas, etc. Most of that resource is wasted the rest of the year.
  - With cloud resources either replacing or supplementing in-house resources, an enterprise needs only buy as much as it needs at the moment

Typical problem domains

PERSISTENT

- Ability to access resources anywhere
- Canonical example is moving consumer data from the desktop to the “cloud.” That way consumer is not tied to any particular location or device in order to get their data
  - iTunes
  - Apple TV
  - Saas (Again)
    - All are moving away from model where you store your data locally to a model where your data is in the cloud and your device simply retrieves it as appropriate

Typical problem domains

PERSISTENT

- Virtualization and multi-tenancy
- Similar to load sharing, but “on the other side.” Means that those providing “data center” services (data, processing, etc.) can use their fixed capital investment to serve multiple customers
  - Can also take advantage of locational factors, such as low electricity costs, high network connectivity and redundancy.
  - Economies of scale translate to lower costs of ownership

Typical domain problems

PERSISTENT

- Acquisition of expertise
  - Complex databases require sophisticated resources
    - You may not want to maintain that resource in-house and may find a better resource at a firm that specializes in it
  - Data security requires sophisticated backup and restore capabilities
    - You may not want to develop and maintain that resource when you can let a firm that specializes in it do it
  - In short, the opportunity to bring expert-level competence to business functions where there would be no justification to develop such competence in-house

Typical domain problems

PERSISTENT

- There is no such thing as a free lunch
- Cloud computing has its risks
- Realities
  - Very difficult to actually eliminate single points of failure in practice. Several painful and spectacular recent incidents (Amazon, Netflix) demonstrate this
  - While many applications are “pleasingly parallel” and easy to scale through aggregation, many “real-world” problems are not

Typical domain problems

PERSISTENT

- E=MC2
  - Also know as “Latency is a *****”
  - For all but the simplest problems, communication delays and overhead can significantly degrade performance and represent a brick wall to scalability through cloud architecture
  - Most prominent example are sparse-matrix computations involving large amounts of communications between nodes.

Typical domain problems

PERSISTENT

- Loss of control
  - Security of your data
  - Availability of your services
  - Opportunity for corporate espionage and loss of intellectual property

Typical domain problems

PERSISTENT

- Legal issues regarding “internationalized” data
  - What do you do if the country where the data center hosting your data is, forbids the kind of data you’re hosting?
    - I.e. you’re a news organization publishing information critical of a sovereign regime. That sovereign regime also hosts the Amazon servers holding some of your data. And they have a law against criticism.

To think about (and discuss)

PERSISTENT

- My thesis is that “Cloud Computing” is not so much about technology itself as it is about using technology to accomplish business / economic objectives
  - There is an argument that the discipline better belongs to the School of Business than it does the School of Informatics
- And my goal has been to make you Informatics students aware of the business drivers pulling cloud computing into the marketplace.

To think about (and discuss) (more)

PERSISTENT

- Note that what I have said here is really only relevant to Cloud Computing in the Enterprise. There is a role, a place, and a function for Cloud Computing within the public sector as well, mostly enabling advanced research that the private sector cannot conduct
  - This presentation would be quite a bit different if I was trying to sell Cloud Computing to the research and academic population.

To think about and discuss (even more)

PERSISTENT

- But many (most) of you, when you graduate, will become part of an IT organization within an existing firm. That firm will either:
  - Produce IT products and services for other firms
  - Use IT products and services for their own production needs
- In either case, you cannot make informed technical decisions without understanding the underlying business objectives and economics

Thankfully, it's not all suits and ties

PERSISTENT

- Most of you, again, are in Informatics not because you secretly want to be in business school, but because, like myself, you love technology.
- Luckily, Cloud Computing is actually (despite everything I've said) a *fertile* ground to indulge in that love.

There are challenges

PERSISTENT

- As I showed in earlier slides, Cloud Computing is no panacea. There are problems that it has trouble solving and there are problems that it cannot solve now.
- The technical challenge is to understand those problems and then build the technology to solve them.

Example

PERSISTENT

- My particular focus is on “big data.”
  - Comes from my database background
- Means the storage, management, and use of very large datasets.
  - Now coming to a cloud near you.

