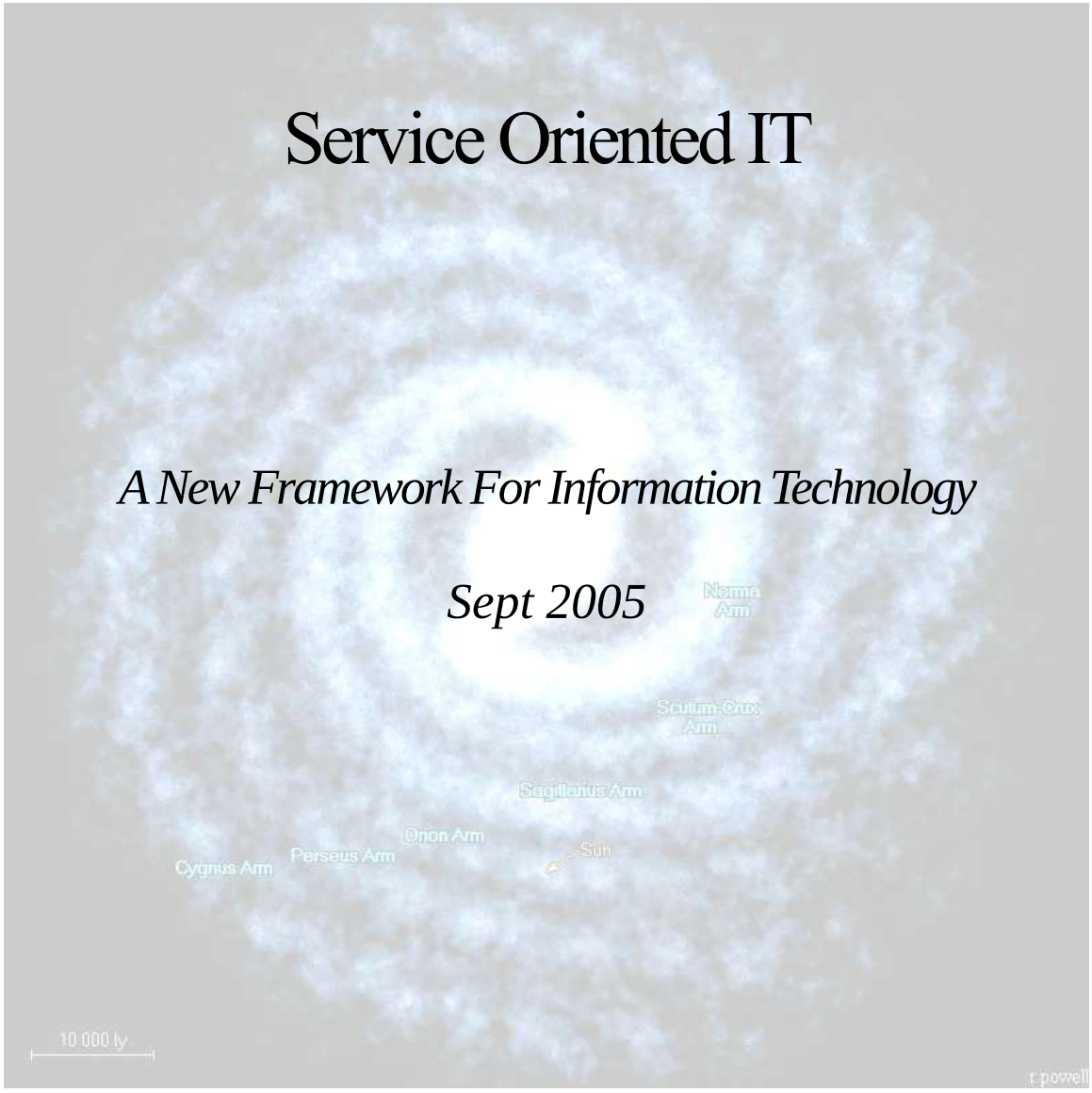
The Lanigan Group White Paper

www.lanigangroup.com
(613) 277-5898

Service Oriented IT

A New Framework For Information Technology

Sept 2005



The Lanigan Group Inc.
14 Cypress Gardens
Ottawa, Ontario
Canada
K2S 1W5

Author

Paul Renaud

renaud@lanigangroup.ca

September 2005

Publisher

Printed in Canada.

