MASSEY
UNIVERSITY
TE KUNENGA KI PŪREHUROA

# ACHIEVING BUSINESS @ THE SPEED OF THOUGHT

**This thesis is presented in partial fulfilment of the requirements for the degree of**

**Master of Information Sciences in Information Technology**

**School of Engineering and Advanced Technology**
**At Massey University Albany, Auckland, New Zealand**

**By**

**Matthew Joseph Alexander Comb, PG Diploma in Information Sciences and BSc Mathematics and Information Sciences**

**February 2016**

# Abstract

Almost two decades ago, the business community touted Bill Gates' book *Business @ the Speed of Thought* as visionary. Gates laid out his blueprint for how the immediate propagation of relevant information from network to network and business to business via a digital nervous system would change the way every business operated. The value of immediately being able to access timely and pertinent information was highly desired by any business seeking a competitive advantage.

Fifteen years later and Gates' vision has yet to be realized. Organizations have difficulty integrating with one another at both a technical and business level, and rapidly changing environments are problematic when trying to maintain resilient business process integration.

This raises some key questions. Why is enterprise integration so difficult? What are the barriers to enterprise integration? Can identified barriers be overcome? Ultimately, we need to know if the digital nervous system analogy is realistic or if there is a maximum achievable level of integration and cohesion.

This report investigates approaches to integration, compares them against current attitudes and practices within enterprise organizations, and takes valuable steps towards confronting realistic integration expectations, and outlining future areas of research.

# Acknowledgements

I would like to thank Dr David Parsons and Dr Paul Watters for reinvigorating my passion for information science. Both are passionate about information science and provide complementary viewpoints that have tested the boundaries of my knowledge and encouraged me to go beyond. Their collective support throughout the master's programme was gratefully received.

Dr Parsons' common-sense approach to technology is an acknowledgement that context changes often and that a sound solution in one context may not be so in another. I found his perspective on the evolution of information science refreshing and thought provoking, and will always consider his viewpoint when designing solutions in the future.

Dr Watters' breadth of knowledge is exceptional and the challenges he set me throughout my research were stimulating. I admire the innovative, proactive and business-savvy approach he brings to academia and he inspires me to continue bridging the divide between academia and the commercial world so both may flourish.

A huge thanks to Dr Kristin Stock who became involved late in the process. In the short time I have known Dr Stock, I have been impressed with her efficiency and look forward to working closely with her and learning from her throughout my PhD programme.

I would also like to thank my business partner and mentor Laurence Day whose friendship and guidance through tough times have been invaluable. Laurence not only wants the best for New Zealand, but also actively and generously contributes to its future. I hope to continue following in his footsteps and make a lasting and positive change for information technology in New Zealand.

I would also like to acknowledge Greg Bolton, Paul Harrison, Mark Berry, Maurice Verheijen, Campbell Elliot and Mike Ogle — a group of close friends who have encouraged me to further my education and to realize my dream of transforming the New Zealand knowledge economy.

Thanks to Michael Smith for his editorial assistance, and Ron Holl, graphic designer, who both helped to polish the thesis into shape.

Finally I would like to thank my parents Barbara and Joseph Comb and supporting family who — despite their humble backgrounds — made numerous sacrifices to ensure our family grew up on the North Shore of Auckland and received a quality education. I hope that in recent years I have made them proud.

# Table of Contents

# List of Figures

# List of Tables

# List of Dimensions



# List of Principles

# 1

# Introduction

Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In this chapter we will present the concept of integration at the speed of thought as well as related background information, and identify the key research questions this thesis intends to address.

## 1.1 Introduction

In 1999, the Internet community touted Bill Gates' book *Business @ the Speed of Thought* as visionary. Gates laid out his blueprint for how the immediate propagation of relevant information from network to network and business to business would change the way every business operated. The value of immediately being able to access timely and pertinent information was highly desired by any business seeking a competitive advantage.

Part of Gates' vision was the establishment of the concept "The Digital Nervous System" in which he compared electrical impulses in an organism to information flowing through Internet channels, capable of triggering events, which would allow for business response. This analogy had previously been established and communicated by a number of technologists with Brian K. Seitz first terming the description in his paper "The Cybernetic Factory — the next generation in CIM" in 1987 ("Digital Nervous System", 2015). Gates, however, went further with the definition stating that the nervous system assists decision making by providing business organizations with the information they need and highlighting the most pertinent and valuable information that would promote business software integration (Gates, 1999).

He went on to assert: "Companies need to have that same kind of nervous system — the ability to run smoothly and efficiently, to respond quickly to emergencies and opportunities to quickly get valuable

information to the people in the company who need it and the ability to quickly make decisions and interact with customers" (Gates, 1999, p. 8).

The revolutionary notion that Gates brought to the table was the ability to take the abstract concept of 'The Digital Nervous System' and communicate its application in a practical sense to the general business market (Gates, 1999). He outlined a set of 12 rules for businesses to follow to achieve the Digital Nervous System (Table 1).

| | |
|---|---|
| 1. Insist that communication flows through the organization over email so that one can act on news with reflex-like speed. | 7. Create a digital feedback loop to improve the efficiency of physical processes and improve the quality of the products and services created. Every employee should be able to easily track all the key metrics. |
| 2. Study sales data online to find patterns and share insights easily. Understand overall trends and personalize service for individual customers. | 8. Use digital systems to route customer complaints immediately to the people who can improve a product or service. |
| 3. Use PCs for business analysis, and shift knowledge workers into high-level thinking work about products, services, and profitability. | 9. Use digital communications to redefine the nature of your business and the boundaries around your business. Become larger and more substantial or smaller and more intimate as the customer situation warrants. |
| 4. Use digital tools to create cross-departmental virtual teams that can share knowledge and build on each other's ideas in real time, worldwide. Use digital systems to capture corporate history for use by anyone. | 10. Trade information for time. Decrease cycle time by using digital transactions with all suppliers and partners, and transform every business process into just-in-time delivery. |
| 5. Convert every paper process to a digital process, eliminating administrative bottlenecks and freeing knowledge workers for more important tasks. | 11. Use digital delivery of sales and service to eliminate the middleman from customer transactions. If you're a middleman, use digital tools to add value to transactions. |
| 6. Use digital tools to eliminate single-task jobs or change them into value-added jobs that use the skills of a knowledge worker. | 12. Use digital tools to help customers solve problems for themselves, and reserve personal contact to respond to complex, high-value customer needs. |

*Table 1: Bill Gates' 12 Rules for a Digital Nervous System (Gates, 1999)*

Like any vision, Gates' blueprint had its supporters and critics. However, his view was largely well received as it accepted and extended previously established visions including the service-oriented architecture (SOA) approach to integration, established in 1983. And he added business pragmatics and detailed implementation steps, defining the approach required to achieve the vision.

The objective of Gates' set of Digital Nervous System rules was not to define the end game, but rather to mobilize organizations to be prepared for a world containing not only a nervous system, but collective intelligence as well (a brain). As such, the rules provide action points for organizations to ready themselves, to ensure they are digital and capable of interoperating in a connected world.

Today, businesses, for the most part, have transitioned smoothly to "Digital". The majority now use productivity tools such as Office and, in recent times, "cloud" infrastructure has enabled remote access to online services. Furthermore, a subsequent ripple effect from this increased information availability

has influenced company culture including a trend towards dynamic team formation and the motivation for staff to complete tasks in a faster, more efficient manner.

One aspect of the Nervous System analogy, however, remains problematic. While a nervous system transfers and responds to electrical impulses in fractions of a second, we still live in a world where it is difficult and time intensive to share information between systems.

Therefore does this analogy really make sense? Is it possible to create a global network that is responsive to this degree? Research suggests that in human beings sensory information is computed extremely quickly with neurons needing less than 30 milliseconds to mediate perception (Tovée, 1994). Is it conceivable that the Internet could operate and respond this fast?

The speed of transfers on the physical layers of the Internet — assuming continuing expansion of high-speed networks — is not a barrier to achieving such responses as network latencies are at this rate now and continue to drop. Thekkath and Levy identified, however, that "Control" logic for remote procedure calls is the greatest overhead for response times. These control pieces are the logic and business intelligence segments that exist in online systems (Thekkath & Levy, 1993).

Fifteen years on, and despite some progress, Gates' vision remains largely unrealized. This raises the question: why so little progress? What are the barriers to Enterprise Integration? In addition, what must change for information to flow seamlessly between organizations as quickly as nerve impulses travel through the human nervous system?

## 1.2 The Information Superhighway Myth

Just as the definition and expectation of the Digital Nervous System has evolved, so has the definition of the Internet superhighway, loosely accredited to Al Gore (Calvert, 1997; Cooper Berdayes & Berdayes, 1998). One early explanation defines the superhighway metaphor as the approach to modernize delivery of content by moving from a narrow to broadband model where volumes of information can be transferred in a short period. Specifically the approach to digitize content meant it could be transmitted more readily (Kenway, 1996).

In today's operating environment, the concept of the 'information superhighway' should perhaps be differentiated from a 'data superhighway'. When data is processed, organized, structured or presented in a given context, it becomes information. This establishment of a context, however, places additional requirements on the platform that serves it, including session management and user access control.

Raw data on the other hand, lacks a rendering context and can flow more freely with additional transport options. For example, peer-to-peer protocols are the most active protocols on the Internet today with over 50% of all Internet traffic belonging to P2P (Cisco Systems, 2009). Additionally, Netflix takes up a good portion of net traffic, which could be described as data with a small bit of metadata masquerading as information.

Whether it be information or data integration, it can be said that the emerging Internet provided the perfect catalyst for a superhighway approach. However, as with many early-adopted movements, there was a boom period and a land grab for digital solutions as tools and technologies were in their infancy. The automotive industry ended up with different geographic locations driving on opposite sides of the road; in engineering, multiple measurement systems emerged; and the Internet's initial developmental period resulted in a very fragmented market of tools, technologies and solutions — as discussed by W. Wayt Gibbs (1994) in his reference to Brad J. Cox in the article *Software's chronic crisis.*

As with any first-generation product, there are inherent limitations — none of which have governed possible information solutions more than the constraints of the network itself with lack of speed and

connectivity affecting the realistic implementation of system architecture and therefore integration solutions (Gligor & Teodoru, 2011).

Surprisingly this fragmented approach has not only continued, but many of the modern technologies we have in place today have promoted it.

Open-source software, service-oriented architecture, web service development and supporting technologies have brought many well-received freedoms. They have, though, also exacerbated the proliferation of diversity between systems, and thus are fuelling fragmentation. The negative fallout from fragmentation is that it becomes a barrier to integration. As each system defines its own model, meta model, policies and interfaces, integration solutions become increasingly bespoke and solutions that include multiple integration pieces are therefore complex and difficult to manage (David, 2001).

Compare this with early advancements where the Internet standardised components in a way that ensured they would successfully stand the test of time — email protocols such as Post Office Protocol (POP) and Simple Mail Transfer Protocol (SMTP), as well as the Domain Name System protocol (DNS) and File Transfer Protocol (FTP), are examples of implementations that have remained almost unchanged for 30 years.

However, the blame for fragmentation lies not only with disparate tooling and flexible protocols. It may be argued that service descriptions are merely a representation of the underlying business systems. Diversity reigns supreme as businesses seek a competitive advantage.

In a veritable universe of integration options, Gates and his "12 commandments" in a sense started the IT equivalent of the big bang. Without restraint, solutions spread rapidly from a common origin, diversifying with no real motivation to consolidate.

While that kick-start was necessary to gain momentum, German academic Prof. Dr. Wilhelm Hasselbring observed three dimensions which echoed as a reminder that effort and activity aren't the only things required to maintain an interconnected world — the needs of autonomy, distribution and heterogeneity must also be balanced. In a physical sense, Hasselbring's dimensions represent the gravity that may, much like string theory does in physics, pull us back towards an optimal integration solution (Hasselbring, 2000).

Ultimately the problem remains of how to balance the need for business exploration with the need to remain connected in a way that keeps Gates' digital nervous system vision alive.

## 1.3 Hasselbring Integration Problem Dimensions

At a technological level, research into system integration difficulties has resulted in defining three key areas: distribution, heterogeneity and autonomy as indicated in Figure 1.

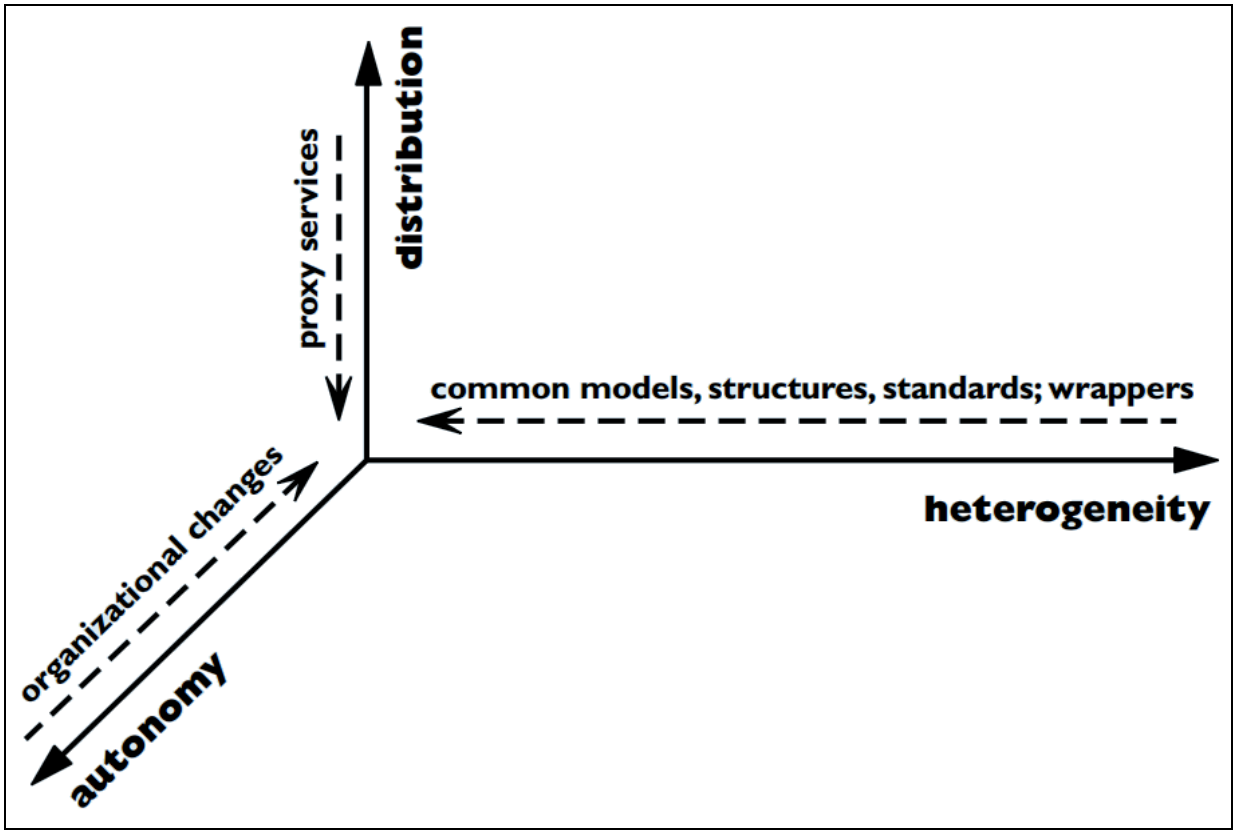



*Figure 1: Identified System Integration Difficulties (Hasselbring, 2000)*

### 1.3.1 Distribution

| **DIS** |
| --- |
| **Distribution**<br>Should not force either centralization or decentralization. |
| *Integration Dimension #1* |

The term Distribution in the context of system integration relates to the barriers that exist when integrating with remote systems compared with interfacing a local procedure call. Barriers may consist of additional security steps such as authentication and session management, utilizing a third-party data model and adhering to a different procedure call pattern.

Methods to circumvent these barriers have included the use of proxy classes, which enable the calling system to treat the method call as though it were a local call, thus hiding the complexities of marshalling and managing the remote interface (Hasselbring, 2000).

W3C's Web Services Description Language (WSDL) standard is one approach created in the area of web services, which aimed to define an interface that could be consumed in an abstract way, thus reducing the overhead of interacting with a distributed web service. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint (W3C, 2001).

More recently, distributed environments combined with multi-agent systems are forging a new path for system and solution development. Based on the fundamentals of distributed artificial intelligence (DAI) — which tackled the problem of splitting complex problems into smaller pieces and distributing the computation — multi-agent systems are concerned with segmenting the overall application solution into parts and decentralizing the processing of these parts by brokering calls when required (Kishore et al., 2006).

This decentralization is really a continuation of the service-oriented architecture (SOA) approach — which segmented a solution functionally — and extends this by accounting for distribution and coordination. The advantage of this approach is that the agent pieces can be reused by multiple applications and therefore further increase reuse potential.

### 1.3.2 Heterogeneity

| **HET** |
| --- |
| **Heterogeneity**<br>Should be resilient to change and tolerate of diverse entities. |
| *Integration Dimension #2* |

Heterogeneity poses possibly the biggest challenge to the enterprise integration space. The term refers to the degree to which change occurs and thus in integration refers to the data and message models, service bindings and message contracts. The greater the degree of change, the more difficult it is to establish and maintain an integration solution (Hasselbring, 2000).

If we consider for a moment just how successful email has been as an Internet mechanism — with the majority of Internet users also being users of email — the success of email can likely be attributed to the message format's minimal change since it was named in 1993 (Wikipedia, 2015).

A diverse world of changing data models promotes a never-ending need to maintain and support integration solutions. A potential solution to this may be to establish common object models, standardized and shared between enterprises, and this research aims to determine whether that is a realistic proposition and, if so, how it would unfold.

### 1.3.3 Autonomy

Autonomy centres on the ability of applications and software components to respond automatically to organizational change. Change may occur at multiple levels due to actual organizational restructuring, or at a technical level, e.g. a change in transactional model or a change in system component (Hasselbring, 2000).

| **AUT** |
| --- |
| **Autonomy**<br>Should be more than a dumb message router. |
| *Integration Dimension #3* |

Recent advancements in cloud technologies have improved autonomy to a limited extent, as virtualized cloud instances can be swapped in and out, or scaled up or down in many cases without the need to modify the underlying application. Technologies such as Microsoft's Windows Communication Foundation (WCF) framework for service communications have externalized service connectivity from the application, which provides some reduction in connectivity control within the application itself. Having said that, major organizational changes such as a shift off a cloud platform or replacing WCF would certainly require a significant engineering effort.

## 1.4 Adding an Architectural Dimension

In the context of a local area network (LAN), Hasselbring's integration problem dimensions are a complete way of classifying integration barriers. In the context of a wide area network (WAN) or cloud-centric world with the consideration that an integration platform could become a common core platform, there is value in promoting architecture from a barrier to a dimension (as seen in Figure 2) as individual architectural considerations, such as scale and the architectural 'abilities', become barriers in their own right (Hasselbring, 2000).

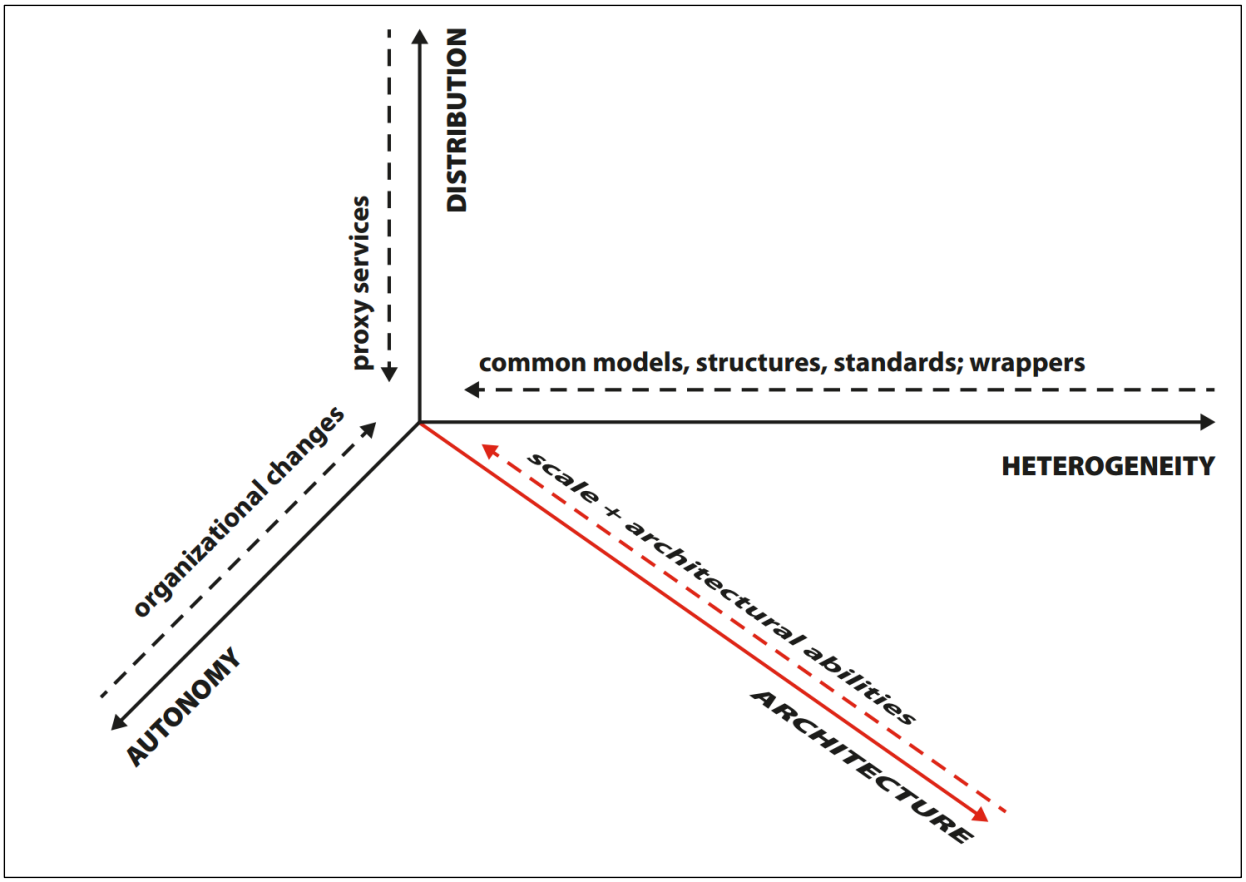



*Figure 2: Hasselbring-Comb Four Integration Problem Dimensions*

There are many definitions of IT architecture, but for the purpose of this research, we will adopt the broader enterprise architecture (EA) definition, which defines a macro view of an organization and its operation. By using this definition, we inject business and strategy as considerations within Hasselbring's model, which help balance the model at scale against the real world (Hasselbring, 2000).

### 1.4.1 Enterprise Architecture Framework (EA3)

| **ARC**<br><br>**Architecture**<br>Architecture is critical to scale globally and support all levels of business. |
|---|
| *Integration Dimension #4* |

In recent times, enterprise architecture (EA) has grown in popularity as a method for increasing organizational agility by documenting and referencing the structure of an organization and the knowledge that exists within. At a macro level, it is a holistic view of the organization, which also provides visibility of the complex internal relationships.

EA3 is a widely accepted framework for enterprise architecture, which breaks down into three distinct organizational levels — strategy, business and technology.