Example

PERSISTENT

- Large datasets in the cloud provide a whole host of new and very difficult challenges
- In particular:
  - Real-time replication
  - ACID (Atomicity, Consistency, Isolation, Durability) requirements
  - Deep correlations (think database joins)
- These are hard things to do when the data is located on a single server and a single disk drive
- In the cloud, they can be intractable

Example

PERSISTENT

- Not all cloud applications will have this problem, not even all “big data” applications
  - There are applications which are easily parallelized/stovepiped
  - There are applications which do not require real-time data updates (i.e. are read-only)
  - There are applications which can tolerate (temporary) data inconsistencies or update latencies
  - And then there are the others…

Example

PERSISTENT

- Customer has a high-volume online application that performs payment processing
  - A big field these days
  - Think PayPal (even if I am not)
- Current architecture involves a single “primary” site and multiple secondary backup sites. Very traditional.
- Scalability, reliability, manageability severely constrained by current architecture

Example

- At first glance, this would appear to be an ideal candidate for cloud-ification. And it is.
- But there are challenges
- Before we talk about that, let's talk about the "happy path" -- the ideal.

Example of the ideal

PERSISTENT

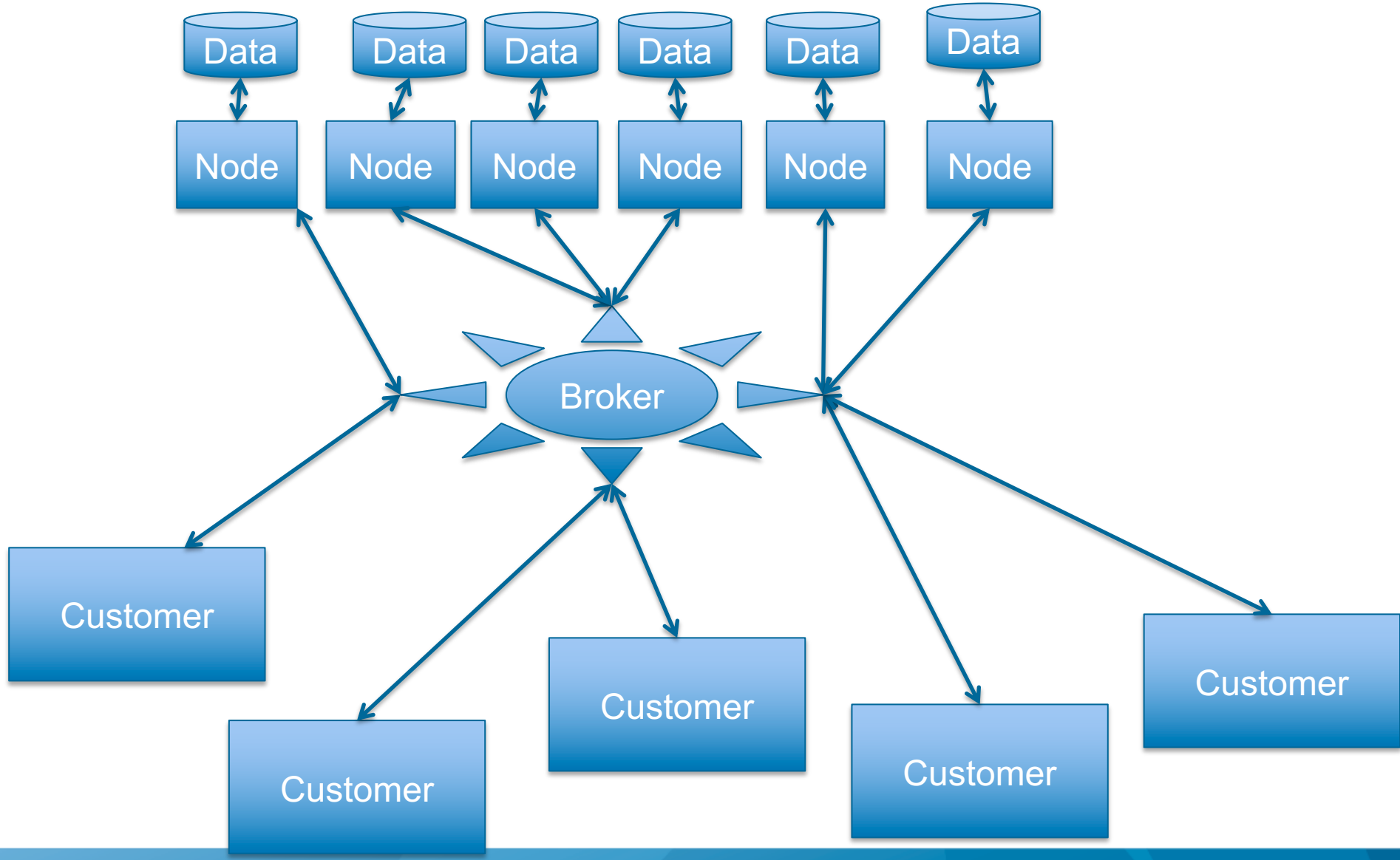
- Any customer can be serviced by any “node” in the cloud. Nodes are fungible and scalable.
  - Node failure does not affect system operation
  - Addition of nodes increases capacity
  - Nodes can be geographically located anywhere
  - Complete elimination of “primary/secondary” sites. Indeed, complete elimination of the “site” concept at all. Things are just out “in the cloud.”