Disclaimer

The Lanigan Group Inc. has made every reasonable effort to ensure that all information in this document is correct. We assume no responsibility for any inadvertent errors and this document is published WITHOUT WARRANTY OF ANY KIND whatsoever. The reader assumes full responsibility for acting on any or all of the information contained in this document.

Any trademarks, registered or otherwise, that may appear in this document are the property of their owners and not of The Lanigan Group Inc.

Executive Summary

The architectural characteristics of the SOA paradigm are:

- *Architecture Idempotence* – enabling transition;
- *Resource Virtualization* – enabling higher levels of asset utilization;
- *On-demand Provisioning* – enabling the ability to pivot resources to meet demand;
- *Service Virtualization* – enabling productivity improvement; and
- *Autonomic Management* – reducing the cost of IT.

The new reference model for computing comprises five new layers between applications and the networks that host them as services:

- *Admission Control* – managing secure access to the environment;
- *Workload Management* – optimizing work according to service levels;
- *Execution Management* – providing a virtual execution environment for application services;
- *Cluster Management* – sharing resources cost-effectively;
- *Resource Provisioning* – providing autonomic control over both physical and logical resources.

This reference model can be used to understand the scope of the new products and standards that will emerge in the new millennium; just as the 7-layer ISO reference model for networking helped frame and position communication technology in the previous millennium.

Transition to the new paradigm can occur incrementally as both applications (generating demand) and computers (containing resources) can be migrated step-wise into the new environment.

.

Service-Oriented IT

The Challenge

The SOA Imperative

The latter part of the 20th century was characterized by wholesale networking of computers motivated by Internet access. Networks were standardized on the Internet Protocol and intra-nets were deployed within enterprises. During that period we witnessed a breakthrough in *information access*; paralleling the Internet's information richness with enterprise-wide information accessibility.

The critical success factor in this information access breakthrough was the triple play of client/server computing, Web browsers and application servers. Front-ending legacy databases with application-server based software unlocked the benefits of the client/server paradigm without the drawbacks of having to deploy and manage heavy-weight desktop applications. This created a 10:1 asset leverage factor since information technology (IT) departments could operate a few hundred application and database servers far more cost-effectively than supporting applications running on several thousand desktop machines.

Unfortunately most IT organizations can no longer continue to leverage the growing speed of computers into increased productivity. Application servers are bloated and bottle-necked. Compounding the situation is the fact that many IT groups purchase and manage a complete set of servers for *each* application. Typically each new application requires its own primary and backup application server, separate production and test database servers, and one or more development servers. That equates to at least 4 computers per application! With so much hardware employed, return on asset utilization in most data centers is typically less than 25%.

Reaching breakthroughs in productivity and cost-effectiveness requires a new computing paradigm based on a service-oriented architecture (SOA). To this end, leading IT organizations have started to deploy SOA in the form of portals, grids, and web services.

The SO-IT Challenge

Deploying SOA to meet the demands of a coherent set of applications within a single line of business, however, is far more straightforward than scaling the entire information technology fabric into this new paradigm. Rationalizing IT resources, applications and information across many lines of business requires transformation of IT business practices as well as IT infrastructure. IT organizations must ensure that the prioritization and allocation of the entire IT fabric is aligned with the enterprises' business vision, priorities, goals, and objectives.