**EA = S + B + T**

*Enterprise Architecture = Strategy + Business + Technology*

As the nature of this research centres on not only organizations and their ability to integrate at a software level, but also the holistic view of integration occurring as an enterprise-wide solution, it is important to consider all levels of the business that may be involved in the process.

By cross-referencing Hasselbring's integration problem dimensions against the EA3 business levels as seen in Table 2, we are able to investigate the most significant barriers to integration, and where they sit within the levels of the organization. Taking a macro view of the Hasselbring dimensions and probing external barriers that may affect integration integrity offers additional insight into the full complexities of integration that technical barriers alone may fail to uncover (Hasselbring, 2000).

| Dimension[1] | Code[2] | Strategy[3] | Business[3] | Technology[3] |
|---|---|---|---|---|
| Distribution | DIS | Strategic Distribution Barriers | Business Distribution Barriers | Technology Distribution Barriers |
| Heterogeneity | HET | Strategic Heterogeneity Barriers | Business Heterogeneity Barriers | Technology Heterogeneity Barriers |
| Autonomy | AUT | Strategic Autonomy Barriers | Business Autonomy Barriers | Technology Autonomy Barriers |
| Architecture | ARC | Strategic Architecture Barriers | Business Architecture Barriers | Technology Architecture Barriers |

*Table 2: Hasselbring Integration Barriers (Hasselbring, 2000) Stratified by Business Level*

[1] Hasselbring's three dimensions + additional architecture dimension (Hasselbring, 2000).

[2] The 'Code' column provides the abbreviated code used to prefix the principal requirements within the integration problem dimension as discussed in the approach section below.

[3] The three levels of business (according to enterprise architecture theory).

## 1.5 Research Questions

The objective of this research is to investigate how it may be possible to establish a global communications ecosystem that would promoted a faster rate of integration for business data and information that would in turn promote business at the speed of thought.

To address this primary objective it is important to identify the barriers, within all four dimensions, that prevent seamless information exchange, and determine whether these barriers have existing solutions or whether new approaches are needed.

Additionally the research aims to step back and take a strategic view of integration rather than relying on organic evolution of the integration space that is unfolding piecemeal over time.

Results will be analyzed by stratifying according to Hasselbring's software integration problem dimensions, to determine which dimension represents the best return on investment.

Primary Research Question:

- Research Question A: How can we establish a global communications ecosystem?

Secondary Research Questions:

- Research Question B: How can we reduce barriers to integration solutions?
- Research Question C: What would a platform designed for "business @ the speed of thought" look like?
- Research Question D: How can we motivate integration solutions to use common models and standardised approaches?
- Research Question E: Is seamless information exchange achievable?

## 1.6 Approach

Although the research approach is fully covered in the methodology section, it is important initially to provide a high-level introduction as the approach influences the literature review.

As this research represents new thinking in the area of integration it was important to establish an approach that would encourage gradual movement towards a solution as research uncovered additional considerations.

An 'action research' (alternatively known as 'collaborative inquiry') approach presented the best opportunity to establish a group that would identify the problem, discuss considerations, establish a solution and review it against the current operating environment. The process illustrated in Figure 3 would be repeated in an iterative co-learning way that would develop the research ontology over time.

Action research aims to address theoretical, practical and social aspects of the problem and ultimately determine if the possible solution represents a desired direction. To ensure the process remained scientific, outcomes from each phase were collected, analyzed and presented to the participants in a systematic fashion, ensuring each iteration delivered additional value.

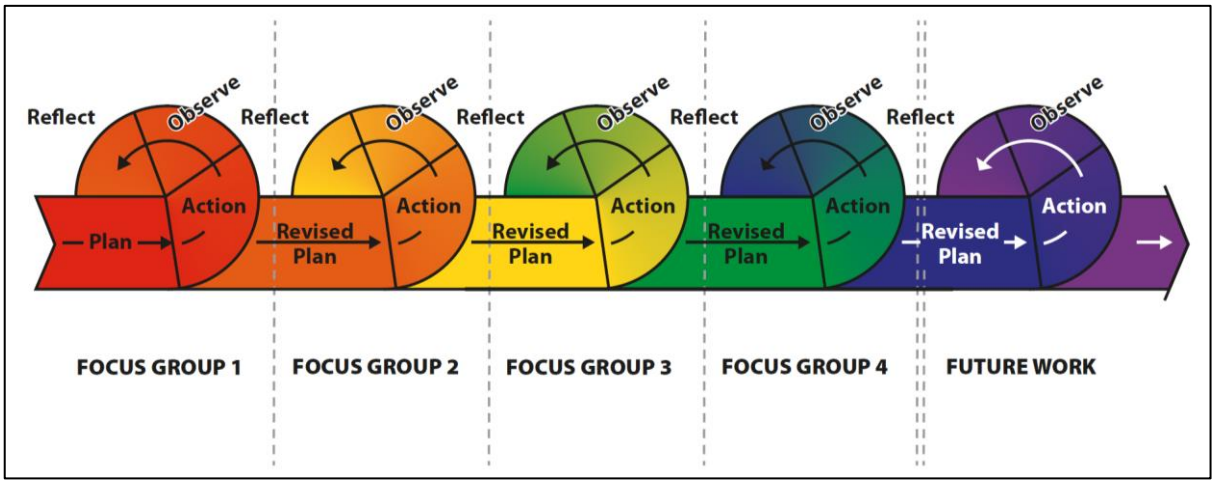



*Figure 3: Research Approach*

As the research topic represents a real problem in a real-world situation, the social dimension was a significant component of the discussion and, as such, researchers and participants alike sought a solution in a collaborative fashion.

Outputs from the action research process were delivered against an establish research ontology which helped create structural clarity for those involved in the study, and provided a useful tool when referring to an item during discussions.

The codings of domain, dimension, principle and assessment criteria shown in Figure 4 were used throughout the research.

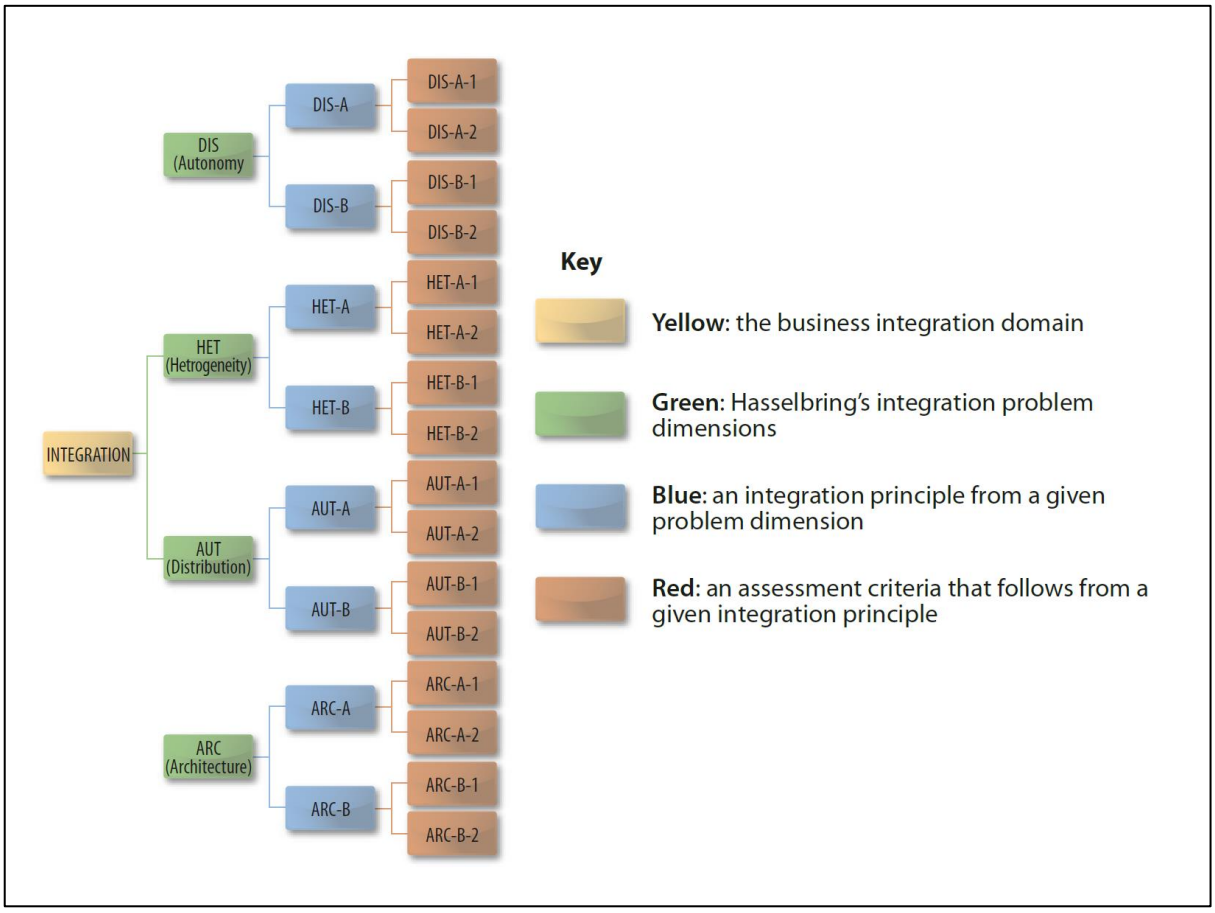



*Figure 4: Research Ontology*

## 1.7 Conclusion

In this chapter we have introduced the problem domain of enterprise integration and how it prevents information from flowing at the speed of thought. In particularly we have introduced Hasselbring's work, in particular his three integration problem dimensions. We briefly discuss the relevance of these dimensions in today's operating environment, and suggest the addition of an extra dimension to consider as we explore ways that integration can be improved.

Additionally, we introduce the research questions and the research approach. An 'action research' approach presented the best opportunity to establish a group that would identify the problem, discuss considerations, establish a solution and review it against the current operating environment.

In the following chapters, we will layout the following key areas:

- *Literature Review* — a thorough review of existing academic material including journal articles, theses and other published works in the field of enterprise integration.
- *Methodology* — we will discuss, in more detail, the action research method and how it was conducted for the purpose of this research.

- *Results* — we will review the results attained from both the survey conducted and the focus groups.
- *Reactor Platform* — we will take what was learnt from the results and propose a hypothetical solution ("the Reactor Platform") to enterprise integration barriers. The platform is presented as an entry to final discussions.
- *Final Review* — we discuss the platform with the focus group and conduct interviews with subject matter experts in order to finalize problem areas and potential solutions.
- *Discussion* — we summarize and discuss all results and focus on the most significant findings ahead of conclusions made.
- *Conclusion* — finally we present conclusions and review areas for future work in this field.

In the next chapter, we will explore existing academic material, journal articles, theses and written works that relate to the field of enterprise integration. Additionally we will review trends and changes that have occurred in recent years in this field.

# 2

# Literature Review

Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In the previous chapter we introduced enterprise integration and discussed how it is a very real problem impacting business efficiency. In this chapter we explore the explosion of integration solutions options that have unfolded since the world of the superhighway was unleashed, and how permutations of solutions continue to diversify with no motivation for convergence. Additionally we look at how this divergence is impacting on the digital nervous system and preventing it from reaching its potential.

## 2.1 The Big Bang (Integration Solution Diversification)

### 2.1.1 Introduction

When we talk about a lack of motivation for convergence, we are really acknowledging that, like many things in life, there are two sides to the equation. On one side is a constant stream of technological advancement backed by the commercial pressure to leverage technologies tactically for financial gain. On the other is a strategic best-practice approach, consolidation of technology and the opportunity to build a better tomorrow rather than just making do today.

The problem is that the Internet today is like the bedroom of a small boy who lacks parental supervision — it's littered with old toys, chocolate bar wrappers and PlayStation game cartridges, all preventing access to the computer. There is no motivation for the boy to clean his room and likewise with Internet technology, there is no great incentive for developers to align with best-practice approaches to keep the World Wide Web tidy and efficient — or at least the incentives that exist are less of a force than the commercial pressures.

If we were to start a movement to improve the Internet's current structure, we would begin by setting a direction for the future. Fortunately, Bill Gates and other similar technology visionaries have

established a blueprint for us — *Business @ the Speed of Thought*. Accessible from a plethora of devices, it appears to be a realistic pathway forward.

To achieve this goal we must establish a plan of attack.

1. First, we need to take stock of what technologies, patterns and practices exist today, and create a base technology registry for integration.
2. Next, we need to determine what our integration requirements are to achieve the *Business @ the Speed of Thought* vision and conduct a gap analysis between that and the existing registry.
3. Finally, we need to establish transitional architectures that will mobilise us towards the end goal.

The Internet today is a global collection of online services, which interoperate to provide solutions to consumers. If we intend to improve the Internet's interoperability, we must first understand how it currently hangs together and determine how we can resolve some of Hasselbring's integration challenges (Hasselbring, 2000).

The Internet is a constantly evolving network and is always online. Any proposed transitional architectures must therefore be capable of being phased into operation without interruption to existing networks and services.

### 2.1.2 Point-to-Point Integration

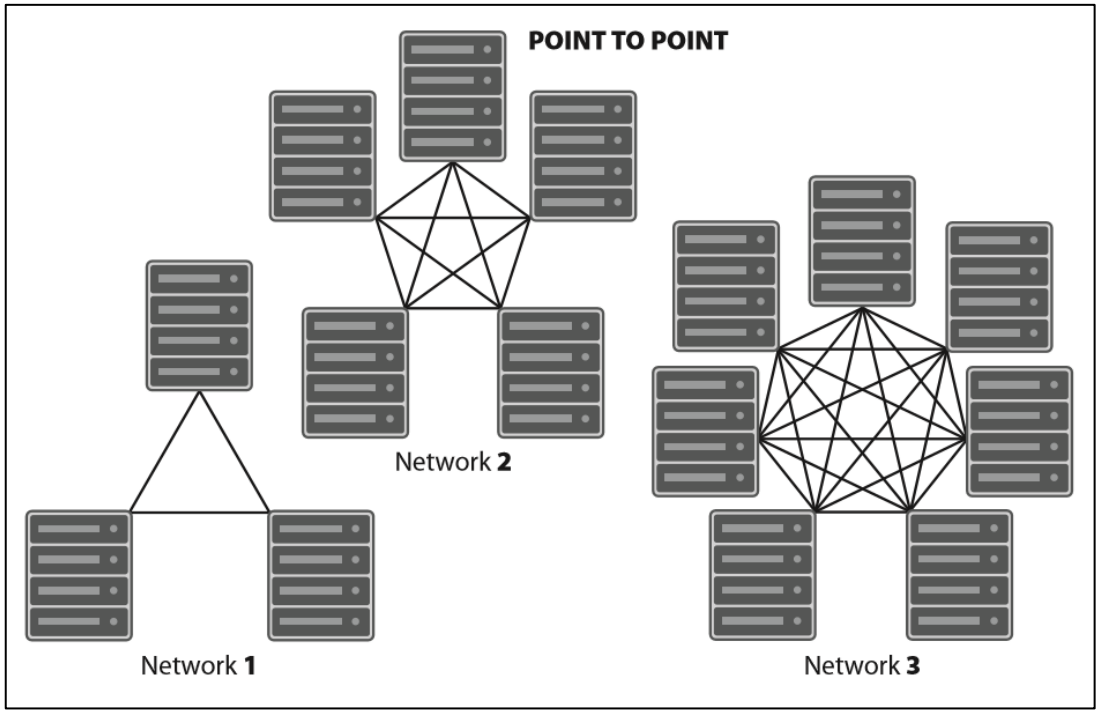



*Figure 5: Point-to-Point Integration Pattern (Lam & Shankararaman, 2007)*

Our logical starting point is to consider the integration patterns employed which allow applications to communicate with one another. The simplest way to integrate two applications is via a direct link or point-to-point integration where each pair of applications is coupled with a unique adapter component responsible for translating and integrating application messages. P2P has its advantages in that it is technically the easiest integration approach to achieve, especially when controlling the development of both applications, which in turn affords control of the communication protocol. Given that communication is direct, and additional routing is unneeded, performance is good and latency is low.

**AUT-A**

**Autonomy**
Establish a self-governing ecosystem.

*Autonomy Principle #1*

At face value, this sounds reasonable. However, when we consider Hasselbring's integration problem dimensions, we can observe some inherent problems (Hasselbring, 2000).

Autonomy is completely decentralized, which means the more applications there are in the network, the more fragmented the overall solution becomes, which can be difficult to engineer as complexity increases.

Distribution also does not scale. As seen in Figure 5, each additional component adds an increasingly large number of connections to the network of application components, which is an unnecessary overhead, and requires sophisticated orchestration from each node to ensure data is synchronized and/or available in a timely manner.

Decentralization also affects Hasselbring's heterogeneity dimension, as it is difficult to make a change to any one component without invoking a great deal of change within the other components. This leads to an environment with decreased flexibility, and reduced and inaccurate documentation (Lam & Shankararaman, 2007; Hasselbring, 2000).

**DIS-A**

**Service Oriented**
Must integrate cloud-based services.

*Distribution Principle #1*

Point-to-point integration is the most rudimentary of integration techniques — but what is surprising is that although it was originally designed as a solution for the local area network (LAN), connecting IT components within an organization's network, comparisons can be made between LAN-based P2P solutions and the way cloud-based integration is unfolding today.

It's sobering to think that wide area networking (WAN) could be adopting a P2P approach when one considers the number of permutations of integrations that would be required to link all online services.

### 2.1.3 Hub and Spoke

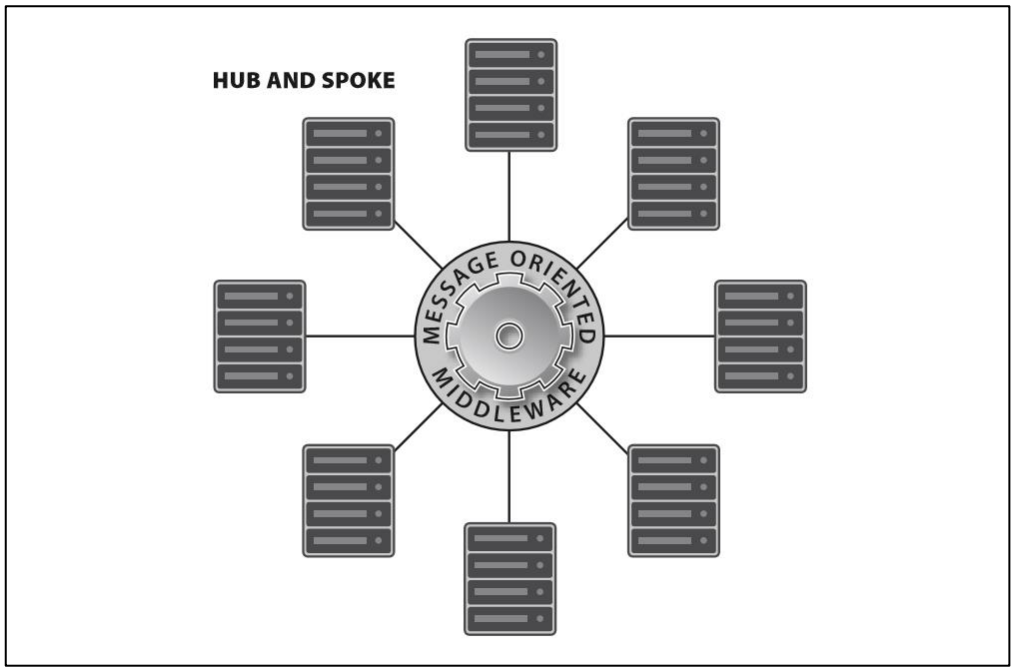



*Figure 6: Hub and Spoke Integration Pattern (Goel, 2006)*

Hub and spoke integration, as seen in Figure 6, is designed to remove complexity by reducing the need for application components (spokes) to talk to each other. Instead each component communicates with an intermediary (hub) which, in turn, orchestrates communications with the system's other components. (Goel, 2006).

**AUT-B**

**Intelligence**
Hubs should be more than just routers.

Autonomy
Principle #2

When reviewing the performance of this pattern against Hasselbring's dimensions (Hasselbring, 2000) it can be seen that autonomy is improved by varying amounts depending on the central hub component's intelligence.

Distribution has improved due to the reduction in necessary communication pathways, as has heterogeneity because of the reduction in data model owners. However, if the middleware piece is dumb, effectively a message router, it may be necessary for integration adapters to be implemented alongside the spoke components. An adapter links directly to its corresponding component and acts as a transformer for messages to and from the spoke component and the middleware piece.

**HET-A**

**Maintainable**
The platform should provide tools that ensure it can be managed effectively.

Heterogeneity
Principle #1

Another advantage of hub and spoke integration is its resilience to change, as a component upgrade will only affect one spoke in the system and therefore only one adapter must be modified. In a point-to-point network, all connecting components will require change to consume the target component's new interface (Goel, 2006).

Conversely, the disadvantage with hub and spoke integration is that as the number of components increases, more load is placed on the middleware piece that would have been distributed under a point-to-point implementation. A federated implementation alleviates some scalability concerns while providing improved management of the components.

When considering current cloud-based trends, we are left wondering what the central piece is in a universe of cloud solutions. There are many hundreds of thousands of service nodes in the cloud without a clear autonomous hub. This reinforces the view that cloud-based integration solutions are indeed point to point. As cloud computing and distributed environments become more prevalent, solution components have become increasingly dispersed and middleware is performing more of a facilitator role with the challenge of adapting to a more fluid market rife with change (Emmerich, 2000).

**DIS-B**

**Auto-Configuring**
Distribution routes should be resilient and auto-configuring.

*Distribution Principle #2*

Because the number of nodes in the cloud is so large, it seems illogical that there would be a single centrepiece for all services. In more recent times we have seen the advent of peer-to-peer technology where access and load are distributed efficiently across a mesh of nodes — an approach that has yet to be applied to the business service space.

The problem, therefore, remains of whether the scale of the cloud demands additional scale to the traditional hub-spoke model — perhaps where one or more formalized intermediary levels exist that can be extended in a mesh-type structure. This will be investigated and discussed later in the document.