Problems

PERSISTENT

- This is relatively easy to do with pleasingly parallel applications where individual users/customers can be siloed into a partition
  - Just need a broker (a single point of failure!) to direct any customer to an available node. Node can then process customer request independent of any other node.

Looks like

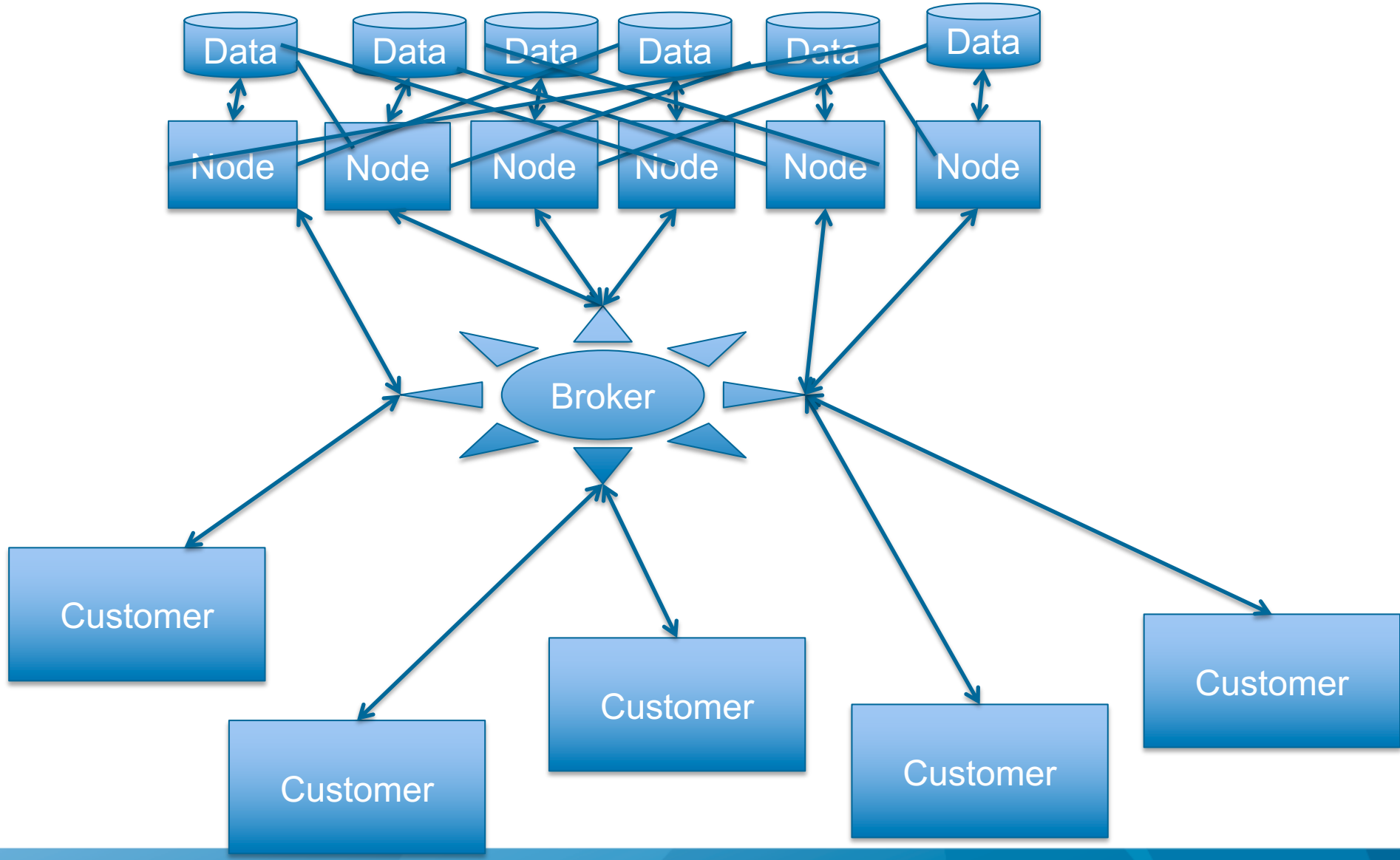


But it's not so easy

PERSISTENT

- Our application doesn't lend itself to that architecture
  - Customers are doing reads and writes to process payments
  - Customers are dependent on other customer's data (customer A can pay Customer B can pay Customer D can pay customer A)

Looks like (highly simplified!)



All of a sudden, things just got hard

PERSISTENT