IT governance-driven IT organizations are trying to re-invent themselves as more responsive, client-oriented, organizations. To fully maximize IT investment, leading IT organizations need to transform themselves at both a business and infrastructure level into being more service-oriented. This service-oriented IT challenge requires not one, but three critical success factors:

1. IT infrastructure must become more general purpose so that it can serve the needs of multiple lines of businesses. This necessitates the use of SOA as the blueprint for IT infrastructure;
2. IT applications must become more re-useable so that they can be fully leveraged within an SOA paradigm. This necessitates the use of a re-usability framework as the blueprint for applications;
3. IT organizations themselves must become more service-oriented, applying IT governance practices to better align with line of business objectives. This necessitates the use of a service provider paradigm for IT management.

SO-IT Transformation

Transformation Framework

Since these three critical success factors have overlapping implications on IT, a transformation framework is required for successfully effecting the transformation of IT.

Characteristics of SOA

Idempotence

The architecture of network-based computing is fundamentally the same as the architecture of a single computer. This idempotency[1] of computing architectures is readily seen in the following illustration:

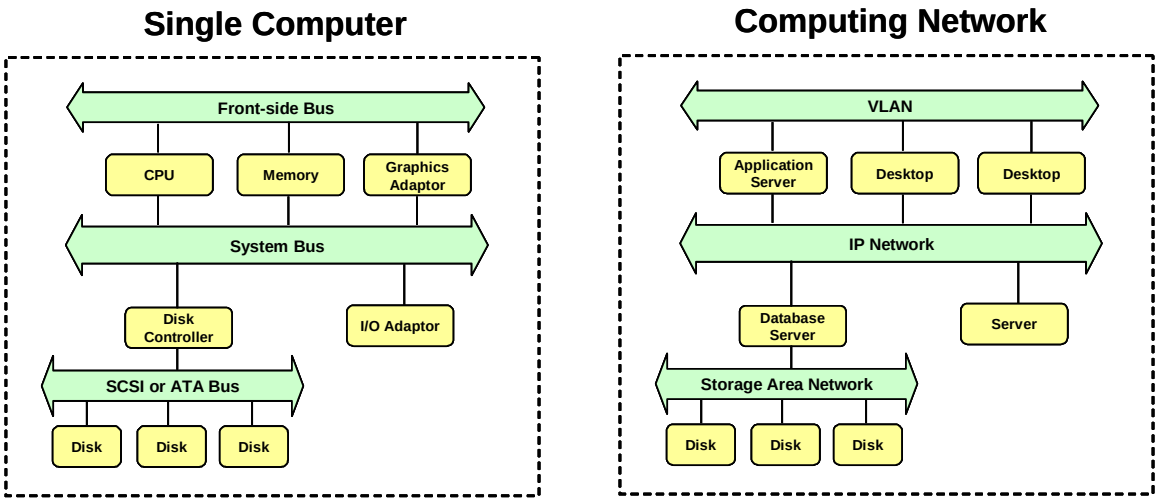


Figure 1: Idempotent Architecture

This property was first documented over 10 years ago by [Renaud, 1991] and its implications are profound. Fundamentally every mechanism used to operate and manage a single computer has a corresponding virtual representation in a networked environment:

- Files are represented by network-attached storage (NAS) or storage area networks (SANs).
- File Systems are represented by network-addressable databases and network file systems.
- System buses are represented by high-speed local and wide-area networks. In many cases, the speed of the network is faster than the speed of the system bus in the computer! Even the intimate interconnect between CPU-memory-graphics controllers is represented by virtual private LANs (VLANs).
- Central processing components (CPU, memory, graphics) are represented by entire computers.
- Operating systems are represented by distributed resource management middleware.
- Application programming interfaces (APIs) are represented by web services interfaces.

[1] An element that remains unchanged when multiplied by itself is considered to be idempotent. A computing network is a multiplicity of computers interconnected by a network.

- User interfaces are represented by HTML-enabled web browsers. (The evolution of the desktop metaphor in Windows into an embedded browser metaphor is a good example of this.)
- Management consoles are represented by network operations centers.
- And so on.

The major benefit of architecture idempotence is that it provides an evolutionary path from the existing paradigm to the new architecture. Familiar functions still occur, however, the tools to perform them must evolve. Similarly, applications can be migrated to the new paradigm without encountering a discontinuity in how they should be designed.