### 2.1.4 Middleware

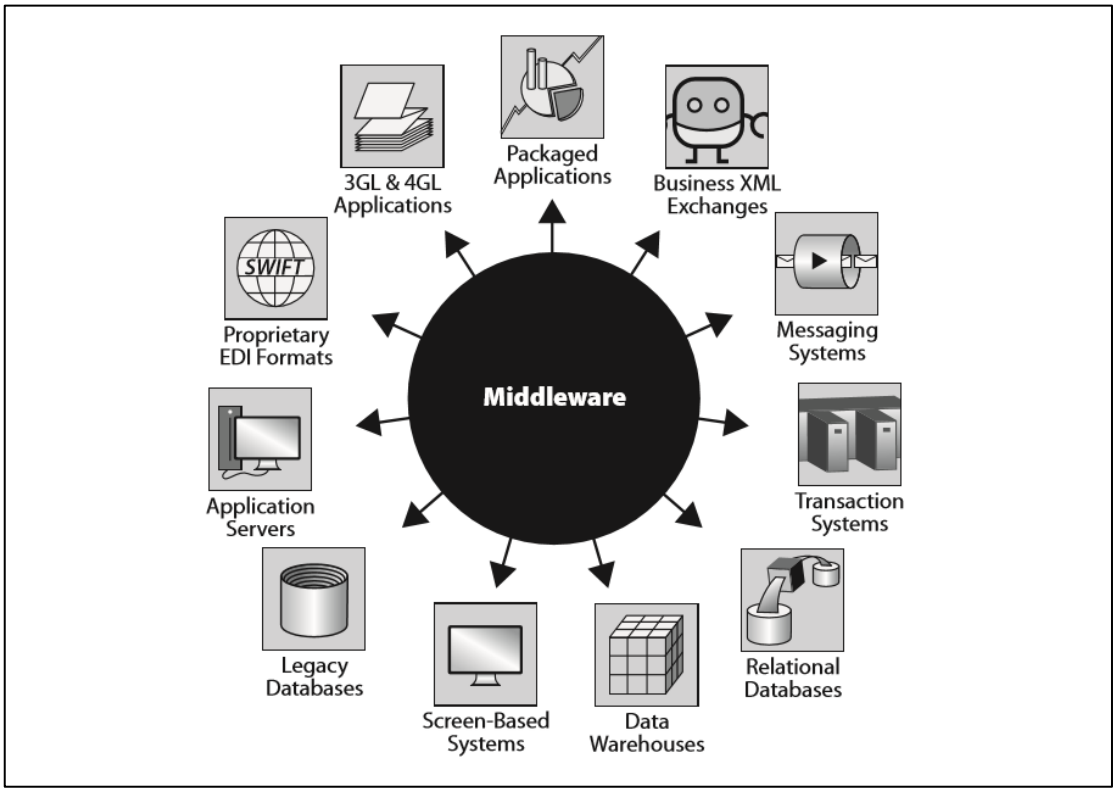



*Figure 7: Examples of Middleware ("Middleware Services" (n.d.), 2015)*

As seen in Figure 7, middleware is often used to aggregate and coordinate business application components. Middleware offers further advantages over point-to-point when intelligence is added to make decisions in accordance with interaction plans on whether to route, split or combine messages, as well as controlling synchronicity by optionally queueing messages on the way to their destination.

This increase in autonomy provides numerous benefits including the ability to manage communication channels, monitor messages and trigger events based on message flow, which can be beneficial especially when integrating components that have common payloads.

When we look deeper at middleware we see that it works by exposing a common managed interface while overlaying and hiding underlying layers, components and interfaces, which facilitates communication and simplifies access due to a reduction in heterogeneity (Emmerich, 2000), and protects against component change.

Advancements in modern languages have also led to middleware development. One example is reflection, which has enabled middleware to be reconfigured on the fly — capable of modifying its behaviour without the need for coding changes. The middleware interface remains intact and is able to support both reflective and traditional middleware approaches (Kon et al., 2002).

To construct a mesh integration platform, we must not only address barriers to integration but also ensure support for suitable best-practice approaches in the design. Middleware has been broken down into a number of valuable subtypes:

*Communication middleware* manages distribution and communication channels. We have established distribution as a foundation principle. However, it is important to also acknowledge that for an integration platform to be successful at scale, it must not be tightly bound to those communication

channels. They must be dynamically constructed based on dynamic binding configuration available at the time of need.

*Component middleware* (Klefstad et al., 2002) involves fabrication of solutions by coordinating reusable software components which provide a subset of functionality. This is a significant difference between traditional middleware, which typically provides the solution service interface, and an integration platform, which must remain solution agnostic to comply with Hasselbring's heterogeneity dimension (Hasselbring, 2000). The more the platform knows about the solutions it services, the more it will be impacted by change. Solution logic, therefore, must be constructed dynamically with a business intelligence engine, which may incorporate a container-based processing model.

**HET-B**

**Abstract**
The platform must be loosely coupled to integrating systems.

*Heterogeneity Principle #2*

*Model-driven middleware* (Gokhale et al., 2008) is concerned with extending component middleware with support for model-based development. Similar to the heterogeneity consideration with component-driven middleware, for an integration platform to avoid change management due to a change in model, it must at a base level be agnostic to the models it is working with. This can be achieved by linking transformation technologies such as XSLT with the business intelligence engine to transform and execute actions as the result of a payload injection.

*Adaptive middleware* (Clarke et al., 2001) provides the ability to reconfigure components and or system configuration based on a user request. This is a highly desirable feature that helps reduce heterogeneity concerns in the platform, which can respond to change without redeployment or system administration.

**ARC-A**

**Reliability**
The platform must be available at all times.

*Architecture Principle #1*

*Context-aware middleware* (Rouvoy et al., 2009) has the added capacity to reconfigure itself based on environmental conditions or a change in context, e.g. failover mechanics or operating hours support. While this is not a focus of Hasselbring's dimensions, it is critical at scale (Hasselbring, 2000). The integration platform will become a core integration infrastructure and therefore must have high levels of availability.

### 2.1.5 Enterprise Application Integration

Gable (2002) describes Enterprise Application Integration (EAI) as the unrestricted sharing of data and business processes among any connected applications and data sources in the enterprise.

Before EAI, as indicated in Figure 8, IT solutions were rarely prepared for integration as they were mostly built in an ad hoc manner. Additionally, because a preference for purchased solutions existed, an organization's application suite contained application components which were not made to work with each other, leaving valuable information siloed within individual components (Tse, 2015).

Furthermore, due to a fluid business environment with rapidly changing requirements, it has been difficult for the market to establish a consistent framework or approach to conducting integration solutions. As such, the safer option of applying conventional integration is often taken which, although functional, often anchors or inhibits an organization through lack of agility. Enterprise application integration, however, provides a componentized approach empowering the coupling and decoupling of system components, and embracing change. The result is a more flexible solution where reuse of code, process and components may exist (Irani et al., 2003).

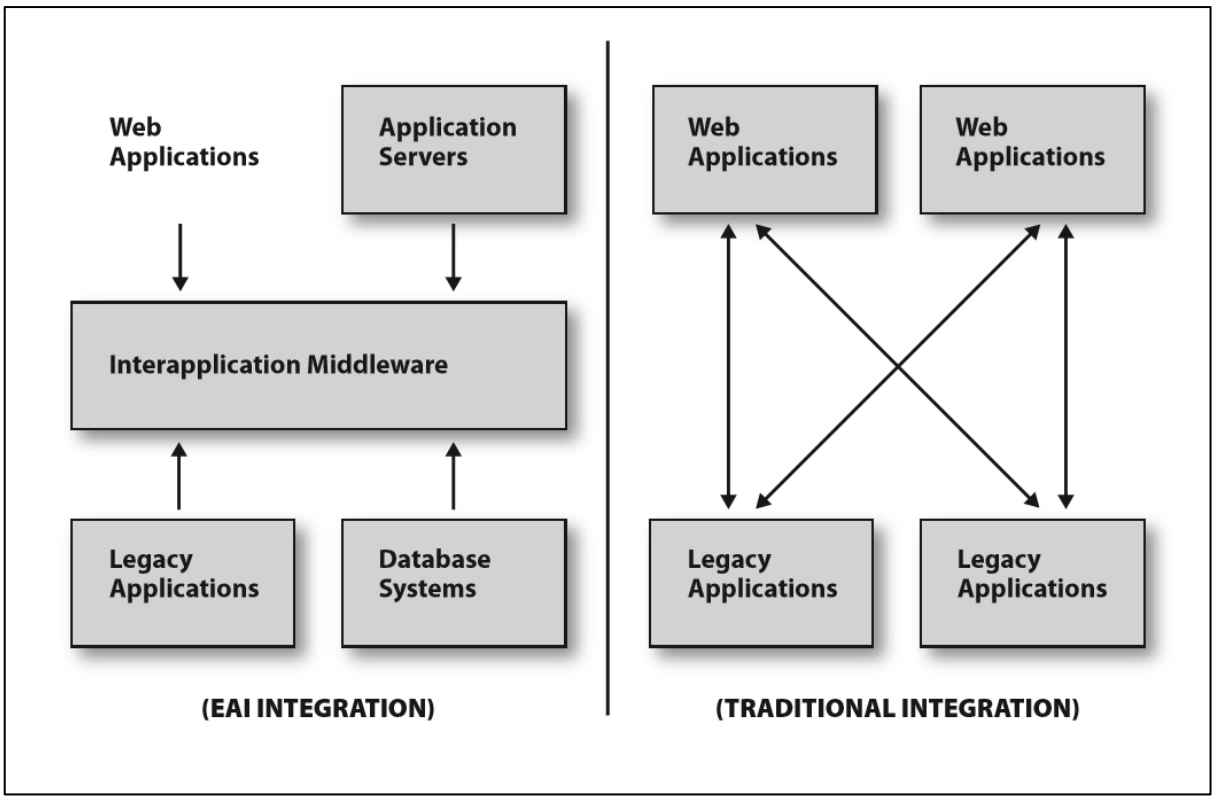



*Figure 8: EAI/Middleware vs Traditional Integration (Pappa & Stergioulas, 2008)*

As well as a dynamic and changing internal business environment, Irani and Love observed that the business domain was continuously expanding, due, internally, to available technology and business solutions — and, externally, to the desire to integrate between organizations (Irani & Love, 2001).

Abstract middleware is often used synonymously with enterprise application integration. However, although there are similarities, differences are significant and therefore the concepts should be treated separately (Alonso et al., 2004).

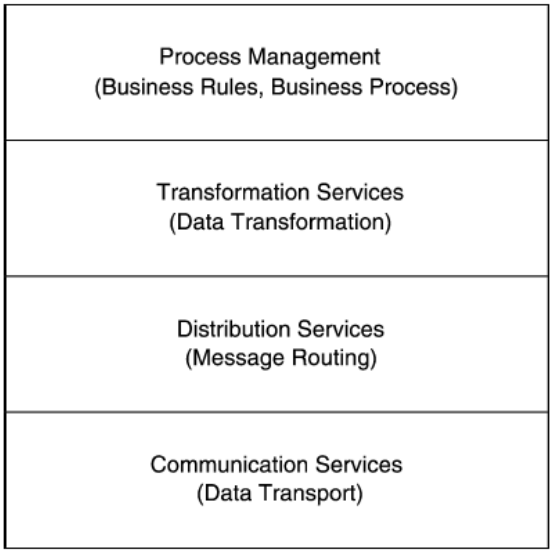



*Figure 9: EAI Integration Key Functions (Eshel, 2000)*

Enterprise application integration (EAI) products often house a number of key functions which form something of a stack as illustrated in Figure 9. It is important to appreciate these layers as although they manifest in different ways in modern solutions, the fundamental responsibilities remain the same:

- The *communication services* layer manages the transportation of data to and from the application components. Communication can be either synchronous or asynchronous depending on the technology used. In an enterprise environment messaging is often queued to protect against loading spikes. Components are typically linked by adapter services which transform messages between the component and the EAI solution.
- The *distribution* layer is responsible for routing requests to the appropriate system component and operates in a request/response or publish/subscribe mode (Yu et al., 2008).
- Depending on how advanced the adapter services are, the data received by the EAI component data may require additional transformation before being provided. Additionally if parallel processing is involved, the transformation layer would be responsible for joining multiple component responses together. The transformation layer is the ideal position from which to handle transaction management.
- Finally, the *process management* layer provides the opportunity to implement an organization's business intelligence. Activities and tasks can be applied at this level to ensure business value is delivered, and it also provides the ability to participate in inter-organizational process sharing (Eshel, 2000; Lublinsky, 2001; Puschmann & Alt, 2001).

### 2.1.6 Enterprise Service Bus

**ARC-B**

**Usability**
The platform must be functional and easy to use.

*Architecture Principle #2*

In recent times, a new form of middleware has become popular as a foundation for application services. Service-oriented architecture (SOA) has been accepted as a best-practice approach to deliver IT solutions against business requirements without incurring excessive expense (Woods, 2003). To aggregate functionality from multiple service providers, SOA employs enterprise service bus (ESB), an advanced middleware component which exposes business functionality to service consumers using standard data semantics and transport protocols (Lam & Shankararaman, 2007).

In the broadest sense, the ESB, as seen in Figure 10, is an enabler for SOA by providing the connectivity layer between services. A service in this context is not governed by any specific protocol in the way that SOAP or HTTP web services are, and is not required to describe itself using any specific standard — although commonly this is the case. The main objective of a service is to provide reusable functionality by publishing its metadata description and exposing itself to loosely coupled systems (Schmidt et al., 2005).

Although the ESB does a great job of loosely coupling its functionality, it still requires the consumer to connect, bind and consume the metadata definition — thus the relationship is not 100% dynamic. To resolve this, and to address Hasselbring's heterogeneity problem dimension, any improved platform must allow for a data-driven service registry which can be referenced for connectivity information for service endpoints. This enables administration of the connected interfaces to be controlled through a management interface and the platform remains solution agnostic (Hasselbring, 2000).

With such an integration platform approach, application servers remain relevant as providers of business functionality, and solutions continue to have a longer lease of life as technological deficiencies garnered over time can be hidden from the outside world behind the service interface (Chappell, 2004). This allows providers of functionality to control their own interface change management, which reduces cost by removing the need to upgrade versions continually to be in line with the integration platform.

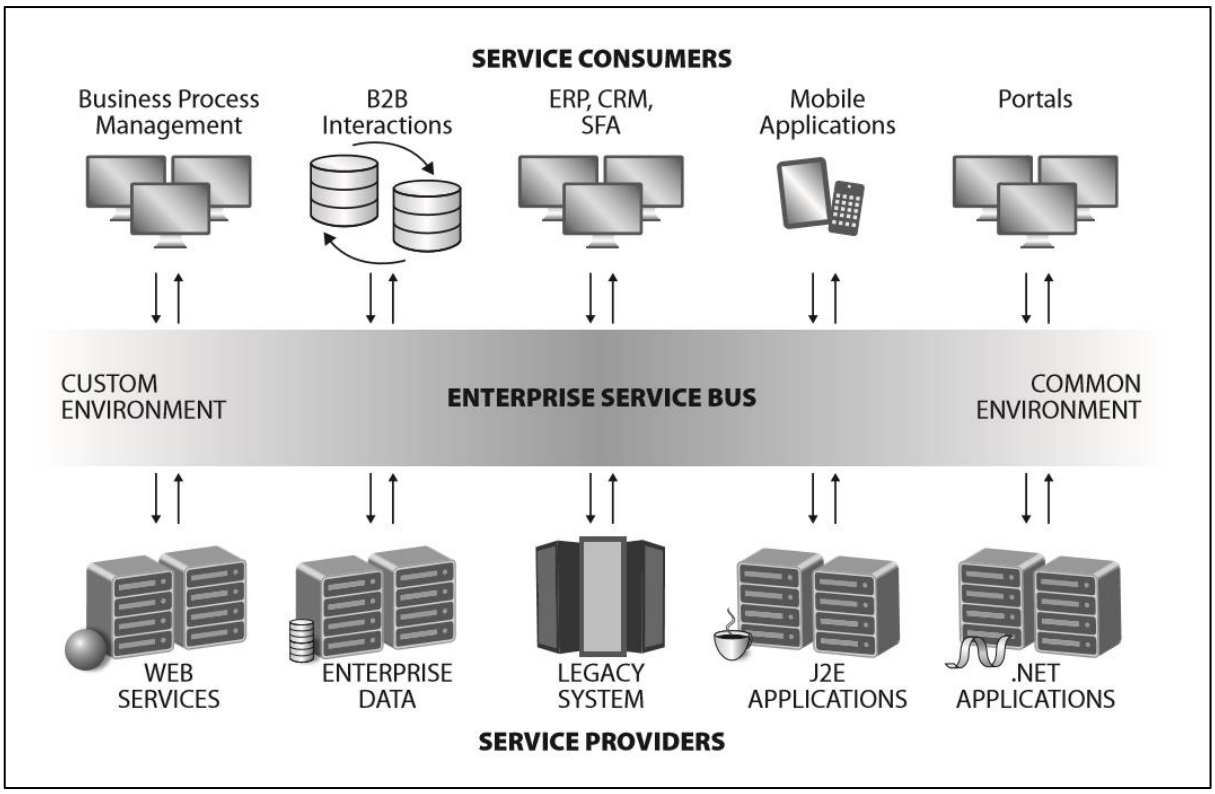



*Figure 10: Enterprise Service Bus (ESB) (Lam & Shankararaman, 2007)*

Normally as businesses connect their information systems to the cloud and adopt wide area network (WAN) and/or web service approaches to their information systems, patterns for integration become more complex. Increased decentralization of information systems requires the maintenance of internally held standards such as data integrity, security and reliability (Foster et al., 2002).

**HET-C**

**Standards Based**
The platform must be built on industry-accepted best-practice standards.

*Heterogeneity Principle #3*

Ferguson proposed that while organizations may use an enterprise service bus internally as a solution for resolving integration complexities, a similar Internet service bus approach might need to be adopted to address greater WAN complexities (Ferguson, 2007).

Key functions of an ESB include Invocation, Routing, Mediation, Adapters, Security, Management, Process Orchestration and Complex Event Processing (Menge, 2007)

*Invocation* - For the enterprise service bus to provide value, it must be capable of aggregating business functionality from multiple provider services. It achieves this by using invocation, which is the process of sending requests and receiving responses from connected component services and resources. In modern times, this involves the support of web service technology including WSDL, SOAP, UDDI and WS-related standards. Additionally, common communication protocols such as TCP, UDP, HTTP and SSL are often used, and other protocols may be supported should the need arise.

*Routing* - An enterprise service bus uses routing functionality during delivery of messages. Delivery may be determined by different forms of routing logic including content inspection, load balancing, broadcasting, business process or configuration. The act of routing affords the ESB the opportunity to delay delivery and split or combine messages.

*Mediation* - refers to the ability to transform or translate messages from one format to another. This is a critical part of the integration process as it ultimately unifies disparate business functionality from multiple systems and allows it to be presented in a common structure.

*Adapters* - Application adapters are used to separate the physical act of connecting to integration service interfaces from the logical act of invoking business intelligence. By abstracting access to business functionality in this way, an adapter can protect against the replacement of an integration service component, e.g. CRM, and can reduce the effort required to manage integration services that originate from a common provider.

**ARC-C**

**Security**
The platform must be secure.

*Architecture Principle #3*

*Security* - The enterprise service bus provides support for security by managing message encryption, handling authentication and authorizing access via an access control layer. Security requirements imposed by integration services are aggregated and managed by the ESB.

*Management* - One of the most beneficial features of the ESB is its ability to provide support for centralized management. This may or may not include monitoring, logging, configuration and administration. A single administration endpoint reduces overhead and maintenance costs.

*Process Orchestration* - One of the limitations of an ESB is its statelessness, which is to say it does not maintain the state of a transaction over multiple invocations. To counter that the enterprise service bus may provide support for process orchestration — which enables business intelligence to be supported and in supporting business process, aligns the ESB with desired business functionality.

2.1.6.1 Complex Event Processing – Given that ESBs support asynchronous messages, a type of event message, it stands to reason that they are also capable of supporting event processing. When an event arrives, it can be matched with appropriate processing logic through an adapter module configured to process the event

2.1.6.2 Integration Tooling – Advanced ESBs also allow for dynamic control of integration services via third-party software interfaces, removing the necessity, in some cases, to modify build and deploy component services.

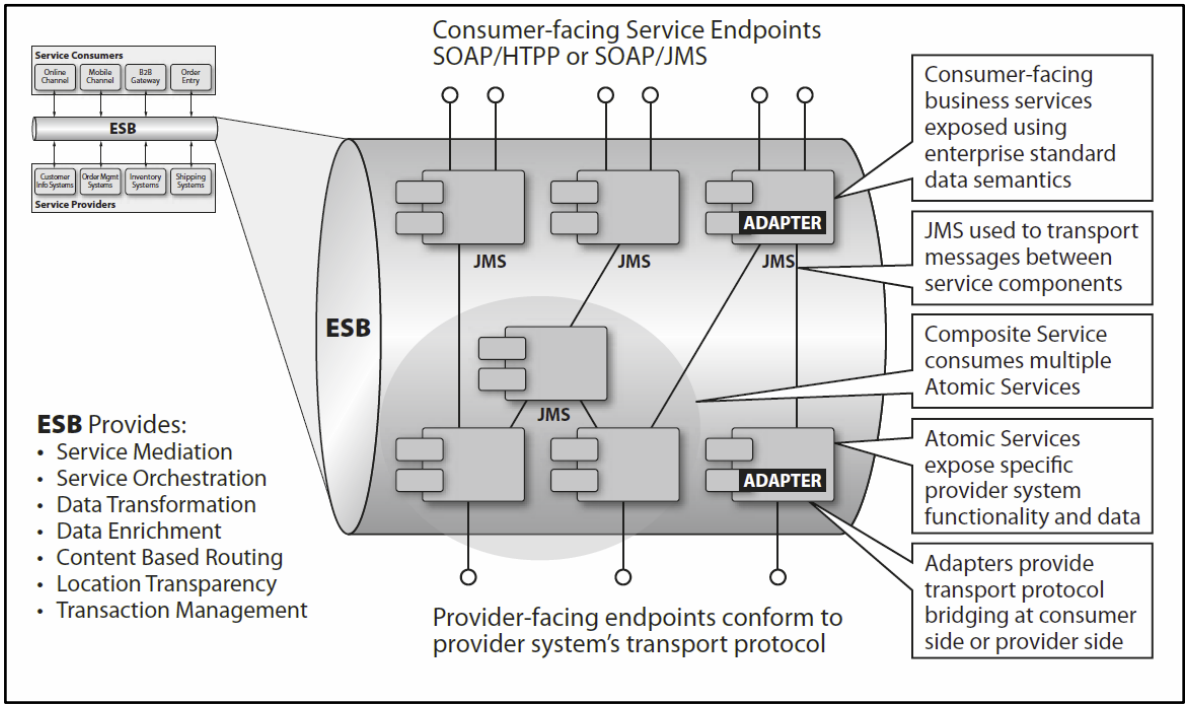



*Figure 11: Enterprise Service Bus with Adapters (Lam & Shankararaman, 2007)*

Figure 11 helps us understand the internal functions of an ESB including how adapters deal with inbound and outbound messages, how messages are transformed in transit, how transaction management is handled using composite services and, ultimately, how services are exposed to the outside world. All of these items are critical for any middleware piece of software that plans to control data flow between business systems.

### 2.1.7 Event-Driven Architecture

As mentioned previously, an enterprise service bus may optionally link in a workflow engine capable of executing business process. A business process is "a set of activities and tasks that, once completed, will accomplish an organizational goal" (Appian, 2015, para. 1).

Business processes are extremely valuable to an organization as they represent value to the customer and to the organization in terms of intellectual IP (Shankararaman et al., 2007).

Application service components may include multiple sequences of sub process or activities, which can execute functionality, but the ultimate goal with business processes is that they are fully automated and capable of being bundled in a way that promotes reuse between collaborating enterprises (Lam & Shankararaman, 2007).

Given that an ESB is a service-driven architecture, it requires invocation of business process to be triggered by something external such as a human interaction with a user interface, a scheduled task or another event-based trigger.

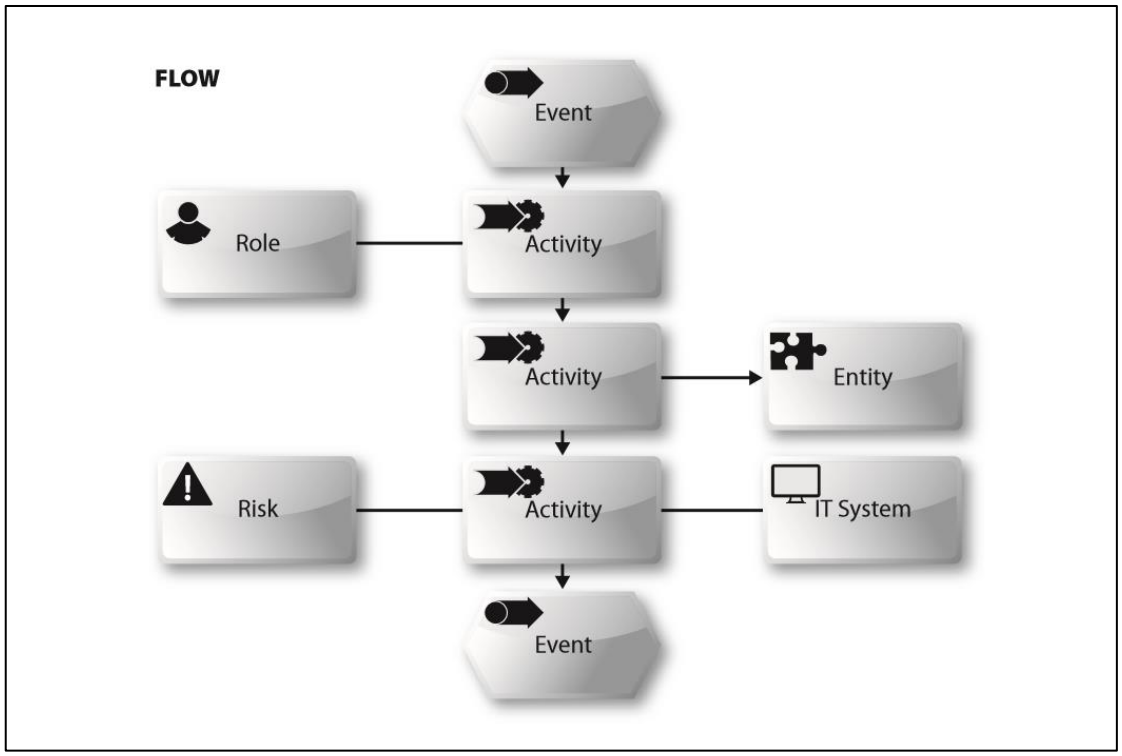



*Figure 12: Example Workflow Pattern (Van der Aalst et al., 2003)*

The relationship between a business process and software services can be confusing. The challenge of how software manages a real-world process is critical to future integration endeavours as integration solutions look to harness and extend integrated business process to add value to their existing structure.

Van der Aalst et al. (2003) provide a workflow pattern, as illustrated in Figure 12, which is commonly adhered to today. A workflow begins with an event or trigger and continues through conditions similar to a stage/gate structure as inputs, further events or computation achieve set milestones and allow the flow to progress.

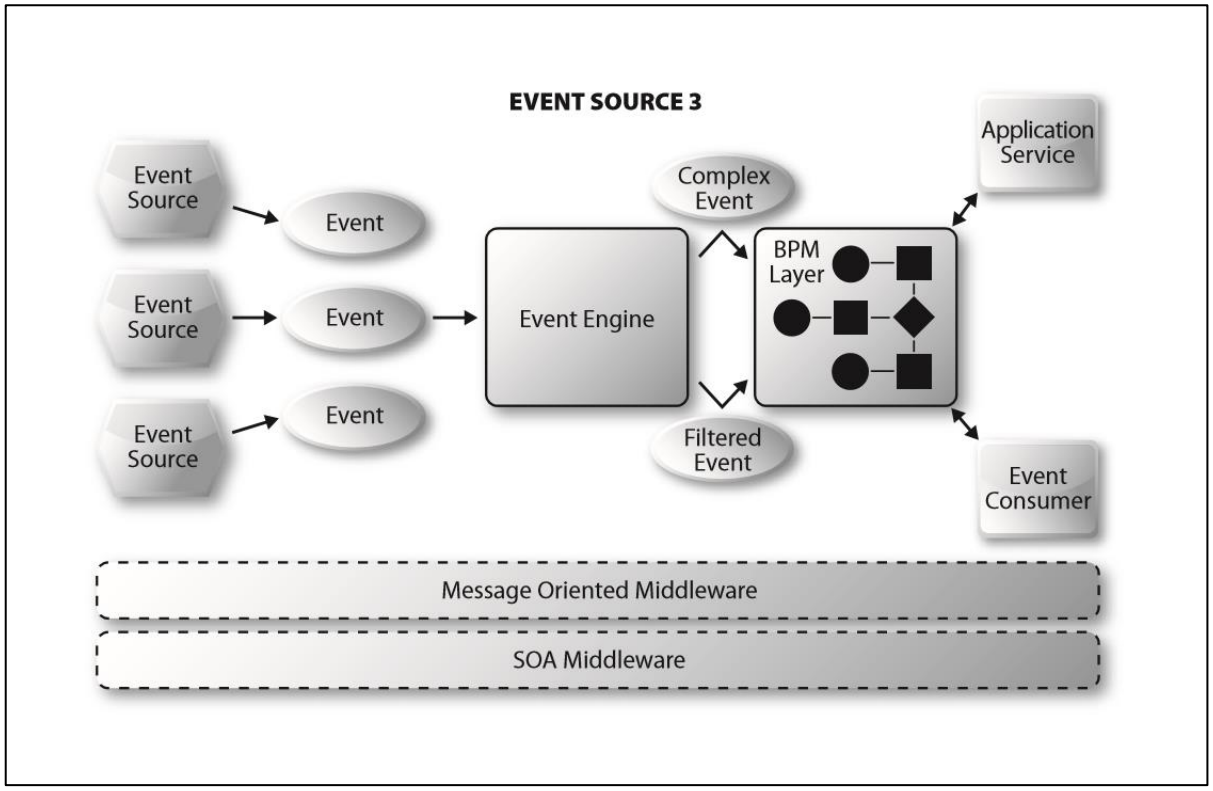



*Figure 13: Enterprise Integration Patterns III (Hohpe & Woolf, 2003)*

According to Michelson, an event is a significant occurrence within or outside an organization. Events may indicate different business scenarios and may be informational, warnings, problems or thresholds, or other situations that a business would typically respond to.

Events traditionally incorporate a messaging-type structure with a header that enables the service to route it, and content which allows it to be processed. Typically the content includes all information relating to the event that is required to achieve desired outcomes (Michelson, 2006).

Event-driven architecture by nature is loosely coupled and therefore fits well with a service-oriented architecture, although the two are not synonymous. In an attempt to promote automation of systems, event-driven architecture is geared to respond to an event condition, and when coupled with business process engines as in Figure 13, is capable of registering additional internal events and chaining multiple sequences of business intelligence together without requiring further user interactions (Michelson, 2006).

Schulte proposed key patterns for Event Processing — namely, Simple Event Processing, Complex Event Processing and Event-Driven Processes (Schulte, 2004).

2.1.7.1 Simple Event Processing – With this pattern, the event source sends a business event to the receiving application as the event consumer, which in turn processes the business event. The event normally follows a "publish-subscribe" pattern and the message-oriented middleware component may optionally transform the message in transit (Schulte, 2004). An example of this flow is a purchase order being transformed from one system to another, as illustrated in Figure 14 below.

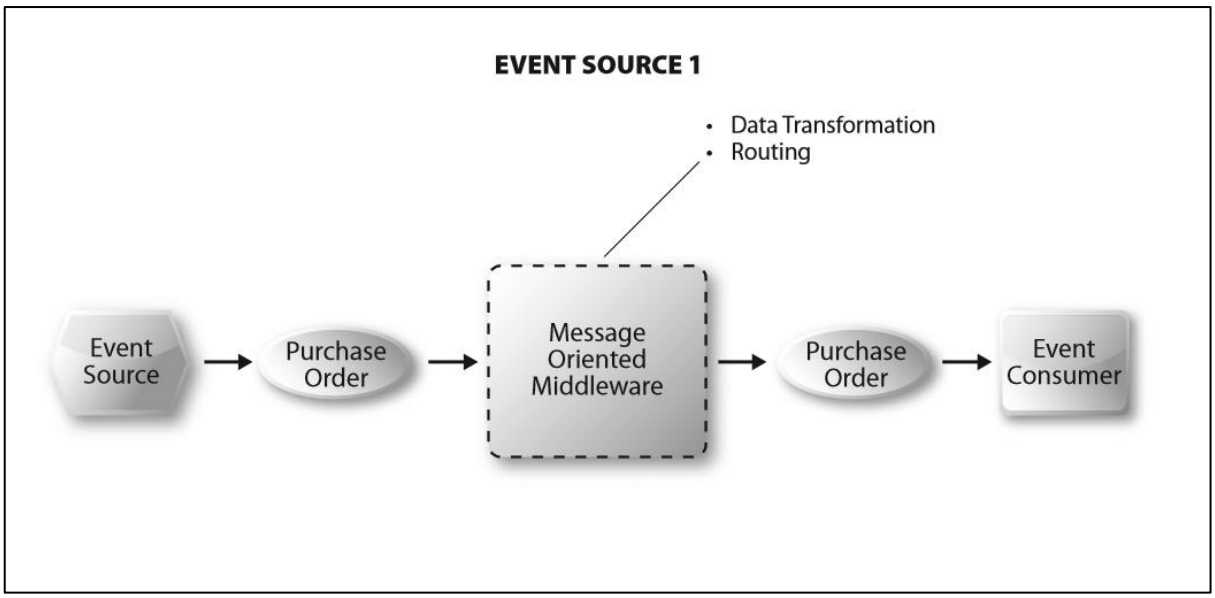



*Figure 14: Enterprise Integration Patterns I (Hohpe & Woolf, 2003)*

2.1.7.2 Complex Event Processing - With this pattern, the messaging middleware has been replaced by an Event Engine, as illustrated in Figure 15, which may process or interact with the events on their way to the consumer. An Event Engine might be useful in cases where filtration is required or when splitting or merging of messages is desirable.

2.1.7.3 Just as the application service has the option of containing its own application model definition, separate from business service components, the introduction of an Event Engine enables the possibility that event messaging on either side may be different (Schulte, 2004).

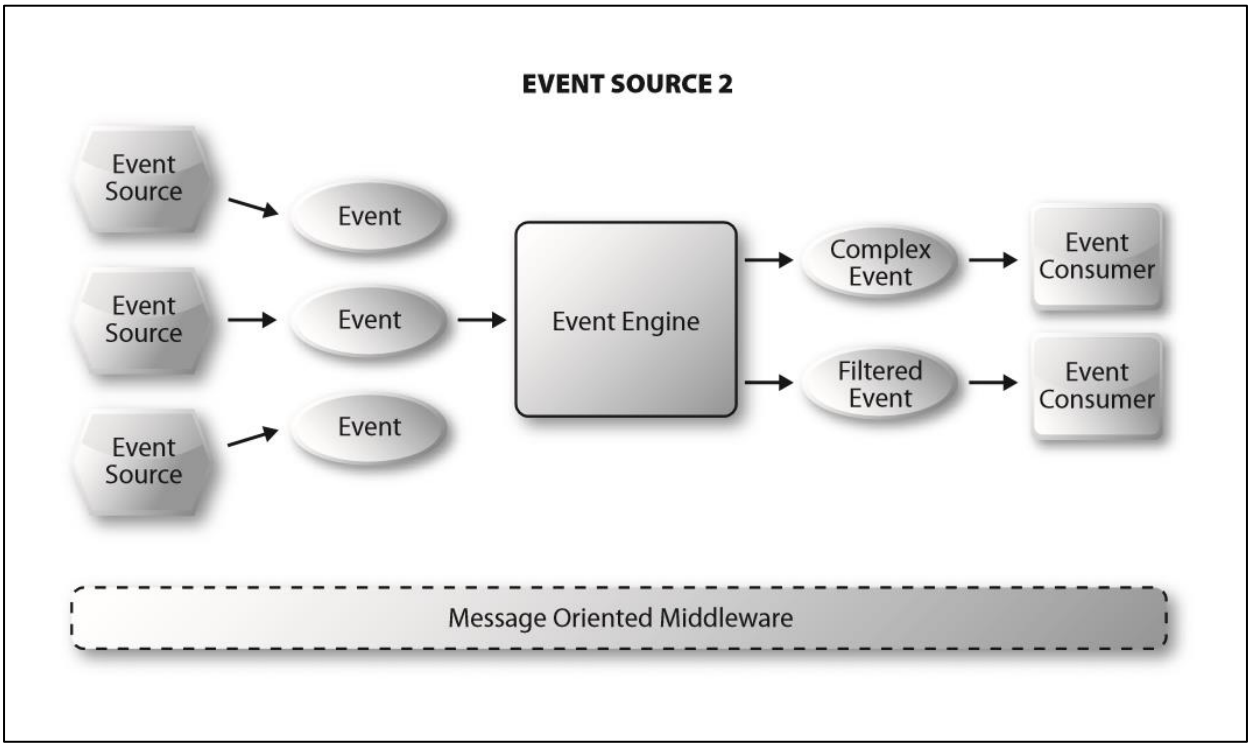



*Figure 15: Enterprise Integration Patterns II (Hohpe & Woolf, 2003)*

2.1.7.4 Event-Driven Processes - Event-driven processing links events with implemented business process mechanics. Events are associated with process flows and/or activities. The business process management layer is responsible for persisting any necessary state-fullness that is required between instances of events to ensure the process flow continues from its previous point (Schulte, 2004).

2.1.8 Web Services Architecture

2.1.8.1 Web Standards

The World Wide Web Consortium (W3C) controls and matures a number of foundation standards which are used as the building blocks for the World Wide Web. Figure 16 lists the current set of web standards provided by W3C.

In particular, the following key areas are the focus of W3C:

- *Web Design and Applications* - Standards in this area are concerned with the rendering of web content for display or consumption at user interface level. This is particularly pertinent for interface integration solutions that relocate key data from one user interface to another (Paulheim, 2009). Current standards that apply include HTML, CSS and other contributing web application technologies.

- *Web of Devices* - The focus for Web of Devices includes device connectivity and mobility, and enables the access of key data by users on the move including mobile phone technologies, connected cars, tablets, mobile computing devices and consumer appliances. The need to focus on standards for connected devices suggests it also has a bearing on integration solutions as systems may or may not include device-based components and related technologies.

- *Web Architecture* - Web Architecture provides foundation components and best-practice approaches which underlie the World Wide Web including URIs, HTTP and transport-related technologies. Web architecture is an important aspect of enterprise integration solutions as it is involved in the communication between systems through connection negotiation or establishing the base communication rules.

- *Semantic Web* - The focus of the Semantic Web is to define a standardized way of storing and transporting data so that it can be realized in an increasing number of ways by humans and computers alike. As identified previously, diversity of data models is a key aspect of heterogeneity as increased diversity can hinder the ability to consume the data (Hasselbring, 2000) unless constructs have been established to deal with that diversity in a structured way.

- *XML Technology* - XML and related technologies provide a data management layer for storing, retrieving and transforming web data. XML provides an important standard for information exchange and encourages a common method of communication. However, it really stops where the Semantic Web starts as it does not solve the problem of how specific data models are defined within the XML payload.

- *Web of Services* - The concept of the Web of Services centres on the fundamental integration need for a 'Consumer' to request data from a 'Provider'. The Web of Services exposes functionality in a standardized way, which promotes this type of information exchange and relies heavily on WSDL, XML, SOAP and HTTP technologies.


*Figure 16: W3C Web Standards (W3C, 2015)*

Recently, an existing technology, REST (Representational State Transfer) has become a popular extension protocol for HTTP-based information exchange. REST's simplicity provides an advantage over heavier technologies as it removes unnecessary overheads when constructing integration solutions (Pautasso, 2008) and promotes Business to Consumer (B2C) integration solutions such as social media integration.

### 2.1.9 Application Integration

#### 2.1.9.1 Old Data, Application, UI-Centric Integration

Data integration, possibly the most prevalent form of integration (Al Mosawi et al., 2006), has been defined as "the problem of combining data residing at different sources, and providing the user with a unified view of these data" (Lenzerini, 2002, p. 1). A compact definition that appears to hold true today despite the disparity of data sources now exists.

This view of the world may be acceptable on a LAN or a WAN with select partners, but is architecturally inconsistent with a true WAN approach where storage implementation should not be exposed externally. A WAN-based integration platform is responsible for brokering the interconnectivity between services at the presentation layer and should be data-level agnostic.

This greatly simplifies the number of supported interface technologies and, as a by-product, addresses heterogeneity. By forcing integration at this level it also ensures no business intelligence is bypassed, thus improving autonomy. The ability to bypass service layers to access, transform and mobilise data quickly and efficiently (Samtani et al., 2002; Linthicum, 1999) should also be enhanced by the more efficient distribution capabilities of the platform.

#### 2.1.9.2 New Service + Model-View-Control Centric Integration

*Model*

Particularly with service-oriented architecture, separation of concerns for a system often involves a Model, View, Control design where "a Model object encapsulates an entity from the application's functional core, Views present data and information to the consumer and Controllers are associated with views and allow manipulation of the provided data and information" (Buschmann et al., 2007, p. 18).

Typically, separation in this manner promotes integration as models can be consumed at different levels within a system — before or after different levels of controller logic or business intelligence have been applied. To achieve model integration a number of model patterns can be applied. 'Referencing', in essence, reuses a model from another component or system directly; 'Transforming' takes a reference-able source model and transforms it to a destination model; and 'Merging' combines models from two or more sources and presents the resulting model view its View interface (Kühn et al., 2003).

One of the by-products of the Industrial Revolution of the 19th century was the advent of reusable or interchangeable parts. This approach was used initially in the manufacture of guns and led to a transformation in production, with identical parts being used as components for multiple products.

In a sense, models within a system may be considered the interchangeable parts of software. Different software systems fabricate models from local or remote systems, optionally adding business intelligence to provide a software solution (Lankhorst, 2004).

Typically, an enterprise system has three principal layers. The presentation layer is responsible for displaying information or providing services, the domain layer is responsible for applying business logic, and the data source layer is responsible for managing access to the underlying data storage component (Fowler, 2002).

With a distributed integration platform, however, data storage is not a concern as data payloads are processed and re-routed as message packets. Despite this, the platform should still support the concept of models and meta-models because separating models through layers of abstraction encourages reuse of application, domain and business functionality (Kühn et al., 2003). The platform should address this through transformation where models can be combined and extended as needed in conjunction with the business intelligence engine and routing configuration.

*View*

In a service-driven architecture, the view does not refer to the user interface but rather to the service interface with which integration interacts. This layer exposes and/or obfuscates functionality as needed to ensure the system's integrity. Consequently, a number of traditional integration avenues are encapsulated in the service view.

With method-level integration, applications are usually integrated by sharing common business logic accessible through a set of common methods — an approach that most sources acknowledge is, in fact, a more complicated form of application-level integration and is therefore rarely used in place of middleware (Linthicum, 1999).

A WAN-based integration platform, however, is able to control which method functionality it wishes to expose through its service interface, significantly reducing complexity and increasing the rate at which integration can be established (Linthicum, 1999; Johannesson et al., 2000; Lankhorst, 2004).

*Control*

Working at a higher level than objects or methods, business process — alternatively called the business method (Samtani et al., 2002) — is the formalization of activities or tasks that deliver functionality and

execute business intelligence. The goal of process integration is to interrelate process sequences across applications and/or share between organizations.

Linthicum asserted that when integrating two systems, the approach would focus either on data or the business model. In practical terms integrating with the business model meant integrating with either the application interface level, the method level or the user interface level. These were the three possible ways to extract information as the result of business intelligence (Linthicum, 1999).

Traditionally with a multi-tier enterprise network there are multiple levels at play. For example, the focus or scope of an application service is generally broader than a business service. Also, application services are concerned with preparing functionality for interaction with the outside world, whereas business services are typically meant to be silos of functionality within an organization (Lankhorst, 2004). Models defined at application level may or may not be the same as those implemented at business service level. However, these transformations are also hidden by the service interface, which again reduces integration complexity.

Hasselbring elaborated by grouping areas of application architecture into a horizontal enterprise application integration layer to encourage an organization to think holistically and to produce an overall application vision (Hasselbring, 2000). A WAN-based integration platform encourages the same type of thinking.

In a WAN-based environment, process logic and business intelligence can also be exposed through service interfaces, thus presenting application functionality to the outside world while extending implementation flexibilities internally. Application component services may be off-the-shelf or custom-built and the challenge of application integration is to broker collaboration between application services and collectively deliver against the business vision (Schmidt et al., 2001).

The role of the WAN-based integration platform, therefore, is to be capable of executing this business intelligence as a result of an event that has been triggered in the integration platform. Because the business intelligence can be instantiated by many different users or systems in different contexts, it motivates a service provider to expose a purer implementation of the business logic which improves autonomy through simplified interactions (Al Mosawi, 2006). It is important to note that "integration is concerned with the process itself, not with its output" (Aubert et al., 2003, p. 4) — meaning the business process sequence may have a different result when executed by two separate systems as they may be reliant on data or integration with other services, which would provide a different context to the process.

## 2.2 Business Integration

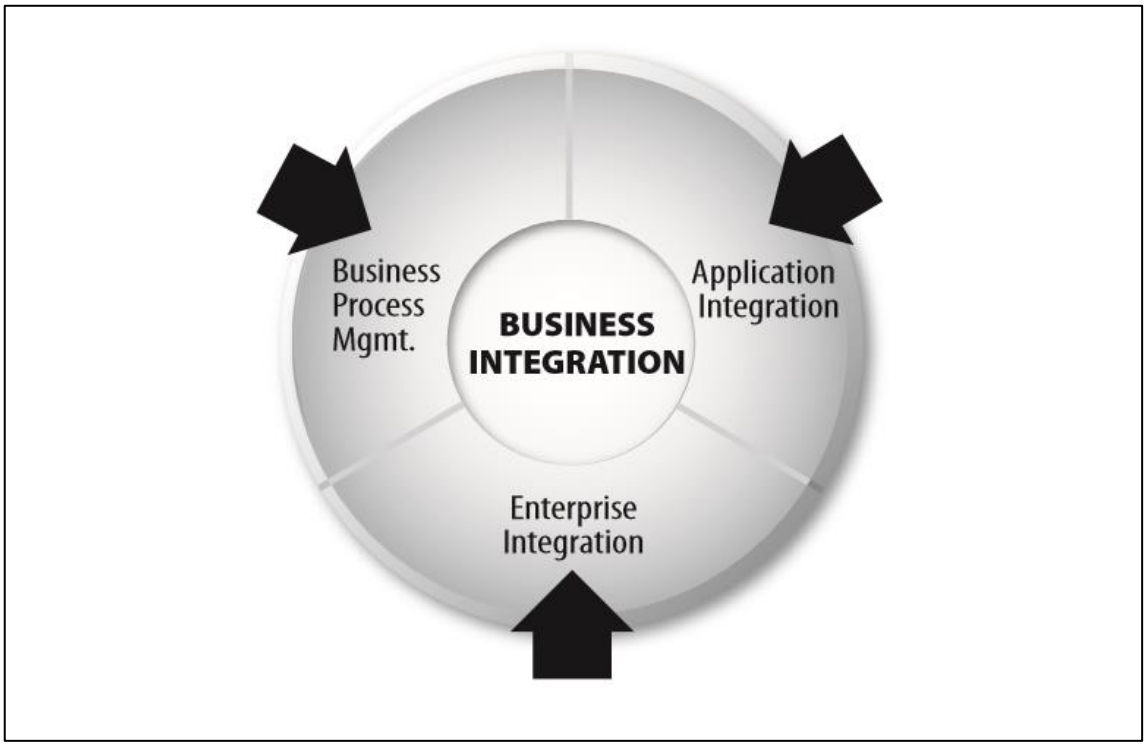



*Figure 17: Business Integration Components (Lam & Shankararaman, 2004)*

Business integration in an enterprise integration sense is the measure of alignment between technology and business goals and objectives. By aligning business and technology, an organization can better deliver against its overall strategy. In accordance with determining 'what' before 'how', business integration is often conducted before application integration can begin (Lam & Shankararaman, 2007).