Resource Virtualization

The keystone to understanding SOA is *virtualization*. Networks enable us to do much more than simply connect machines together. Interconnected machines can co-operate to deliver an end-to-end service across their connection. The resulting service is *virtual* because it does not reside in a single machine. Where is the location of a long-distance phone call?

Just as the public switched voice network (PSTN) engages multiple telephone switches and two telephone sets to create an end-to-end long distance call, computing networks engage browsers, application and database servers to create an end-to-end network service. Hence in a networked computing environment, applications become network services.

In fact all resources can be virtualized in a network: software, databases, disks, communications gateways, etc. The network accomplishes this by hiding the actual location of the resource behind a network-wide address. You don't need to know where the resource actually is; to use the resource all you need is its address.

The magic of a long-distance phone call lies in the ability of the PSTN network to resolve an E.164 address (commonly known as a "phone number") into a voice connection. Similarly, the true power of the World-Wide Web comes from the Universal Resource Locator (URL). The magic of the Web is that all its destinations are a click away; located by URLs. Leveraging the Web paradigm all resources can be virtualized by a Universal Resource Identifier (URI). URLs are simply a special case of URIs that happens to reference HTML pages.

Virtualizing a resource has many operational advantages:

- The resource can be shared among users; substantially improving its utilization. Depending on the resource, this sharing can be concurrent or sequential in nature.
- The location of the resource can be changed without notifying all the users of that resource. This provides tremendous flexibility in configuring the IT infrastructure.
- The resource can be upgraded, re-provisioned, or reconfigured without affecting all the users of that resource. This facilitates management of resources since individual resources can be managed independently.
- Multiple instances of the resource can be provisioned and allocated to users on-demand. This provides substantial scalability of the resource.
- A damaged resource can be rapidly restored or replicated to a different location; substantially accelerating recovery of service. Virtualized resources result in much higher availability of service than directly accessed resources. High-availability virtualized resources are also more cost-effective because they can be provisioned using N+1 sparing strategies. For example, why allocate a separate standby application server to each application when the same server can be shared among all applications?

On-Demand Provisioning

Not only can you virtualize any resource in a computer, you can virtualize the computer itself. When a collection of computers are virtualized into a common computing fabric the result is *grid computing*.[2] Applications are

[2] The term "grid" arises from the analogy with the electrical grid which also creates a common fabric of power.

provisioned and launched onto this fabric just as they might be on a single computer. The grid environment however, optimally distributes applications to the best-available computer at the time that the application is required.

To cost-effectively leverage a computing grid, all computing resources should be as undifferentiated as possible. Automated provisioning software can be used to cost-effectively configure these computers as required to service the unique needs of different applications. For example, different applications might require different versions of system or support software installed, or even different operating systems.

Using on-demand provisioning software, such as Opsware or IBM's Tivoli Intelligent Orchestrator, individual computers within computing grids can be pivoted into different roles to provide the capacity to meet changes in workload mixes. For example, online services may require the majority of the provisioned capacity of the grid during the business day but be scaled back during off hours to provide more headroom for batch processing. Or system failures may cause other systems to be automatically reconfigured on-the-fly to ensure that service levels are maintained for mission-critical applications.

Service Virtualization

Most examples of virtualization involve the *supply-side* virtualization of resources for the benefit of an application that may, or may not, itself be network-based. Even if the application is network-based, as in the case of an application running within a J2EE application server, the application itself may not necessarily be virtualized. The application may simply be the consumer of virtual resources.

However, when the application itself is also virtualized we enter the realm of *demand-side* virtualization of processing. The simplest example of this occurs when load-balancing is used to distribute transactions (or IP traffic) to multiple application instances. Other common examples of demand-side virtualization are the distribution of transactions among publishers and subscribers in a message queue; two-phase commit transactions in a distributed database; the network-wide scheduling of batch jobs in a compute farm; and content cache managers in a broadband network.