In an attempt to maximise the efficiency of this alignment, information technology has been separated into different layers, with different business objectives. The application layer typically implements business processes and is closely linked with business integration. The business architecture layer affords an opportunity for enterprises to establish patterns, practices and shared models that can be accessed by multiple applications. The technology layer provides the technical means with which to deliver against application and business architectures (Hasselbring, 2000). Any constructed platform for integration must consider all layers of integration for it to be successful. These relationships can be viewed in a number of ways depending on the context and perspective. One possible view is the business integration component view provided in Figure 17.

At an enterprise level, integration has been defined as the result of the cooperation and interoperability between two separate applications. Cooperation involves the readiness of the applications to be integrated, and interoperability relates to the technical ability of the systems to exchange information and handle requests (Klischewski, 2004). Only with these two enablers in place can integration occur successfully.

Beyond the abstract definition, the classification of enterprise integration (EI) has continued to diversify as the solution space has increased in complexity, technology has changed and integration demands have increased.

Initially Linthicum defined integration as the point in the system where integration takes place. This included data, method, application or interface (Linthicum, 1999).

Later, Al Mosawi et al. transitioned the classification of enterprise integration by providing two classifications. The first provided a more granular view of Linthicum's classification, while the second was a more holistic approach in which classification was determined by system layering (Al Mosawi et al., 2006).

Soon after, Lam & Shankararaman embraced the bigger picture and categorised enterprise integration according to the problem being solved. Lam's categorization as enterprise application integration (EAI), business to business integration (B2Bi) and web integration was popular with both architects and sales as it focused on 'what' was being integrated rather than 'how' the integration was achieved (Lam & Shankararaman 2007).

Depending on the context of the integration solution, all classifications (as viewed in Table 3) remain relevant. A straightforward data integration may conform more to Linthicum's classification while a high-level business-to-business integration may reflect Lam's view better (Lam, 2005). Regardless, any future integration platform must provide foundation integration capability for all of these views.

| Linthicum (1999) | Al Mosawi (2006) | | Lam (2007) |
|---|---|---|---|
| | Category 1 | Category 2 | |
| Data | Data | Business architecture layer | Enterprise application integration (EAI) |
| Application interface | Object | Business process layer | B2B integration |
| Method | Function or method | Information architecture layer | (B2Bi) and web integration |
| User interface | User interface | Inter-organizational layer | |
| | Application interface | Application layer | |
| | Presentation | Enterprise application layer | |
| | Process | Technology layer | |
| | Internal process | Middleware integration layer | |
| | Cross-enterprise | | |

*Table 3: Enterprise Integration Classifications (Linthicum, 1999; Al Mosawi et al., 2006; Lam & Shankararaman, 2007)*

#### 2.2.1.1 Logical Business Layers

According to Hasselbring, the business architecture layer is the highest level and most closely reflects an organization's practices, processes and rules (Hasselbring, 2000).

The business architecture layer is primarily concerned with behaviour such as business processes or function, but is often expanded to include actor involvement and takes care to establish a distinction between actors and roles, and their relationships with the system. Decoupling functionality from an actor in this way enables role functionality to be assigned across multiple actors, thus increasing flexibility (Lankhorst, 2004).

Drilling further into the business architecture layer, Brown established the distinction between business process and information architecture. As noted previously, business process reflects the activities and tasks that make up the system's business intelligence (Brown, 1999).

Hasselbring further elaborated by grouping areas of business architecture into a horizontal inter-organizational process layer to encourage an organization to productize its architecture internally and prepare for inter-organizational collaboration (Hasselbring, 2000). As shown in Figure 18, this was one of the first diagrams to illustrate non-technical considerations of integration and the partitioning of integration into organizational units. This is significant as the future of integration may lie with business constructing and sharing workflow components, rather than technical teams implementing integration solutions.

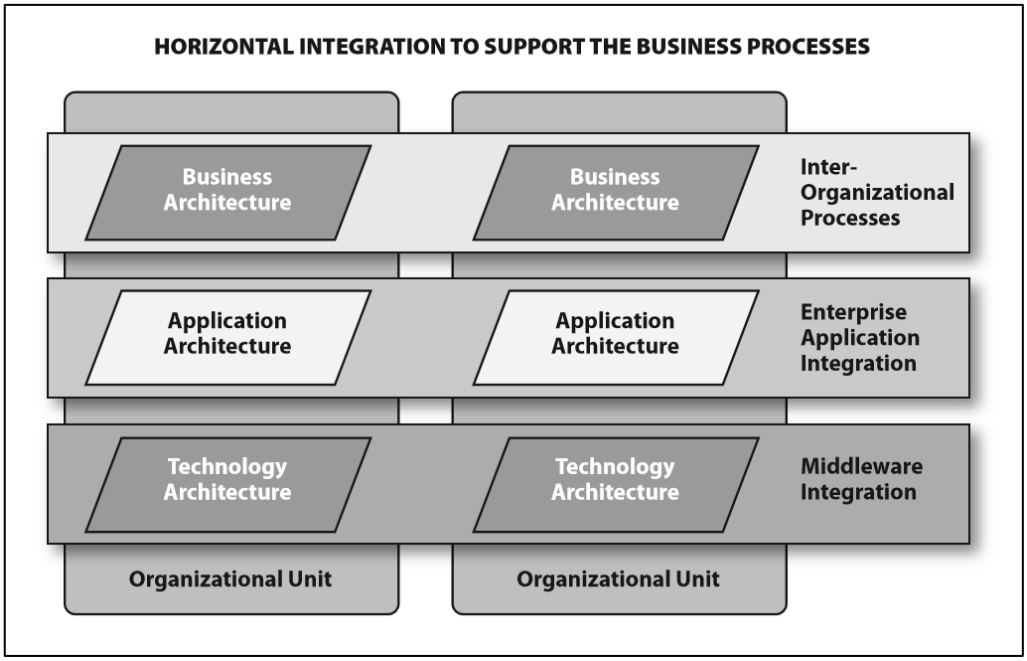



*Figure 18: Horizontal Integration to Support the Business Processes (Hasselbring, 2000)*

### 2.2.1.2 Business-to-Business Integration (B2B)

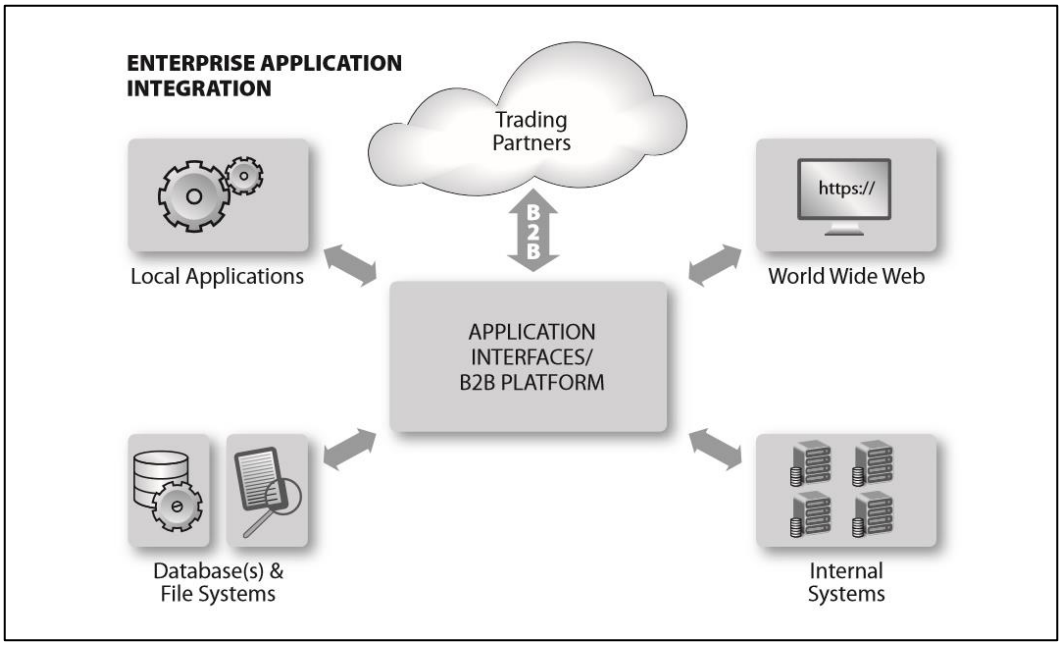



*Figure 19: Enterprise Application Integration (Linthicum, 1999)*

Traditionally, long-term business relationships meant that business-to-business information exchange, as illustrated in Figure 19, could operate with a reasonably fixed structure such as electronic data interchange (EDI). Today, however, cloud technologies and an increasingly agile business environment are encouraging the adoption of more flexible payloads when integrating between enterprises (Linthicum, 1999).

The challenge of reducing the cost of establishing B2B integration has been overcome by implementing flexible standards such as XML or RosettaNet, which uses XML to formalise messages used in interactions. This approach has been implemented in a way that allows for some flexibility should business processes change (Trastour et al., 2003).

By integrating at a process level rather than a data level, organizations afford themselves the opportunity to align business practices, e.g. in the case of business partnerships (Ortiz et al., 1999; Huat Lim et al., 1997). Standardisation at this level exists with business process languages such as BPEL4WS and BPML assisting with process integration (Peltz, 2003).

#### 2.2.1.3 Integration Solutions

While there are many approaches, patterns and practices to the technical side of integration, it is also important to understand that technology is only one aspect of integration and that much consideration should also be given to business and conceptual levels as indicated in Figure 20 (Bauknecht, 2003).

Blending enterprise layers is generally regarded as an architectural mistake for good reason. When areas of concern overlap, they start to merge and influence each other, which can become problematic when responding to change such as during a growth phase.

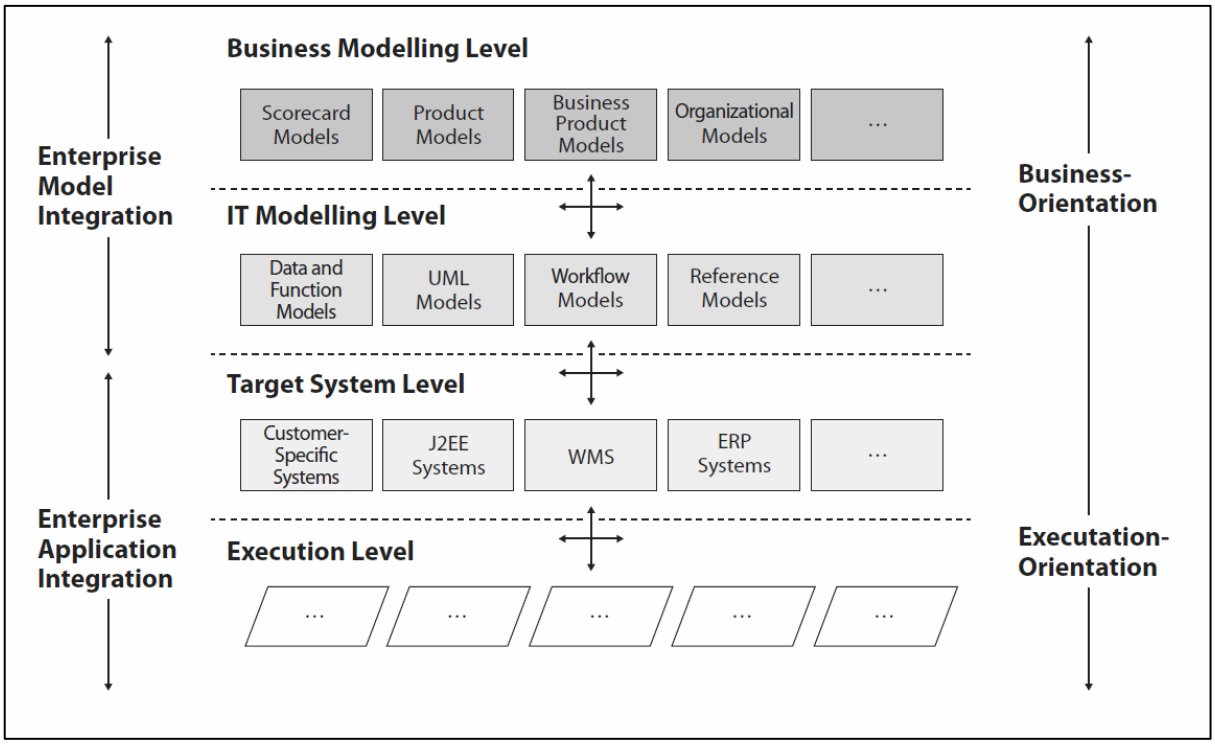



*Figure 20: Business Modelling, EMI to EAI (Kühn et al., 2003)*

Lankhorst makes the important point that it is not the responsibility of software solutions to force a company to change its business models (Lankhorst, 2004). Likewise, integration solutions should respect the same philosophy.

For this reason, it is important to have models defined at each level and to carefully manage the translation between models. Many experts have addressed the need for models at different levels and have evolved the separation of business and technology architectures (Hasselbring, 2000; Lam & Shankararaman, 2004; Perry et al., 2004).

Barriers to integration therefore transcend technology into strategic and organizational areas (Themistocleous & Irani, 2003), which raises the important point of whether diversity in business is infinite or whether there are levels of generalization that can be applied without restricting the company concerned. This thinking has implications in industry best-practice solutions such as widely distributed cloud-based integration approaches that are prevalent today and are catered for using modern web service and service-oriented architecture approaches.

At a higher level, integration barriers can be classified in a number of ways. A study conducted by Lam in 2004 — which involved multiple case studies of government organizations and their integration solutions — identified a number of key integration barriers and the levels at which they manifest (see Figure 21) (Lam, 2005). While this investigation is very useful to consider, a more generalized breakdown is needed to form the basis of integration advancement.

| | Barriers |
|---|---|
| Strategy | Lack of shared e-Government goals and objectives |
| | Over-ambitious e-Government milestones |
| | Lack of ownership and governance |
| | Absence of implementation guidance |
| | Funding issues |
| Technology | Lack of architecture interoperability |
| | Incompatible data standards |
| | Different security models |
| | Inflexibility of legacy systems |
| | Incompatible technical standards |
| Policy | Concerns over citizen privacy |
| | Data ownership |
| | E-Government policy evolution |
| Organisation | Lack of agency readiness |
| | Slow pace of government reform |
| | Absence of an e-Government champion |
| | Legacy government processes |
| | Lack of relevant in-house management and technical expertise |

*Figure 21: Identified Governmental Integration Barriers (Lam, 2005)*

## 2.3 Modern Trends and Technologies

When considering software integration approaches, it is important to cross-reference integration theory against current trends and technologies. The majority of theoretical integration, for example, was researched and documented in a pre-cloud era and, as such, has an entirely different operating context when applied today.

### 2.3.1 Mesh Networks

Although mesh theory is not new, examples of mesh such as those used in wireless networks, and cellular networks such as mobile ad hoc networks (MANETs), are impressive in terms of the sheer size of the network and reliability achieved. Mesh networks automatically adapt their routing based on

nodes present in the configuration and re-route automatically when outages occur or if shorter paths become available (Amir et al., 2010).

**ARC-D**

**Performance**
The platform must perform all functionality well at scale.

*Architecture Principle #4*

A node in a mesh network cooperates with other nodes and is responsible for relaying data in the network. This presents a potential solution for disseminating data globally by forming the foundation of a global integration network platform. Although data in cellular and wireless networks generally has only one dimension, the benefits of the networks are amplified when the data has multiple potential recipients — such as when there are multiple subscribers to the same data.

Mesh networks inherently have broadcast capability (flooding), as well as routing-based mechanics and are therefore valuable in addressing Hasselbring's distribution problem dimension. The ability of mesh networks to self-heal by constructing new routes, if a node disappears or is added, makes them extremely resilient (Campista et al., 2008; Hasselbring, 2000).

### 2.3.2 Peer-to-Peer

Another network structure worth considering is the peer-to-peer network, which has become well known via file-sharing networks such as Napster. Peer-to-peer typically operates by splitting a workload into tasks and distributing them to nearby (in the abstract sense) nodes for processing (Kamel et al., 2007).

Although a peer-to-peer network at face value seems similar to a point-to-point network, which we have already deemed problematic, there are some differences. Peer-to-peer is designed to be a resilient network capable of scaling to very large volumes. It does this by employing similar techniques to those of mesh networks where routes are being tested continuously and re-routed based on the appearance and disappearance of network nodes. While it does not address Hasselbring's autonomy concerns with point-to-point networks, it may provide insights into addressing distribution problems (Hasselbring, 2000).

Conversely, we should not confuse peer-to-peer as being the same as a mesh network, although there are similarities. Mesh nodes are considered equal and they relay data to a destination using a routing algorithm. Peer-to-peer nodes are not equal in the sense that some nodes may have the required data and some may not, so there is additional logic to determine who to distribute data to.

Each peer maintains a localized routing table consisting of its neighbouring peer's node ID and IP address. A peer can ask its neighbour about a node ID or torrent info hash using lookup queries and each node will return a list of known nodes, which are 'close' or a match to the destination hash. The peer will then ask each of the new nodes for information on the destination hash code. It is possible to repeat this iterative approach until the destination node is found and/or all routes are exhausted (Lua et al., 2005).

A global integration network needs to consider a balance between these two technologies. Mesh networks can offer a dispersed resilient network, while peer-to-peer at certain points may offer some additional performance gains and increase efficiency when dispersing large volumes to multiple recipients or by sharing node resources for the same workload.

It should also be noted that peer-to-peer protocols are currently the most active protocols on the Internet, with over 50% of all Internet traffic belonging to P2P (Cisco Systems, 2009), and are commonly acknowledged as a standard way of distributing files due to enhanced scalability options (Liu et al., 2012).

### 2.3.3 Virtualization

In recent times virtualization has played an important role within IT infrastructure solutions, particularly when they are distributed. Cloud implementations have acted as a catalyst for virtualization to become mainstream, as it can be used to deploy multiple system components in isolation on the same physical host, thus reducing operating overheads.

Recently we have seen the birth of a new approach, 'Containers', which instead of separating multiple virtual systems on the same physical host, can isolate individual processes. It's a far more granular and efficient way of managing as you do not need resources to manage multiple virtual operating systems on a single physical host (Kuck et al., 2002)

In the world of integration, it presents the opportunity to execute business intelligence rules as the result of an integration event inside a container. Crucially, the container can be instantiated virtually anywhere that has resource available without compromising system or network integrity.

Taking this approach to the extreme, it could eventually be possible to respond to a system event by injecting business intelligence into a third-party system in a safe and controlled manner.

> Example: When stock level is low, raise purchase order directly in supplier's system, if supplier has stock or if stock will be arriving within 10 days.

The prospects are exciting for interoperating business intelligence provided that a foundation for global integration is established and built upon. It would, however, require significant research and development before this level of interoperability becomes possible.

## 2.4 Conclusion

In this chapter, we have explored existing academic material, journal articles, theses and written works that relate to the field of enterprise integration. In doing so we have identified the basic components of enterprise integration at technical, business and strategy levels. We have uncovered a number of key challenges with integration and explored how potential solutions have arrived and failed to resolve fully the problem area. Finally, we have reviewed current modern trends and technologies that may play a part in improving the field of enterprise integration in the future.

As technology is always changing and there are an infinite number of possibilities available for transmitting and receiving data from one system to another, only the most common approaches have been reviewed for the purpose of this research. A more extensive literature review could focus on a more diverse set of approaches, in particular, the many cloud based middleware services that are starting to emerge at the time of writing this thesis.

Additionally, the modern trends and technologies is far from comprehensive, and includes the primary advancements made in recent years. A more extensive literature review should also include hypothetical approaches that have not yet reached the market (e.g. are discussed at universities or by internet governing bodies) in order to consider all options.

In the next chapter, Methodology, we describe in detail how this research was conducted. The methodology description takes into account the introduction and literature review sections, and lays out a plan for researching and analysing results that support the problem domain.

# 3

# Methodology

Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In the previous chapter, we explored existing academic material in the field of enterprise integration. In this chapter we discuss the research methodology used in this study to investigate specific barriers to enterprise integration and potential pathways for resolution. The study design, population and sample are introduced including the geographical area where the study was conducted. The instrument used to collect the data — including methods implemented to maintain the validity and reliability of the instrument — is also described.

## 3.1 Introduction

Business integration occurs, in one form or another, in almost all companies. This presents a unique challenge in that we must focus the research on those within an organization who have experience with integration. At the same time, we must seek out a diverse set of participants who have valuable experience with a variety of integration solutions.

In addition to this challenge, as a lot of integration theory is well and truly entrenched, we begin by conducting an extensive literature review of over one thousand academic articles and, at the same time, digging deep beneath the surface to find the truth about how people would conduct integration, if they were unshackled from existing industry approaches and trends.

Since the goal of this research was to measure thoughts and opinions, qualitative methods were employed. Research was conducted using one-on-one interviews, focus groups and a questionnaire, as illustrated in Table 4.

| Research Methodology | |
|---|---|
| 1. Literature review | Review of over 500 academic references |
| **2. Focus Group 1 (FG1)** | Introduce subject matter and form initial survey |
| 3. Survey | Conduct survey |
| **4. Focus Group 2 (FG2)** | Review results of survey and establish platform assessment matrix |
| 5. Review existing integration platforms | Conduct review of integration platforms against assessment matrix |
| **6. Focus Group 3 (FG3)** | Review results of integration platform reviews and cement requirements for platform design |
| 7. Produce integration platform design | Establish global integration platform design |
| **8. Focus Group 4 (FG4)** | Review global integration design |
| 9. Expert interviews | Audit the process and discuss outcomes from previous steps in the methodology |

*Table 4: Research Methodology*

### 3.1.1 Target Population

- The population for this thesis was professionals who have a stake in business integration solutions within their organization.
- As discussed previously, integration is more than just a software consideration. Much of the terminology is first defined at a business level and the solutions themselves run on supporting infrastructure. Consequently, it was important for the research conducted to consider people within each of these areas and for this reason we established three key populations:

1. *Population A* – **Strategy**: CxO-level strategic governance including CEO, CIO, CTO, etc.
2. *Population B* – **Business**: Day-to-day business operation, including Sales, Marketing, Business Analysts, Product Managers and Program Managers.
3. *Population C* – **Technology**: Delivery of software and information solutions including Solution/Systems Architect, Systems Administrator, Senior Developer.

Participants in the focus group must have been in their current position for at least one year, which was validated against the individual's LinkedIn profile and curriculum vitae.

Interview participants were subject matter experts identified amongst the focus group participants.

### 3.1.2 Sampling Method

The target population was sampled using a non-random sampling approach via self-referral. Volunteers who worked in an organization that had implemented a software integration project from a strategic, business or technical perspective were called for via social media, which ensured opinion was offered willingly.

For the focus groups, initially a purposive sampling approach was taken to narrow the group of survey participants into a smaller group of experienced integrators at all levels of the business. Those considered for invitation to focus groups were grouped on the seniority of their job title and, after some discussion, their greater experience with integration.

#### 3.1.2.1 Stratify by Business Level

Both the survey participants and the focus groups were stratified to include equal representation of people from business, strategic and technical backgrounds to ensure all perspectives were considered before reaching any conclusions. However, the focus and larger portion of the survey questions were aimed at technical participants to ensure the research focus was comprehensively addressed. The target was to get roughly 50% of technical participants and 25% of both strategic and business participants for a balanced perspective.

#### 3.1.2.2 Stratify by Business Size

The second stratification was by business size. It was important to garner feedback from participants from different-sized companies to help determine if there were any trends or opinions present in companies of a certain size that were not present in others. As numbers in one strata dropped, companies with the same business size were targeted for feedback.

Once the sample population of 140 participants was established, 12 focus group participants were ultimately selected from that group with the same strata ratios, using a convenience sampling approach based on access to the participant, availability and location (Auckland, New Zealand).

### 3.1.3 Data Collection

#### 3.1.3.1 Focus Groups

Focus groups, which involve organized discussions with selected groups of individuals, are particularly suited for obtaining several perspectives about the same topic, allowing us to gain insights into shared understandings (Gibbs, 1997).

Focus groups were chosen as an approach for this methodology because they provided an opportunity to learn from integration specialists what current challenges and approaches to integration exist in the industry. By having centralized discussions, we were able to create some enthusiasm for the topic at hand which disseminated beyond the immediate focus group and into their workplaces — thus having a positive flow-on effect by attracting additional integrators to participate in the survey.

The focus groups involved a valuable step-by-step process, which helped to stimulate discussion, advance opinions, beliefs and attitudes about research issues, and assess current thinking. This approach considerably enhanced decision making throughout the course of the research and maximized the value of the results.

1. Focus Group 1 (FG1) — The initial focus group was an opportunity for participants to be introduced to the research topic, to discuss and agree the initial known barriers to integration (based on literature review and industry experience and which addresses the

research question B) and to establish the groundwork for the survey to be conducted with a wider audience.

2. Focus Group 2 (FG2) — The FG2 session was devoted to reviewing the results of the survey, understanding what the results meant, and discussing surprises from the survey and how they might challenge contemporary thinking with regards to integration assumptions and practices. Finally the group defined a new set of functional requirements constructed by combining the Hasselbring integration dimensions with the feedback from the survey. The overriding task was to theorize what an integration platform might need now and in the future to achieve business at the speed of thought (Hasselbring, 2000).

3. Focus Group 3 (FG3) — The FG3 session reviewed the research findings on current integration platforms and identified a gap analysis in line with the set of functional requirements that were created as a result of FG2. FG3 was used to discuss the gap and potential solutions to the shortfalls, and as a base to design a next-generation integration platform.

4. Focus Group 4 (FG4) — Once the next-generation integration platform was designed, the final focus group explored its suitability as a core Internet service for the greater Internet community.

#### 3.1.3.2 Survey

The primary form of quantitative data collection used was a questionnaire loosely based on a highly modified version of the "Barriers to e-government integration" questionnaire (Lam, 2005), with questions based on feelings towards barriers to integration segmented by different levels of governance.

The objective of the survey was to get an indication of where the market sits with regards to integration solutions relative to the current technological trends and approaches. In addition, it sought to identify some attitudes towards particular aspects of integration which might determine whether certain approaches are welcome.

The survey consisted of four sections.

1. *Section A — General.* The general section asked questions that could be answered by all strata.
2. *Section B — Strategy.* The strategy section addressed those persons in an organization who hold CxO roles, e.g. CEO, CIO, CTO.
3. *Section C — Business.* The business section addressed those persons in an organization who hold a position involving business or operations, including but not limited to sales, marketing and logistics roles. Typically business roles would constitute the customer or user of an integration implementation solution.
4. *Section D — Technology.* The technology section was aimed at those persons involved in the implementation of integration solutions and included architects, developers and system administrators. The majority of questions posed were in the technology area.

#### 3.1.3.3 Interviews

Throughout the research process certain individuals presented more value due to their increased involvement in the world of software integration. It therefore made sense to conduct a deeper assessment of their opinions regarding integration and the potential for a global solution to integration problems.

Interviews consisted of a semi-structured one-on-one meeting with the participant, which involved asking a series of open-ended questions and offering the participant the freedom to express any additional thoughts or concerns they held on the subject matter. Feedback from the participant sometimes led to additional questions being asked. All information gained added qualitative value to that provided by the focus groups and key thoughts were fed into the focus groups for consideration.

#### 3.1.3.4 Technology Review

The purpose of the technology review was to use the insights gained during the course of this research (literature review, focus groups, surveys and interviews) and apply the set of evaluation criteria constructed against known integration platforms, thus determining weaknesses that might exist when applied over a wide area network environment on the scale of the global Internet.

Insights gained during this process helped to identify critical points of weakness which must be addressed before an integration platform is successful at scale.

#### 3.1.3.5 Prototyping

In considering current industry trends and factoring in focus group and survey data, this thesis will lay out a potential solution by way of a prototype design and thus addressing research question C, what would a platform designed for "business @ the speed of thought" look like? The prototype will then be factored into the final focus group discussions to help assess validity and potential for success. Identified failings of the prototype will be added as areas for future analysis.

### 3.1.4 Data Identification and Analysis

As this research contained both quantitative and qualitative research techniques, significant efforts were made to protect participant confidentiality and data privacy.

For quantitative research, rules were set to ensure anonymity, e.g. a minimum of 20 participants were required for each subset of data. The data from the questionnaires was collated, summarized and anonymized using MINITAB. Boxplots were used to compare the sets of data overall and also by strata (strategy, business and technology).

For each question the following analysis was conducted:

- base statistical analysis — including mean, median, range and variance for each question
- non-parametric sign testing to determine significance of indicators.

For the qualitative component, a balance was established during the interviews to ensure richness of qualitative data while protecting participants from exposure to risk. Any indication of an individual's identity or their place of work was carefully removed from the study. Each participant was given the opportunity to review sections relating to their interview to ensure accuracy, and to review the conclusions drawn from their answers.

## 3.2 Justifications

### 3.2.1 Action Research

A global integration platform is the culmination of many converging trends. Integration, Internet standards, cloud infrastructure and consolidation of technology approaches are all factors involved in a global integration solution. The complexity of this environment requires the consideration of an enormous number of points and impacts. Action research helps ensure ongoing research continues to reap rewards. Additionally, an iterative approach (in true agile style) ensures there is always a working-

set solution available, something that is imperative if the commercial world is to consider adopting it as a core infrastructure service.

### 3.2.2 Alternative Research Methods

This research was conducted using a pragmatic approach by choosing the method that appears to best suit the problem area. Primarily this research was conducted using qualitative methods including a survey, focus groups and interviews with subject matter experts due to the complex nature of the field involved. Only a small number of technologists and business people are actively involved in the field of integration, and focusing are those that are helped ensure that findings were of a high quality.

While a quantitative approach would have been valuable such as measuring the timeframes involved in integration projects, there was not enough time available for uncovering suitable projects and monitoring them for the duration of their projects. Additionally this approach would also require in-depth insight to determine whether delays incurred with integration were due to the technology itself or whether the business involved had operational issues that prevented progress occurring. Consequently, it was best to discuss integration with key resource who could accurately summarize their experiences.

### 3.2.3 Determining Integration Platforms / Frameworks

The literature review pointed to a number of key technologies including CORBA and web services. Some platforms were ruled out because they were proprietary and required the adoption of closed-source standards.

### 3.2.4 Stratifying at Business Level

During early discussions with participants, it became apparent that there was a pre-existing assumption that each level of the business would blame another level for deficiencies in the implementation of integration solutions. As a sanity check, it was important to include non-technical roles in the survey, to test this assumption and to investigate whether a significant barrier to integration is, in fact, alignment of the business levels rather than any specific work involved or technical expertise issue.

### 3.2.5 Stratifying Against Business Size

Similarly, during initial discussions it emerged that there was a preconceived assumption that organizations of different scale conduct integration in a different manner. For example, when an organization is large enough, and has an in-house research and development unit, it can design and develop all required solutions internally and does not need to integrate with a third party. It was important to test the validity of this thinking.

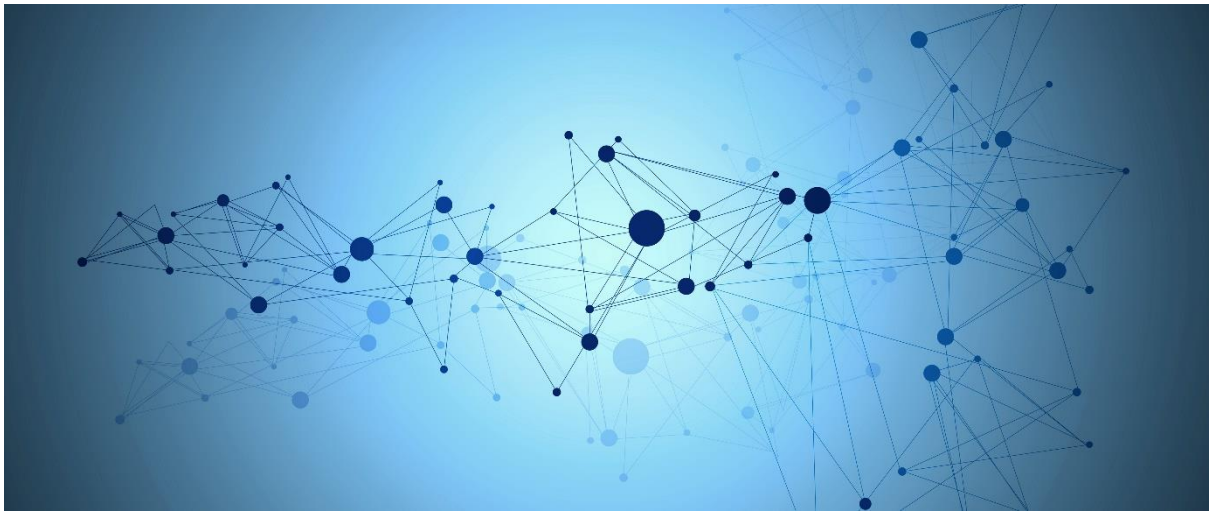
## 3.3 Conclusion

In this chapter we have discussed the research methodology used in this study to investigate specific barriers to enterprise integration and potential pathways for resolution. We introduced the target population, the sampling method and discussed how data was collected and analyzed. We also discussed the justification for this research approach and some considerations for the segmentation of data.

In the next chapter "Results" we will explore the results in full and identify which are statistically significant.

# 4

# Results



Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In the previous chapter we discussed the research methodology used in this study to investigate specific barriers to enterprise integration and potential pathways for resolution. In this section, we report the findings of the research based upon the staged methodology outlined. As well as statistical analysis, we review the input gained during the focus groups, interviews and platform reviews. In some cases, data has been summarized and additional detail can be found in the appendices.

## 4.1 Focus Group 1 (FG1)

The initial focus group started by reviewing the research topic introduction, followed by an assessment of the immediate reaction of whether the group felt the concept of a ubiquitous global integration platform was feasible.

Initial feelings were mixed with some doubting any platform could offer enough benefits to draw developers away from proprietary standards, technologies and associated commercial solutions. Those on the other side of the fence indicated that the same argument was put forward when local networks were considering whether to become part of the emerging Internet.

While the negative side of the argument is immediately apparent given today's commercial pressures, the advantages require some forward thinking and a journey of discovery. The focus group agreed to defer final judgement and to begin that journey by constructing a survey which would test the integration waters and hopefully provide insight into how the market conducts integration today, including attitudes and trends.

The output from the first discussion group, when combined with the literature review, led to the development of the following survey as outlined in the methodology.

*Section A – General.* The general section asked questions that could be answered by levels of the business and was useful in assessing overall attitudes towards problems with integration and which levels of the business were most problematic. Each participant identified themselves as either a strategy, business or technology business person, which determined whether they answered Section B, C or D.

*Section B – Strategy.* The strategy section, in accordance with Lam's survey asked participants about key strategy barriers including the ability to share goals and objectives, ambitious milestones, lack of ownership, lack of implementation guidelines and funding concerns. Here we also provide the candidate the opportunity to raise other strategic barriers to integration (Lam, 2005).

*Section C – Business.* The business section asked questions about key known business barriers including data ownership, policy evolution, lack of business readiness, pace of integration adoption, absence of a champion and common terminology. Business candidates also were given the opportunity to raise other business barriers to integration.

*Section C – Technology.* The technology section contained the bulk of the questions as it represented the most significant area of integration concern. To begin with, participants were asked about known technology barriers including architecture interoperability, incompatible data standards, incompatible technical standards, different security models and inflexibility of legacy systems. Participants were again given the opportunity to raise other technology barriers to integration.

From here the survey expanded into more focused questions about current practices and which standards and technologies were in play. This included questions about SOAP, XML, COTS integration, bespoke integration, business intelligence, transformation technologies, REST, service oriented architecture and asynchronous messaging.

The survey also asked technical participants a special question on whether or not they would adopt an open-source common ontology if one existed. This was critical to determine whether the world was ready for this or not.

Finally the survey asked technical participants three questions to test their general approach to development. If they were a modern new-age developer using NoSQL databases, or if they were more traditional in their approach e.g. maintaining schemas for data models.

Collectively the answers to the survey provided the insight required for the focus group to continue the investigation into integration platforms.

## 4.2 Survey Results and Focus Group 2 Analysis

Results were grouped, analyzed and tested in an attempt to find significant indicators of attitudes towards integration.

### 4.2.1 General Results from Survey Section A

*A2. What is the size of your company?
*A7. Which level of business do you most strongly associate with?

| | Strategy | Business | Technology | All |
|---|---|---|---|---|
| Small (1-19) | 13 | 10 | 19 | 42 |
| Medium (20-999) | 5 | 15 | 31 | 51 |
| Large (1000-5000+) | 7 | 14 | 26 | 47 |
| All | 25 | 39 | 76 | 140 |

*Table 5: Results – Company Size vs Business Level*

The major component of the survey is technical in nature as the objective of this research is to design a global integration platform. However, there is some merit in balancing the technical arguments against strategic and business perspectives.

Overall there were 140 responses from various business sizes as shown in Table 5. Although not statistically significant in the global population of business people involved in integration, the survey does provide a useful indication of real-world sentiment towards integration attitudes and trends.

*A3. Does your company (or on behalf of a client) integrate information with a third-party system?

| Company size | No | Yes | All |
|---|---|---|---|
| Small (1-19) | 11 | 31 | 42 |
| Medium (20-999) | 9 | 42 | 51 |
| Large (1000-5000+) | 8 | 39 | 47 |
| All | 28 | 112 | 140 |

*Table 6: Results – Company Size vs Integrate with a Third Party*

The results in Table 6 do not indicate any significant difference in rates of integration at different business levels, most likely because the sample size is too small. However, as an indication we can see that medium-sized businesses appear to have the highest level of integration occurring.

*A4. Compared to five years ago, do you integrate with fewer, more or the same number of systems?

*A5. Compared to five years ago, is it less, more or equally difficult to integrate systems/solutions?

*A6. Are you less, more or equally concerned (the same) with where your information is stored than you were five years ago?

| | N | N* | Below | Equal | Above | P | Median | |
|---|---|---|---|---|---|---|---|---|
| A4 - System number | 126 | 14 | 10 | 26 | 90 | 0.0000 | 3.000 | S |
| A5 - Ease of integration | 130 | 10 | 48 | 44 | 38 | 0.8822 | 2.000 | |
| A6 - Data storage | 130 | 10 | 15 | 47 | 68 | 0.0000 | 3.000 | S |

*Table 7: Results – Sign Test for Median: System Number, Ease of Integration, Data Storage*

In testing questions A4 through A6, we applied a sign test with a level of significance $\alpha=0.05$ to determine whether the answers indicated a significant increase in occurrence compared with five years ago.

- $H_0$: the median is 2 (the same)
- $H_1$: the median > 2 (more) is positive $\alpha=0.05$

Because the p-value for both questions A4 and A6 shown in Table 7 is less than the level of significance $\alpha=0.05$ we can reject the null hypothesis that the median is equal to 2, and accept the alternative hypothesis with 95% confidence that a greater number of systems are integrated today compared with five years ago. Additionally we can accept with 95% confidence that there exists greater concern today about where information resides than five years ago.

In the case of question A5, we acknowledge that the responses are balanced. We, therefore, accept the null hypothesis in this case indicating that integration is no more difficult than it was five years ago.

The focus group agreed with these findings but also raised the point that transitioning to the cloud is difficult, whereas starting in the cloud is easier. The unwillingness to give up current operating models can be a significant barrier to cloud adoption and therefore online integration solutions. One possible

solution in these scenarios is to move legacy applications into the cloud. However, this is only successful in some cases due to technological constraints.

The focus group also raised the point that cloud integration may daisy chain from data providers. Although an organization may respond that they do not integrate with the cloud, they may integrate with a partner who data mines in the cloud, resulting in a cloud-based integration scenario. This may appear like quibbling over semantics but it does raise the possibility that organizations may be moving closer to the cloud despite not taking any internal steps to do so.

The focus group also considered the provider network's geographic location. Larger companies may have a 'private cloud' set-up on the premises which may not be considered cloud integration, yet the solution exhibits similar behaviour.

With these considerations in mind, it should be acknowledged that any global integration platform must interoperate with both cloud and non-cloud networks to bridge the gap between modern and legacy solutions.

*A8. From your perspective, how often is 'strategy' a barrier to integration?

*A9. From your perspective, how often is 'business' a barrier to integration?

*A10. From your perspective, how often is 'technology' a barrier to integration?

| | N | N* | Below | Equal | Above | P | Median | |
|---|---|---|---|---|---|---|---|---|
| A8 - Strategy a barrier (frequency) | 129 | 11 | 21 | 54 | 54 | 0.0001 | 3.000 | S |
| A9 - Business a barrier (frequency) | 130 | 10 | 18 | 47 | 65 | 0.0000 | 3.500 | S |
| A10 - Technology a barrier (frequency) | 133 | 7 | 33 | 51 | 49 | 0.0488 | 3.000 | S |

*Table 8: Results – Sign Test for Median: Strategy, Business and Technology (Frequency)*

In testing questions A8 through A10, we applied a sign test with a level of significance $\alpha=0.05$ to determine whether the answers indicated a significant increase in occurrence compared with five years ago.

- $H_0$: the median is 3 (the same)
- $H_1$: the median > 3 (more) (frequently or more a problem)
- $\alpha=0.05$

Because the p-values in the three questions shown in Table 8 are less than the level of significance $\alpha=0.05$ we can reject the null hypothesis that the median is equal to three, and accept the alternative hypothesis. We can be 95% confident that 'strategy', 'business' and 'technology' all contribute frequent barriers to integration and that no level of a business is without fault.

The focus group discussed this finding and agreed with the outcome but did make some interesting points. While these survey questions determine if barriers occur at different business levels, they do not test for dependence — for example, strategy barriers such as funding may dilute if integration solutions are easier and therefore cheaper to implement.

Additionally the focus group agreed and identified that integration touches all levels of business. Not only are there individual barriers to integration within each level, but alignment between the three levels can potentially create further barriers.

*A11. Please rank the business levels in order from the greatest overall barrier to integration to the smallest barrier to integration.
*A12. Please rank the following three challenges in order from most difficult to least difficult to address.

| | N | N* | Below | Equal | Above | P | Median | |
|---|---|---|---|---|---|---|---|---|
| Strategy a barrier (overall) | 139 | 1 | 47 | 51 | 41 | 0.5940 | 2.000 | |
| Business a barrier (overall) | 140 | 0 | 46 | 60 | 34 | 0.2188 | 2.500 | |
| Technology a barrier (overall) | 140 | 0 | 47 | 29 | 64 | 0.1288 | 2.000 | |

*Table 9: Results – Sign Test for Median: Strategy, Business and Technology (Overall)*

To test questions A11 and A12, we applied a sign test with a level of significance $\alpha=0.05$ to determine whether the answers indicated a significantly more problematic barrier to integration.

- $H_0$: the median is 2 (the same)
- $H_1$: the median ≠ 2 (more) (more problematic)
- $\alpha=0.05$

Because the p-values in the three cases shown in Table 9 are greater than the level of significance $\alpha=0.05$ we can reject the alternative hypothesis and accept the null hypothesis that the medians are equal to two.

Therefore, we can be 95% confident that 'strategy', 'business' and 'technology' are all equally responsible for providing barriers to integration.

The focus group found this result slightly mystifying and felt it would make sense that 'business' was the biggest blocker. The group termed 'technology' an enabler, with 'strategy' mostly decision making, leaving 'business' to navigate the complexities of engagement.

The focus group felt it was possible that integration barriers in other areas of the business were not always transparent and, therefore, it was difficult to judge the value of those barriers against barriers that existed in a participant's own business level.

This sentiment reinforced the need for integration solutions to be designed and implemented by a team represented by all levels of the business. Doing so would ensure all stakeholder concerns would be addressed and problems encountered would be visible to all levels. It would also prevent an inefficient integration from being initiated, rather than being recognized mid-implementation.

### 4.2.2 Strategic Results from Survey Section B

| Metric | N | N* | Below | Equal | Above | P | Median | |
|---|---|---|---|---|---|---|---|---|
| Strat - Share goals | 22 | 113 | 8 | 7 | 7 | 1.0000 | 3.000 | |
| Strat - Milestones | 22 | 113 | 7 | 3 | 12 | 0.3593 | 4.000 | |
| Strat - Ownership | 22 | 113 | 8 | 5 | 9 | 1.0000 | 3.000 | |
| Strat - Guidelines | 21 | 114 | 4 | 7 | 10 | 0.1796 | 3.000 | |
| Strat - Funding | 22 | 113 | 6 | 7 | 9 | 0.6072 | 3.000 | |

*Table 10: Results – Impact of Strategic Barriers on Integration*

In testing question B1, we applied a sign test with a level of significance $\alpha=0.05$ to determine whether the individual strategy barriers indicated a less-than-occasional or more-than-occasional barrier to integration.

- $H_0$: the median is 3 (occasional)
- $H_1$: the median ≠ 3 (more) (more or less than occasional)
- $\alpha=0.05$

Results in Table 10 showed that none of the individual strategic integration barriers represented significantly high or low consideration. It should be acknowledged that only those participants with a 'strategy' role in business completed this section of the survey (of which there were 22). This number is small and the set of data would likely benefit from a larger sample size.

The focus group felt all of the barriers represented obstacles within specific business contexts. Strategic decisions about integration were less about how to achieve it and more about whether the business case stacked up.

One of the members of the focus group highlighted a case involving the merger of two telecommunications companies where the decision was made to integrate with little consideration given as to how it would be achieved — leaving the business and technology teams to solve the problem.

Additionally, as a matter of practice, it was felt that the 'strategy' level of the business may not share goals with 'technical' or 'business' due to commercial sensitivity in the initial stages of engagement, until the decision was made to expose details. This reinforces the need to include leaders from each level of the business within the trusted circle during engagement.

* B2. Are there other strategic factors which present a barrier to integration?

The focus group raised a number of important strategic barriers to integration including the following:

1. Lack of executive knowledge around technology. While integration solutions may be relatively well understood within technology-based companies, they are not necessarily as well understood by businesses in other sectors. Lack of technical understanding can lead to poor decision making around integration.
2. Vendors are a consideration. For companies that outsource integration components (e.g. server hosting or middleware development) to third-party vendors there are additional risks as vendors may not fully understand the integration context and its inherent needs.
3. Accessibility of team members. Similarly, the more separate the parties that are integrating, the greater the challenge of aligning resources including team members. Often business and technology teams have a roadmap and getting them to align internally is difficult; even more so if a third party is involved.
4. Regional, language and cultural differences can become barriers if the integrating parties are located in different geographic regions.
5. Industry barriers. Similarly when integrating entities have a background in different industries, achieving strategic alignment can become more of a challenge.