Since applications (acting as surrogates for users) generate the demand for resources supplied in a network, virtualizing an application results in *a virtualization of the demand for resources*. This means that workloads can be optimally allocated within the network – greatly improving performance, throughput, and completion rates. In effect, demand-side virtualization manages the added complexity of deciding where work should occur as well as which resources should be consumed.

Autonomic Management

Since applications can also invoke the services of other applications, a new generation of virtualized application services can be provided as building blocks to incrementally create new network-enabled applications. Using web services protocols it is now possible to rapidly develop new applications that leverage component modules that are provided as reliable, highly scalable virtual services.

Ultimately constructing new applications out of virtual services will drive the evolution of a new class of distributed autonomic management software. Consider the case of using software components, such as Java beans, in an application server. The calling application invokes the component which is loaded and run in the same application instance as the application server (or possibly load-balanced into another application server instance in the same cluster). Using web services, the called component can now reside in a completely separate application server instance (or cluster).

However, due to the limitations of the current generation of application server software (provided by Web Sphere, Apache, or Web Logic, etc), this distribution of work is largely unmanaged since it spans the jurisdiction of more than one application server instance.

As a result, the next generation application server will itself be a virtual service. As a virtual service it will be able to provide distributed management of workload across a grid of computers. Examples of this next-generation of distributed workload management can be found in GridServer from DataSynapse and LSF Symphony from Platform Computing.

These next-generation application servers will marry on-demand provisioning with distributed workload management to manage both demand-side as well as supply-side virtualization. They will also incorporate other autonomic management functions such as:

- *Performance Monitoring*: both distributed workload management and on-demand provisioning require real-time monitoring of performance indicators to determine how to optimally allocate workload and resources under peak loads.
- *Self-Healing Recovery*: congestion control, detection, containment, and recovery from catastrophic hardware / software / network failures, runaway processing, as well as transient glitches.
- *Security*: access to distributed resources must be mediated to assure that only authorized workloads can consume or acquire them. Integrity of the distributed environment depends on the prevention, detection, and recovery from malicious usage.
- *Accounting* for the usage of resources to ensure that the environment can be operated cost-effectively using activity based costing of resource consumption by workload, users, and lines of business.

SOA Reference Model

Reference Model

Enterprise IT architectures will evolve into 21st century computing environments by layering virtual services on top of virtualized resources as illustrated in the following reference architecture:

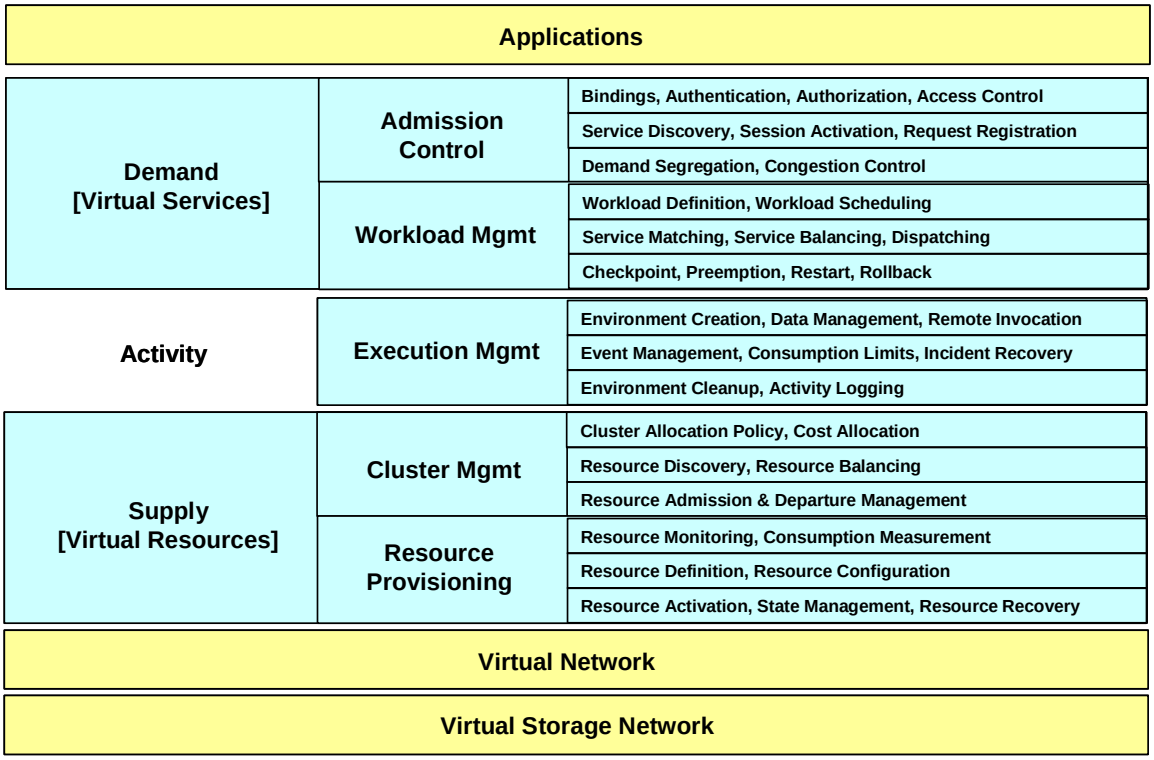


Figure 2: Reference Architecture

By balancing supply and demand, the resulting virtual computing environment will provide an order of magnitude improvement in cost-effectiveness and productivity.

Note that this reference architecture is not a description of a specific implementation or implementation specification. The purpose of the reference architecture is to facilitate an understanding of the functions required as well as to facilitate the scope of specific implementations or standards.

For example, the implementation specifications produced by the Global Grid Forum generally addresses the Supply and Activity levels of the reference architecture but not the Demand level. Meanwhile, specific implementations such as Platform's LSF may address multiple levels in the reference architecture but only for specific types of workloads such as batch jobs.

Admission Control

The first layer in the reference architecture is concerned with managing access to the virtual computer. This ranges from the various mechanisms used to securely access the virtual environment (API bindings for example), to locating and invoking available services.

Some implementations may also support request reservation in addition to request registration and submission. Others may also support admission to federated virtual computers (i.e. the ability to exchange workload between independently managed virtual computing environments).

Ensuring that demand is properly segregated between user communities and that traffic control is enforced during peak periods is also integral to production-quality admission control.

Workload Management

This layer matches demand against the best available service based on workload priorities and resource requirements. Workload management typically involves two phases: workload scheduling (i.e. deciding what work to run next) and service instance matching (i.e. deciding where to run it). Service instance matching is also commonly referred to as *dispatching*.

Various scheduling policies may be supported depending on the implementation including calendar, workflow, and event-based triggers. Similarly various service matching policies may be supported including:

- Resource affinity (i.e. select the service instance that has the best overall match to the resources requested),
- State matching (i.e. invoke this transaction on the same service instance as the previous transaction for this user), and
- Performance matching (i.e. invoke this service on the fastest available computer), etc.

This layer also provides the necessary support for stopping/resuming, rescheduling and recovering workloads as required to assure that overall service levels are maintained across all workloads.

Execution Management

This is the layer where virtual services are actually executed. In addition to remotely invoking the requested service, this layer provides the necessary support services to create a virtual execution environment regardless of physical location (within the constraints of available resources). This includes pre- and post-execution environment setup / cleanup as well as exit status monitoring.

Depending on the implementation it also provides data management services such as input/output redirection, transaction support, file caching and pre-loading, checkpoint support, and application log services.

It also provides support for monitoring, activity logging, signal and event management, error propagation (i.e. notification to clients of the service), incident reporting and recovery, as well as threshold limits for resource consumption.

Cluster Management

This layer manages the overall supply of resources as clusters (or pools) of similar resources. Each cluster defines a scope within which resources are shared. Resources are discovered, admitted and balanced among clusters.