### 4.2.3 Business Results from Survey Section C

| Metric | N | N* | Below | Equal | Above | P | Median | |
|---|---|---|---|---|---|---|---|---|
| Bus - Data ownership | 38 | 100 | 11 | 14 | 13 | 0.8388 | 3.000 | |
| Bus - Policy | 36 | 102 | 14 | 12 | 10 | 0.5413 | 3.000 | |
| Bus - Readiness | 38 | 100 | 9 | 13 | 16 | 0.2295 | 3.000 | |
| Bus - Integration pace | 39 | 99 | 6 | 15 | 18 | 0.0227 | 3.000 | S |
| Bus - Champion | 37 | 101 | 11 | 9 | 17 | 0.3449 | 3.000 | |
| Bus - Terminology | 39 | 99 | 12 | 12 | 15 | 0.7011 | 3.000 | |

*Table 11: Results – Impact of Business Barriers on Integration*

In testing question C1, we applied a sign test with a level of significance $\alpha=0.05$ to determine whether the individual business barriers indicated a less-than-occasional or more-than-occasional barrier to integration.

- $H_0$: the median is 3 (occasional)
- $H_1$: the median ≠ 3 (more) (more or less than occasional)
- $\alpha$=0.05

Because the p-value for the pace of integration, shown in Table 11, is less than the level of significance $\alpha$=0.05 we can reject the null hypothesis in this case that the median is equal to 3, and accept the alternative hypothesis with 95% confidence that the mean is not equal to 3. In this individual case, we further tested this question to determine if the median is > 3.

- $H_0$: the median is 3 (occasional)
- $H_1$: the median > 3 (more) (more than occasional)
- $\alpha$=0.05

With a p-value of 0.0113 against a level of significance $\alpha$=0.05 we can be 95% confident that the pace of integration is a more-than-occasional barrier to integration.

For the remainder of the business barriers, we must accept the original null hypothesis and therefore acknowledge that the barriers are occasional barriers to integration.

The focus group agreed the pace of integration was problematic, but also felt other business barriers were significant. It was expressed that a business is never truly ready for integration because of the large amount of diversity with integration solutions. However, there was some acknowledgement that you can prepare by implementing an EA framework, which allows integrating parties to become familiar with the complete challenge (not just technical) before initiating the project.

The focus group also felt there was a growing appreciation for applying a 'minimum viable integration' approach to conducting integration. This would involve the implementation of multiple phases, the first of which would only deliver against critical milestones. It was conceded this approach has a greater chance of success given the scalable nature of cloud-based technologies, e.g. Amazon or Azure.

Similarly the group raised the point about whether business would resource dual systems in parallel or choose to take a plunge approach to integration and deal with problems post-implementation. This decision came down to the inherent business risks involved with a failed implementation.

### 4.2.4 Technology Results from Survey Section D

| Metric | N | N* | Below | Equal | Above | P | Median | |
|---|---|---|---|---|---|---|---|---|
| Tech - Arch | 63 | 77 | 13 | 28 | 22 | 0.1877 | 3.000 | |
| Tech - Data | 65 | 75 | 10 | 21 | 34 | 0.0002 | 4.000 | S |
| Tech - Tech | 65 | 75 | 14 | 27 | 24 | 0.0717 | 3.000 | |
| Tech - Security | 64 | 76 | 14 | 20 | 30 | 0.0113 | 3.000 | S |
| Tech - Legacy | 64 | 76 | 10 | 15 | 39 | 0.0000 | 4.000 | S |

*Table 12: Results – Technology Integration Barriers*

In testing question D1, we applied a sign test with a level of significance $\alpha$=0.05 to determine whether the individual technology barriers indicated a more-than-occasional barrier to integration.

- $H_0$: the median is 3 (occasional)
- $H_1$: the median > 3 (more) (more than occasional)
- $\alpha$=0.05

Because the p-values for data, security and legacy barriers shown in Table 12 are less than the level of significance $\alpha$=0.05 we can reject the null hypothesis in these cases that the median is equal to 3, and

accept the alternative hypothesis with 95% confidence that these barriers occur more than occasionally (i.e. frequently or very frequently).

In the case of architectural design and technology barriers, we accept the null hypothesis that they do not occur more than occasionally.

The focus group felt this was correct, stating that security is always problematic as hackers are continually undermining secure solutions, which must therefore adapt to stay one step ahead. This means security solutions are, by necessity, very diverse, and thus it is difficult to prepare for every integration scenario.

Regarding legacy systems, the focus group felt they often linger within an organization because the emphasis is often on how to implement solutions with little thought given to how to retire a solution further down the track. For example, it is difficult to transition customer bases once a solution becomes entrenched.

Finally with regards to data ownership, the group agreed this is an ongoing challenge due to the lack of a common data model and therefore common data storage mechanisms.

| Variable | X | N | Sample p | 95% Lower Bound | P-Value (>0.5) | P-Value (<0.5) | |
|---|---|---|---|---|---|---|---|
| Use XML | 50 | 69 | 0.724638 | 0.622646 | 0.000 | 1.000 | S+ |
| Cons/Pub REST | 39 | 69 | 0.565217 | 0.459159 | 0.168 | 0.886 | |
| Middleware | 43 | 69 | 0.623188 | 0.517340 | 0.027 | 0.985 | S+ |
| SOA | 26 | 65 | 0.400000 | 0.297527 | 0.959 | 0.068 | |
| Asynchronous | 28 | 67 | 0.417910 | 0.315723 | 0.929 | 0.111 | |
| Cloud Integration | 46 | 68 | 0.676471 | 0.571213 | 0.002 | 0.999 | S+ |
| Best-Practice Model | 48 | 67 | 0.716418 | 0.612230 | 0.000 | 1.000 | S+ |
| Rules Engine (Auto) | 12 | 69 | 0.173913 | 0.103530 | 1.000 | 0.000 | S- |
| Transform Technology | 26 | 68 | 0.382353 | 0.283539 | 0.981 | 0.034 | S- |

*Table 13: Results – Technologies and Approaches*

In testing questions D7 through D15, we applied a sign test with a level of significance $\alpha=0.05$ to determine whether the rate of individual integration technologies and approaches applied was significantly high or significantly low.

Test 1 (median = 0.5 vs median > 0.5)

- $H_0$: the median is 0.5 (occasional)
- $H_1$: the median > 0.5 (more) (more or less than occasional)

Test 2 (median = 0.5 vs median < 0.5)

- $H_0$: the median is 0.5 (occasional)
- $H_1$: the median < 0.5 (more) (more or less than occasional)

The results for these two tests are shown in Table 13.

According to Test 1 and with a p-value < level of significance $\alpha=0.05$ we can be 95% confident that XML, middleware and cloud integration are applied at significantly high levels.

According to Test 2 and with a p-value < level of significance $\alpha$=0.05 we can be 95% confident that business intelligence engines and transformation technologies are not applied at significantly high levels.

Finally, 48 out of 67 participants responded that if a non-proprietary or open-source, best-practice data model existed for their industry, they would adopt it. This response is a critical indicator against Hasselbring's heterogeneity integration problem dimension, and it was refreshing to see that technologists understood the benefits that come with standardizing data models and that they outweighed the overheads (Hasselbring, 2000).

The focus group reviewed these results and added some important considerations:

1. Cloud adoption. While a movement to the cloud might be occurring in New Zealand and Australia, larger-scale organizations elsewhere have considerable capital tied up in private networks and therefore the rate of cloud adoption may decline at scale. An example given was that Azure was not doing as well in the USA as it was elsewhere.
2. It is tricky to enforce the uniformity of a common data model; however, it was determined that laying a pathway to a common model may be possible and would solve numerous integration problems.
3. JSON, along with REST, is being used these days as a 'quick and dirty' way to get data to and from a website in preference to SOAP. However, because it does not describe itself to an external system, i.e. lacks semantics, it has limitations. It was felt that integration between services should be XML with types and transformation technology.
4. XSLT is not so easy to implement and is effectively an enhancement to XML (i.e. XML is a prerequisite technology). Current trends involve wrapping transformation technology with language-specific implementations to deal with transformation. This is similar to the way RPC and proxy implementations hide service connectivity from the caller, addressing Hasselbring's distribution problem dimension (Hasselbring, 2000).
5. Versioning of system interfaces and data models is an important consideration for any integration platform.
6. Pattern management is important at all levels of the business. Use-cases should be implemented at a higher level than just data model and data transformations.
7. There is a trend, e.g. Google, where responses are mined regardless of the source data model. Google has an algorithm that can determine relevance and this exhibits another potential solution to mining data from diverse sources and rendering in a common way.
8. It is important to support both push and pull integration approaches.

## 4.3 Focus Group 2 (FG2)

In addition to the commentary provided with the survey results, the focus group was tasked with delivering an integration assessment template which would help to assess the ability of known integration platforms to operate at scale. In the process of producing this template, a number of useful points were raised.

The focus group discussed one of the cornerstone issues involved with integration — the absence of a common data model and terminology, which results in numerous interoperability barriers. Hasselbring defines this at length in his research as part of the heterogeneity problem dimension (Hasselbring, 2000).

The focus group noted there may be an emerging opportunity with the Internet of Things (IoT) — a movement that encompasses all levels of business — to standardize common terminology and enforce its use across the IoT domain. This opportunity is rare because of the diverse nature of the monitored

devices involved and the sheer volume at which they communicate through common IoT interfaces. IoT may become a driving force behind consolidation of data models as the need for interoperability increases.

The focus group also raised the point that to achieve a global integration network, there should be a central database for common account and credential information that could be reused throughout the integration mesh. A point of contention developed as to whether centralized credential management constituted a requirement for governmental oversight. The focus group felt that to have a centralized credential, a government would need to approve the approach, because to participate in the network, there would have to be an acceptance of single sign-on credentials with heavy encryption throughout the network to ensure data integrity remained secure.

Transmissions should be atomic and all data should be tied to a session window only, so hacking of transaction payloads would not render anything useful that would compromise an account.

At a business level, ease of operation was a key concern. Modelling, reporting, analytics, dashboard and route management support were raised as critical requirements at a business administration level.

By combining principles established in the literature review with observations made during FGs 1 and 2, the focus group established a table of assessment criteria for a global integration platform as seen in Table 14 below.

| | **Feature matrix** | | **Benefit** | | |
|---|---|---|---|---|---|
| **Dimension** | **Code** | **Requirement** | **Strategy** | **Business** | **Technology** |
| *Distribution* | DIS-A-1 | Scalable distribution | Cost optimization and competitive advantage | Seamless business continuity | One implementation fits all levels of scale |
| | DIS-B-1 | Proxy services | Reduced risk/cost | Greater business continuity | Simplified code base with reduced technological complexity |
| | DIS-B-2 | Endpoint agnostic | Reduced risk/cost | Greater business continuity | Does not require re-deployment when model changes |
| | DIS-B-3 | Service discovery | Ability to optimize cost | Ability to optimize functionality | Ability to optimize performance and easier introduction of system upgrades |
| | DIS-B-4 | Service reuse | Reduced risk/cost | Greater business continuity | Reduction in development effort |
| *Autonomy* | AUT-A-1 | Balanced autonomy | Reduced risk/cost | Greater business continuity | Reduced fallout from failure scenario |
| | AUT-A-2 | Intelligent nodes | Able to execute competitive advantage and deliver business outcomes | Greater level of business response | Ability to extend solutions without redeployment |
| | AUT-A-3 | Virtualization | Cost reduction | Ability to scale operations to meet different demand levels | Improved reliability |
| | AUT-B-1 | Resilient to organizational change | Can make organizational change with reduced risk of negative impact on the business | Can make business change with reduced risk of negative impact on the business | Greater ability to absorb change, leading to greater development productivity |
| | AUT-B-2 | Business intelligence | Reduction in cost | Ability to automate business | Centralized business logic leads to simplified |

| | | | | | |
|---|---|---|---|---|---|
| | | | | intelligence leading to increased quality | software design which is easier to maintain |
| | AUT-B-3 | Complex event processing | Able to execute competitive advantage and deliver business outcomes | Improved manageability and security<br><br>Greater level of business response | Simplification of software structure<br><br>Ability to adapt quickly to different external environments |
| | AUT-B-4 | Service composition | Ability to develop new function combinations rapidly | | |
| | AUT-B-5 | Model-driven implementation | Strategic competitive advantage | Ability to deliver new functionality rapidly | Ability to develop functionality faster |
| *Heterogeneity* | HET-B-1 | Model agnostic | Reduced risk/cost | Greater business continuity | Does not require re-deployment when model changes |
| | HET-B-2 | Solution agnostic | Reduced risk/cost | Greater business continuity | Does not require re-deployment when another solution is integrated |
| | HET-C-1 | Supported standards | Increased strategic opportunities through merger and acquisition | Increased business continuity | Standardization helps reduce complexity |
| | HET-C-2 | Wrappers | Organizational flexibility | Increased business continuity | Increased quantity of integration options |
| | HET-C-3 | Service | Organizational flexibility | Improved information flow | Ability to expose internal functionality |
| | HET-C-4 | Messaging | Increased competitive advantage through integration | Seamless business integration | Ease of technical integration |
| | HET-C-5 | Message security | Increased security and business integration confidence | Messages don't leak sensitive company information | Messages don't leak sensitive company information |
| | HET-C-6 | Message monitoring | Increased security and business integration confidence | Monitoring and administration of message flows | Ability to maintain connected system links |
| *Architecture* | ARC-A-1 | Reliability | Reduced risk to the business | Improved quality of integration solutions and increased business continuity | Guaranteed message delivery provides more robust technical solutions |
| | ARC-A-2 | Availability | Reduced risk to the business | Improved quality of integration solutions and increased business continuity | With ~100% uptime, messages are always delivered |
| | ARC-A-3 | Maintainability | Configurable service registry change management | Increased business continuity | Reduction in maintenance costs |
| | ARC-A-4 | Testability | A testable solution reduces strategy implementation risk | A testable solution reduces business implementation risk | Testable integration means easier fault analysis and solution tuning |

| | | | | | |
|---|---|---|---|---|---|
| | ARC-A-5 | Ease of deployment | Reduction in strategic cost/risk | Reduction in business operating cost/risk | Reduction in operations-related development |
| | ARC-A-6 | Ease of administration | More agile strategic decision making | Hot configurable means increased business continuity | Improves ability to administer and maintain the system |
| | ARC-A-7 | Debug-ability /monitoring | More agile strategic decision making | Faster fault analysis improves business continuity | Ability to trace transactions in and out of a service will help fault resolution |
| | ARC-B-1 | Security | IP is protected and security maintained | Authorization and access are managed for both sender and receiver | Allows for accurate routing of messages to and from the right entities |
| | ARC-C-1 | Portability | Reduced risk to the business | Removes risk involved with acquisition decision | Reduced emphasis on specific technical solution |
| | ARC-C-2 | Functionality / extensibility | Increased competitive advantage | Seamless business integration | Reduced need to expand solution set |
| | ARC-C-3 | Interoperability | Reduction in cost when engaging with another business entity | Faster rate of business integration | Less complex technical integration solutions<br><br>Easier to implement, easier to maintain |
| | ARC-C-4 | Usability | Increased competitive advantage | More efficient business practices | Reduction in maintenance costs |
| | ARC-C-5 | Reusability | Ability to leverage pre-existing solutions is a competitive strategic advantage | Improves return on investment | Reduces required development time for a solution |
| | ARC-D-1 | Efficiency | Lighter, more efficient integration provides cost savings | Faster and lighter integration results in greater business outcomes | More efficient integration means reduced load on servers |
| | ARC-D-2 | Scalability | Infinitely scalable integration networks allow for even the most complex mergers, acquisitions or joint ventures | Scalability allows for increased scope for business integration solutions | Removes technical barriers to scale up solutions. Enables more ambitious projects |
| | ARC-D-3 | Development productivity | Reduction in cost | Faster implementation of products and services | More productive development unit |

*Table 14: Global Integration Platform Assessment Criteria*

## 4.4 Existing Integration Platform Review

The assessment criteria table (Table 14) was used as the basis for the integration platform review. Each platform went through some literature review and source code review if it was available. Additionally, those platforms that were accessible without a commercial licence were installed and tested to confirm assertions made during the literature and source code review sections.

In summary, the platform scored a point if it supported a feature, which addressed one of the problem dimension requirements outlined in table 14.

Example Score Chart:

| Dimension | Requirement | Rating | | Dimension | Requirement | Rating | |
|---|---|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | No | 5.5 | *Distribution* | Scalable distribution | No | 3 |
| | Business intelligence | Partial | | | Proxy services | Yes | |
| | Intelligent nodes | No | | | Endpoint agnostic | No | |
| | Complex event processing | Yes | | | Service discovery | Yes | |
| | Service composition | Yes | | | Service reuse | Yes | |
| | Asset wrapping | Yes | | *Architecture* | Reliability | Yes | 4.5 |
| | Virtualization | Yes | | | Availability | No | |
| | Model-driven implementation | Yes | | | Security | No | |
| | Balanced autonomy | No | | | Portability | No | |
| *Heterogeneity* | Model agnostic | No | 9 | | Functionality / extensibility | Yes | |
| | Solution agnostic | No | | | Interoperability | Partial | |
| | Structures | Yes | | | Usability | No | |
| | Supported standards | Yes | | | Maintainability | No | |
| | Wrappers | Yes | | | Efficiency | No | |
| | Service | Yes | | | Testability | Yes | |
| | | | | | Reusability | Yes | |
| | Messaging | Yes | | | Ease of deployment | No | |
| | Message control | Yes | | | Ease of administration | No | |
| | Message transformation | Yes | | | Scalability | No | |
| | Message security | Yes | | | | | |
| | Message monitoring | Yes | | | | | |

Cloud environments using virtual resources were assessed using a commercial licence donated by a local software development house.

**Note**. Following is a shortened review, for the full review please see Appendices C - G

### 4.4.1 Review – CORBA (OMG)

The 'Common Object Request Broker Architecture' (CORBA) is a standard defined by the Object Management Group (OMG) and provides users with a language and platform-neutral remote procedure call (RPC) specification.

There is a lot to like about what CORBA achieves, as indicated in Table 15, when addressing Hasselbring's heterogeneity and autonomy problem dimensions as it elegantly handles the mapping of objects and provides interfaces and controlled functionality across language boundaries to the outside world (Hasselbring, 2000).

| Dimension | Rating | Dimension | Rating |
|---|---|---|---|
| *Autonomy* | 4.5/9 | *Distribution* | 4/5 |
| *Heterogeneity* | 9/11 | *Architecture* | 4/14 |

*Table 15: CORBA Summary Assessment Table*

Unfortunately, CORBA suffers from a number of critical failings as an integration platform, and additional barriers that prevent it from scaling indefinitely (Henning, 2008) such as a lack of loose coupling and reliance on proprietary standards (see Appendix B for more information).

### 4.4.2 Review – Mapping Platform (Van de Maele)

Felix Van de Maele's mapping platform is insightful and helps to analyze and break down the usefulness of different forms of integration.

| Dimension | Rating | Dimension | Rating |
|---|---|---|---|
| *Autonomy* | 6/9 | *Distribution* | 1/5 |
| *Heterogeneity* | 11/11 | *Architecture* | 6/14 |

*Table 16: Van de Maele Summary Assessment Table*

Results in Table 16 indicate there are some similarities to OMG's platform. Like the model-driven architecture paradigm established by OMG, Van de Maele reiterates the importance of transforming via a platform-independent intermediary model — a critical requirement for a globally accessible integration platform. Both Van de Maele and OMG argue that the usefulness of the platform-independent model (PIM) is in its ability to be transported in its platform-neutral form, so it may be transformed elsewhere to suit its new environment.

Unlike CORBA, which translates from platform-neutral to platform-specific implementations, Van de Maele's mapping platform uses mappings which are language-, application- and paradigm-neutral end to end. The true value of reaching this level of neutrality, when combined with a centralized cloud-based mapping server, is in establishing a foundation for a common ontology where both service consumers and service providers will be required to transform to and from the model.

As discussed in depth by the focus groups and with support from the survey data, we can surmise that the industry is in need of a common ontology, but the motivation to create/adopt one is lacking. Van de Maele raises the possibility that there could not only be a centralized integration platform but also mappings could be shared and applied according to context in a community-based manner — in line with social media movements which increase the motivation to establish and extend a centralized ontology.

### 4.4.3 Review – Web Services (W3C)

The World Wide Web Consortium (W3C) is an international community that develops open standards for the World Wide Web, which help to promote its growth and evolution. The W3C has a vision for the web and provides many standards that constitute best-practice. However, the consortium is powerless to enforce these standards and thus proprietary standards often compete with and derail the W3C vision.

In the web services space, W3C describes a web service as an agent which implements a defined set of functionality and executes that functionality by sending a response message as the result of a request message. The functional definition of what the service can provide is published using a standard called the Web Services Description Language (WSDL); however, the method of implementation remains hidden.

Table 17 highlights the areas where web services perform well, specifically where scalability is involved. It should be noted, however, that the table reflects a best-case scenario. Web services are scalable depending on the caching policy they employ (or lack thereof).

| Dimension | Supports | Dimension | Supports |
|---|---|---|---|
| *Autonomy* | 6/9 | *Distribution* | 3/5 |
| *Heterogeneity* | 9/11 | *Architecture* | 13.5/14 |

*Table 17: Web Services Summary Assessment Table*

As Hasselbring identified, simple non-changing interfaces and payloads are the way forward when establishing a core infrastructure service such as FTP, SMTP, POP and, potentially, the global integration platform (Hasselbring, 2000). For that reason, the integration platform should have a representational state transfer interface (REST), which would allow a payload to be posted in raw format to the integration platform.

Web services scores very well on architecture primarily because services can be almost infinitely scaled with the right infrastructure. Web standards also enforce scalability security models which helps the dimension score more points.

It should be noted, however, that although web services scored well on autonomy, it is mainly due to it being resilient to changes that may affect autonomy. Web services still lacks the potential for collective intelligence, which would need to be housed elsewhere.

Despite these limitations, it should be acknowledged that, in general, web services are easy to scale, administer and use, and form the backbone of today's World Wide Web.

### 4.4.4 Review – Internet Service Bus (Microsoft)

The Internet Service Bus approach from Microsoft is mentioned less for its technical applications and more for its visionary qualities.

Microsoft's description of the Internet Service Bus begins by reinforcing the view shared by W3C that web services as portals into enterprise organizational functionality are the future of integration. However, Microsoft goes further by establishing the premise that end-user programming — being development effort that is ad hoc and unplanned — is actually moving us further away from achieving an interconnected world because the same models are being designed in many different ways worldwide and even sometimes within the same organization.

Furthermore, Microsoft states that software and application models are in the midst of a revolution where external service functionality can be deployed within a custom solution interface rather than requiring the user to leave an application and log into a cloud-based solution.

This concept that while logging into your accounting package, you can view your inventory levels despite inventory being managed by a third-party provider is a powerful notion in line with Bill Gates' initial vision.

To achieve this to a large degree, however, requires the accounting package to easily inter-connect with multiple inventory providers (not just one) so customers who might already have an inventory system are able to use a unified view.