The allocation of resources to clusters may be governed by resource allocation policies and, depending on the implementation, costing policies may also be used to chargeback the dynamic borrowing of resources between clusters.

It also provides cluster usage metrics for performance analysis and usage-based billing. Some implementations also provide placement advice to the Workload Management layer (such as performance metrics used to determine loading and resource availability).

Resource Provisioning

This layer provides the services to manage the lifecycle of resources. This includes defining and configuring resources, resource activation, controlling their state, and monitoring their usage.

For example, if the resource is a software service that must be initiated based on a predefined provisioning schedule, this layer would be responsible for pre-loading the service so that it can be used by the execution management layer.

Depending on the implementation, dynamic provisioning functions may be used to re-purpose resources based on business policies or events such as time-of-day, or catastrophic recovery.

Aligning IT With Business Priorities

Although moving to this new paradigm can be done incrementally on an application by application basis, the benefits grow nonlinearly as more application services are virtualized.[3] Once the installed base of virtualized services reaches a critical mass, new generation applications can be rapidly introduced by leveraging existing virtualized services.

References

Books

[Renaud, 1991] Renaud, Paul E., *Introduction to Client/Server Systems*, John Wiley & Sons, New York, NY, 1991

White Papers

[Lanigan Group, 2004] The Lanigan Group, *Virtual Computing*, White Paper, August 2004

[OGSI, 2003] Global Grid Forum, *Open Grid Services Infrastructure (OGSI),* GFD.15, June 2003

[3] Metcalfe's Law states that the value of a network grows in proportion to the square of the number of nodes in the network. Renaud's Corollaries state that (a) the value of a network service grows in proportion to the value of the network and (b) the value of a virtualized service grows in proportion to the value of the network times the resource intensity of the applications that use it.

About The Lanigan Group

The Lanigan Group provides strategic market engineering services to the technology industry in both the USA and Canada. Founded in 1998, The Lanigan Group creates value for technology by optimizing business strategy, product strategy, value propositions, and go-to-market tactics. The Lanigan Group also provides business development assistance in Asia, the USA and Canada. Clients of The Lanigan Group include major carriers in Canada and the USA, communication product companies, and IT systems management products and services companies.

The Lanigan Group combines state-of-the-art marketing and management science methods with engineering-style discipline to achieve superior results for its clients. More information regarding The Lanigan Group may be found at www.lanigangroup.ca.

About The Author

Paul Renaud is an international expert in distributed computing with over 20 years of experience spanning computing and networking products, services and IT management. Mr. Renaud pioneered the successful use of client/server computing in the 1980s while he was Chief Architect at SHL Systemhouse Inc.; acclaimed by G2 Research as North America's premiere client/server systems integrator. Mr. Renaud then moved to Nortel where, as Director of Computing & Networking, he led the IT transition from centralized to distributed computing as well as championing electronic security. Many of the innovations fostered by Mr. Renaud to manage Nortel's 60,000 node distributed environment have subsequently become standalone companies (including Entrust, Platform Computing, Linmore – now part of Nuvo Networks, Sygniant). Mr. Renaud then became Director of Nortel's Advanced Computing Research Lab where he led research in the management of service-oriented distributed architectures. In the mid-1990s, Mr. Renaud left Nortel to become VP of Development for Cognos' distributed business intelligence products. During his tenure at Cognos he played a pivotal role in driving the growth of the Business Intelligence product line while his products won numerous awards from PC Week, Datamation, and industry analysts. Since his founding of The Lanigan Group in 1998, Mr. Renaud has helped his clients successfully launch 3 distributed computing products and has advised Sprint Corporation's executive on the strategic implications of the internet on their business.

Mr. Renaud's best-selling book, *Introduction to Client/Server Systems*, published in four languages by John Wiley & Sons, is still used as a textbook in universities over a decade after its first printing. He is the former chairman of the Canadian national POSIX standards group, a founder of the grid computing industry's New Productivity Initiative (now Grid Forum), and has been cited in ISO and IEEE standards on open systems and software engineering. He is currently a member of Queen's University's innovation council.