To realize this we need a unified ontology for business applications where the concept of inventory is well defined, and inventory-based functionality is provided in a unified fashion — which is where Microsoft's point sinks in. While end-user programming is rife, we continue to diversify models, and therefore integration solutions — moving further away from an ontological approach which would foster greater and more seamless levels of integration.

Microsoft, however, remains optimistic and says that, eventually, software elements will merge with the cloud space using virtualized containers that house functionality, which can be passed from application provider to application provider to execute functionality. Given how long it has taken us to establish a common ontology, this is still likely to be some time away.

Pragmatically, in the short term, the Internet Service Bus is a form of encouragement for developers to decouple their business entity models and business intelligence from the solution interfaces they develop so they can be reusable — an important step towards a common ontology and a shared business intelligence.

### 4.4.5 Review – Apache NiFi and the HPCS

The NiFi platform is described by its makers as an easy-to-use, powerful and reliable system to process and distribute data. After investigating the product and reviewing the findings in Table 18, it is fair to say this summary accurately reflects the product's capabilities.

At the centre of NiFi's dataflow platform is its expression language, a platform-independent approach to business intelligence that can execute against data payloads as they pass through the NiFi engine. This is a significant advantage over CORBA and VDM, which mostly transform without the opportunity to centralize intelligence. The result is a hub-and-spoke solution as opposed to the point-to-point solution employed by CORBA and VDM.

| Dimension | Rating | Dimension | Rating |
|---|---|---|---|
| *Autonomy* | 7/9 | *Distribution* | 4/5 |
| *Heterogeneity* | 11/11 | *Architecture* | 12/14 |

*Table 18: NiFi Summary Assessment Table*

A clustered formation allows a single endpoint to support many NiFi servers, and clusters can be daisy chained together using an appropriate connection configuration. Unfortunately, there is a known limitation with this approach. Inputs that are generated server side cannot currently be distributed within the cluster set automatically and instead must be manually configured on a single node within the cluster. Apache acknowledge this lack of autonomy as a significant limitation, and they are working to address the limitation.

Deployment and administration of the NiFi platform are excellent with graphical user interface tools controlling the design of the dataflow process components and process groups, including support for templates, which can be shared. Again, however, a lack of scalability is evident. There is no centralized ACL or security layer capable of interoperating with multiple NiFi clusters. As such, the potential for a greater community to drive the platform forward is reduced.

### 4.4.6 Review – Amazon AWS Stack + Lambda

The Amazon stack (AWS) is a collection of services that form a cloud-computing platform. By combining virtualization technology to host virtual servers and web services with core Internet services such as domain name management and email management, Amazon's AWS provides a full stack that can be used to deploy complete business systems in the cloud.

One of the services Amazon provides is Lambda, which executes code in response to events. As we have previously discussed, the need to execute business intelligence is a critical requirement for a global integration platform, and Table 19 highlights the performance of the AWS stack + Lambda in doing so.

| Dimension | Rating | Dimension | Rating |
|---|---|---|---|
| *Autonomy* | 7/9 | *Distribution* | 1/5 |
| *Heterogeneity* | 11/11 | *Architecture* | 12/14 |

*Table 19: Amazon Stack Summary Assessment Table*

Lambda was originally Amazon's defence against container technology, enabling business intelligence to be executed in the cloud. Any deficiency in this critical piece would render the stack untenable for global integration. Fortunately, as Lambda is based on Node.js and therefore provides functional development support, it is more than up to the task. Lambda is able to combine this support with functionality from the Amazon stack including persistence and resource management.

The architecture for Amazon lends itself well to providing a foundation for a global integration platform. Currently the structure of AWS allows for accounts, which own and manage amazon services; but in the case of a global integration platform, services must be managed by the public. It may, again, appear unfair to label Amazon unscalable, but it is important we continue to acknowledge the operating context. Amazon does not provide (out-of-the-box) a way for members of the public to administer

Lambda rules against a single Lambda network with degrees of access control and the ability to distribute those rules appropriately within a network of Lambda nodes.

Amazon has been designed to support many large fault-tolerant deployments, rather than a mega-deployment which would be required for a global integration platform.

## 4.5 Focus Group 3 (FG3)

Broadly speaking, the focus group felt the assessment of existing integration platforms was correct.

The group believed CORBA was no longer relevant; it was just too heavy to be considered a viable integration platform in today's rapidly changing cloud environment. One comment made was that it had suffered the same fate as desktop applications when faced with opposition from web interfaces with proprietary standards being the overriding limitation.

Additionally with the conversion from LAN-based to WAN-based integration, any platform that did not support significant encryption and security features was no longer tenable. Man-in-the-middle attacks were raised as a concern and it was emphasized that any global integration platform must be resilient to attack.

SOAP interfaces had looked promising given they were web-based and the standards supported many features including security layers. However, the group agreed that additional layers were what ultimately led to SOAP's demise and that most developers using SOAP did not implement the extra layers of functionality as they made the implementation too complex.

The group also felt that SOAP's verbose message envelope structure was acceptable for remote procedure calls (RPC), but provided poor performance and data usage when moving large sets of data. The increasing adoption of client-side caching was also leading to service calls that transferred large sets of data in one go. Again, this does not fit well with SOAP given its verbose message structure.

According to the group, when faced with competition from REST, SOAP looked tired, heavy and complex. REST, which increases flexibility without standards enforcement, has rapidly become the integration standard of choice despite having rudimentary support for security, authentication and complex RPC support.

The REST vs SOAP discussion also raised the subject of models and meta-models versus the motivation to use them. Increasingly, formal models were being seen as an overhead that did not provide a good return on investment. It was felt the maintenance of model definition files such as schemas represented an overhead many developers were unwilling to support, and that the industry trend was towards less weighty implementations.

The group raised the possibility of binary transport protocols becoming more common in the service space, e.g. BSON, which is a binary-encoded version of JSON. This is a key feature addressed by the reactor platform design and will be discussed later in this thesis.

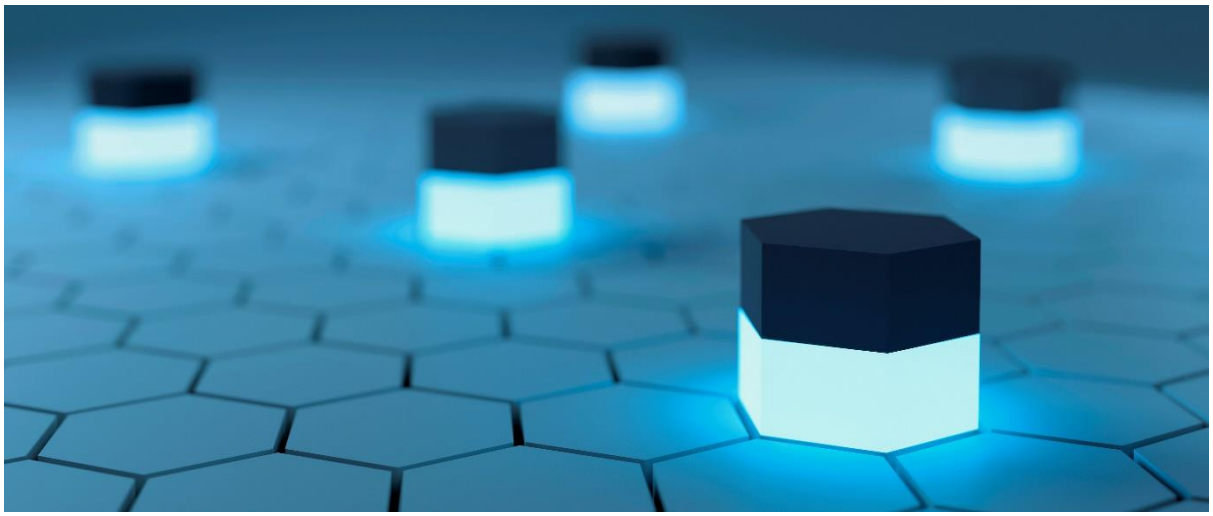
## 4.6 Conclusion

In this section, we have detailed our research findings as well as statistical analysis. We have explored results from both the Survey conducted and the Focus Groups conducted throughout an 8-month period. In some cases, data has been summarized and additional detail can be found in the appendices.

In the next section, we explore a single hypothetical design for a network that may address some of the barriers identified in the results section.

# 5

# Reactor Platform Design



Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In this section, in an attempt to address the findings from the research barrier investigation conducted in previous chapters, we introduce our design for a global integration platform: 'The Reactor', the core component of a global communications ecosystem. The reactor was named to reflect the reaction it is required to have for inbound events. Design of the reactor platform takes into account all input from previous stages in the methodology as outlined in chapter three. The main objective of the reactor is to stimulate further discussion that may lead to new and improved designs, to the point where it can be made ready for production.

## 5.1 Overview

As identified in the platform review section, there are a number of enterprise-based solutions which attempt to address elements of solution integration. Both the VDM and CORBA solutions address centralized and reusable mappings to make transforming between heterogeneous models easier, improving Hasselbring's heterogeneity problem dimension. Additionally, a web services implementation provides an easy way to proxy remote-procedure call facilities, which improves Hasselbring's distribution problem dimension (Hasselbring, 2000).

The problem with these approaches is that they still require the solution to be relatively tightly coupled with a component, via method stubs and/or WSDL consumption. This means the solution is prone to failure should method signature change occur, although model changes are, in some cases, safeguarded by the transformation mappings support.

With the advent of the cloud, however, we no longer have control over all system components. In the case of CORBA, a cloud-based system component could not be forced to implement the method stubs that CORBA provides to transfer data. In the case of VDM, this is mitigated by controlling all transformations locally (client-side) and therefore VDM appears to be more resilient to change.

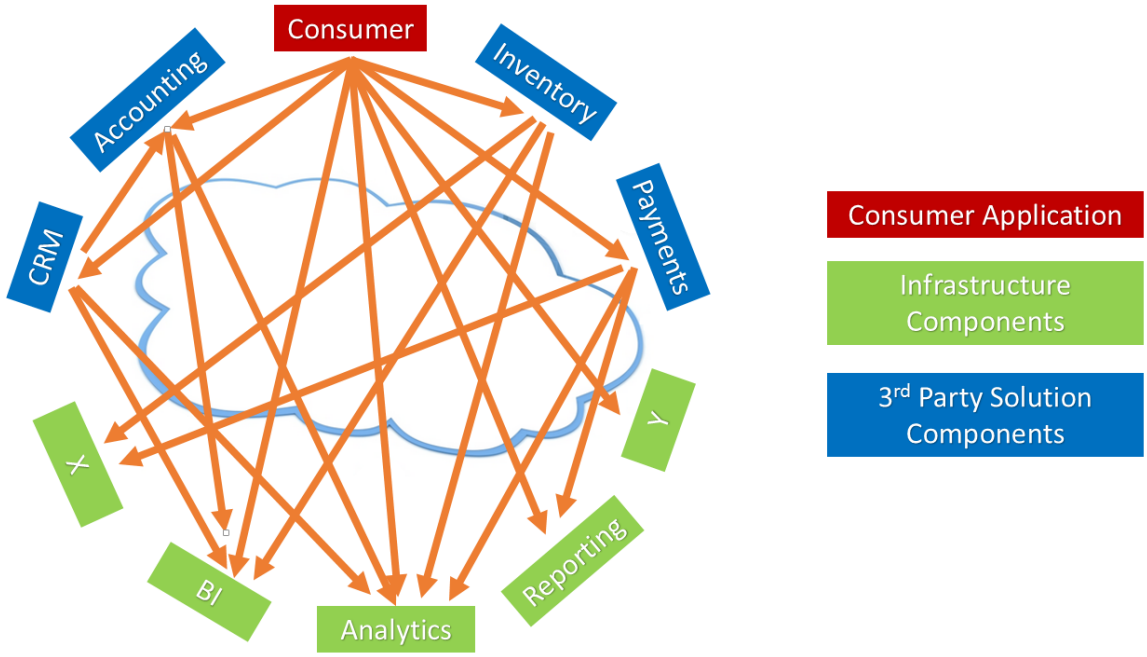



*Figure 22: Cloud-Centric Integration*

As previously identified, traditional integration with middleware occurs in two ways:

- internal - the middleware will broker communication between internal system components
- external - the middleware will integrate with middleware from a third-party system.

VDM and CORBA are effective hub-and-spoke solutions when considered only in the context of the local area network, or organizational domain boundaries. However, it can be seen by Figure 22 that when external integration is conducted in conjunction with the cloud, it creates a giant peer-to-peer integration network where millions of solution nodes are free to integrate with each other, which further enforces 'n!' integration solutions. Just as this proved complex on a local area network with small numbers of components, complexity reaches new heights when the number of cloud-based solutions is considered. Continuing down this path will likely make any chance of achieving a high-speed and responsive digital nervous system impossible.

## 5.2 Reactor Platform Design

### 5.2.1 Introduction

Designing a global integration platform is not a trivial matter and involves considerable future work. However, for the purpose of discussion, this thesis considers all data gathered from the survey and focus groups, as well as the literature review, and lays out one possible design for the reactor peer-to-peer integration network.

Hasselbring himself stated that addressing all integration problem dimensions may not be advantageous to all integration solution scenarios (Hasselbring, 2000). There are no silver bullets and therefore direct integration for remote-procedure call scenarios is probably still the best alternative. However, the design of the reactor aims to address all three problem dimensions and is particularly useful for integration solutions that focus on disseminating data (as opposed to remote-procedure calls).

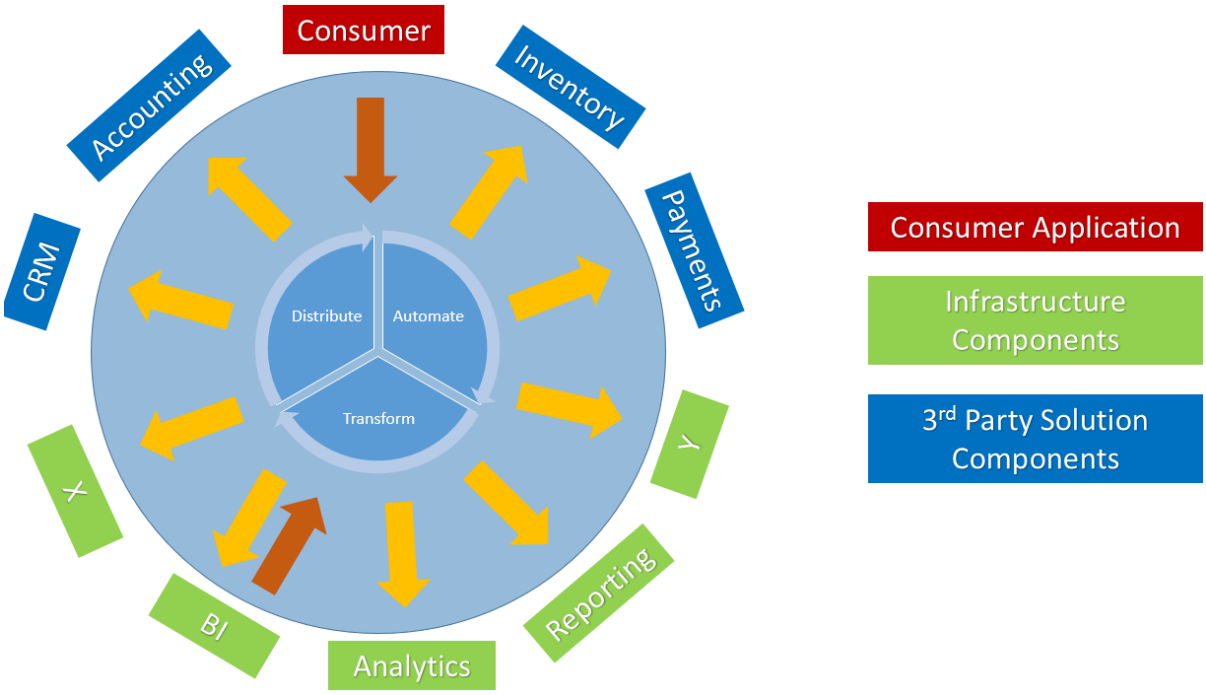



*Figure 23: Reactor - Global Integration Challenge*

We start by applying integration platform logic that exists in a cloud-based setting and extend it by introducing the distribution, autonomy and heterogeneity components as seen in Figure 23. As previously mentioned, it is unreasonable to expect that a single instance (even if it were a large cluster) would service the entire world for a number of reasons, including scalability, redundancy and geographic access.

This initial step does, however, allow us to group the required integration functionality at a logical level before determining how best to distribute load.

### 5.2.2 Node Structure

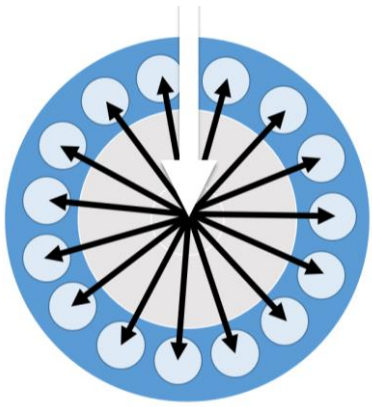


*Figure 24: Clustered Node*

At a physical level, if we are to construct a network that is infinitely scalable, then we must design a network which scales in many ways. We begin by establishing a single reactor instance as a node, which may be made up of one or more servers in the form of a cluster as illustrated in Figure 24.

Requests enter the node via a single endpoint which is load balanced across a cluster of reactor servers, and which employs stateless architecture. Additional servers within a node configuration would extend the maximum processing throughput of a node.

Additionally scalability can be achieved by classifying nodes accordingly and allocating them specific roles within the global network. At a logical level, nodes provide integration functionality via multiple layers as an extension to the OSI model.

### 5.2.3 Node Classification

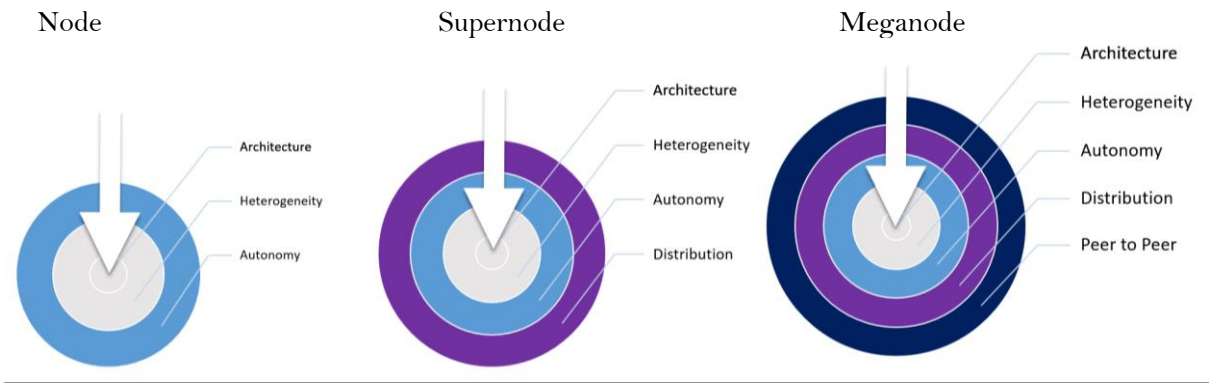



*Figure 25: Node Classification*

Node classification exists so responsibilities can be divided efficiently between layers of nodes. The reactor design supports three types of nodes, each inheriting and extending node functionality as illustrated in Figure 25.

- The architecture layer is the core of all nodes and provides support for infrastructure, architecture, administration, maintenance and deployment.
- The heterogeneity layer is responsible for supporting all internal data and the relationships that exist between them including data models and transformation mappings.
- The autonomy layer is responsible for executing business intelligence rules, process execution and conditional logic. Any feature related to the control path of a data flow is related to the autonomy layer.
- The distribution layer manages the inbound and outbound message flow for the node. It includes managing publisher and subscriber routing, and the internal route table. It does not exist within the basic node because that communicates directly with a supernode and therefore relays events without distribution logic.
- The peer-to-peer layer applies extended distribution functionality used between meganodes to balance load at the mega level.

Figure 26 shows how this architecture would work relative to the open systems interconnection model (OSI). It borrows from some of the principles of the Juxtapose peer-to-peer network structure by using the JXTA core to manage peer-to-peer distribution (Airamo, 2015).

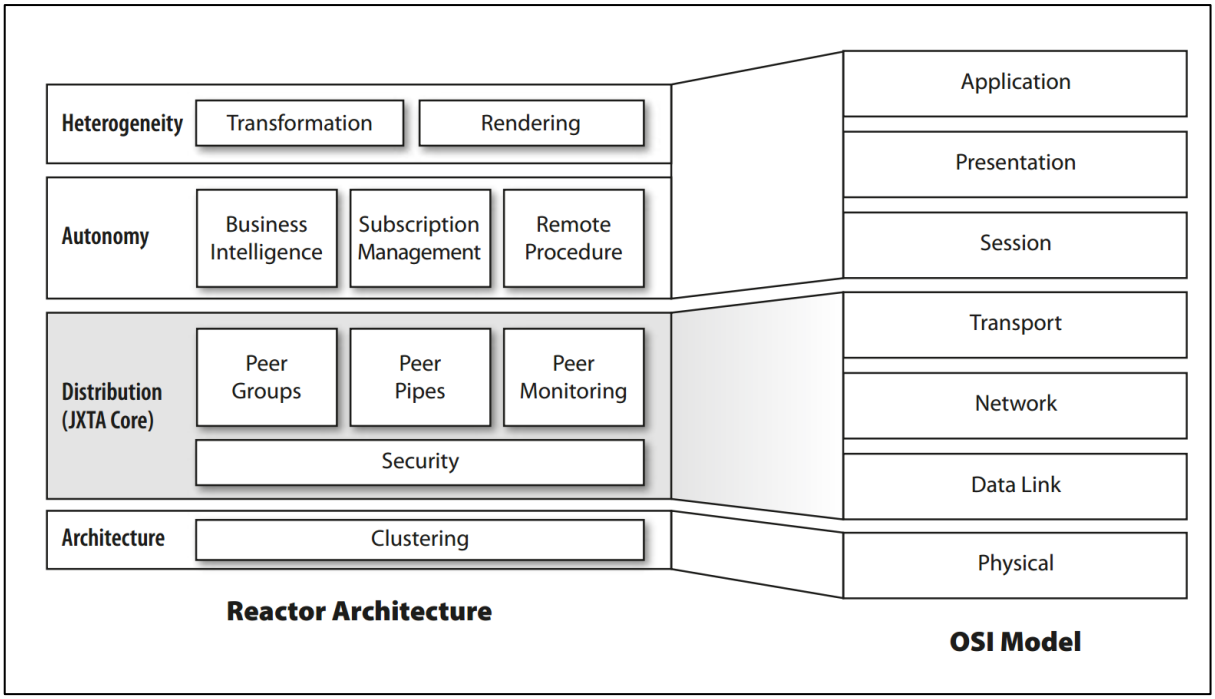



*Figure 26: Reactor Network Model*

### 5.2.4 Superhighway

In the case of conventional routing, the reactor service functions in a similar way to a peer-to-peer node. In peer-to-peer a node may publish a torrent and either download or upload a file associated with a torrent. With the reactor network, an account may publish routes and either post or consume a model payload.

If we continue this comparison, and consider an application to be a reactor node (due to its ability to receive payloads either by being a service or by receiving a response), then we must promote the reactor node itself to the status of a supernode which manages and controls routes.

A meganode is a supernode that has the capacity to interoperate with other supernodes. The act of establishing the meganode/supernode structure is ultimately what enables us to construct the information superhighway. A visualization is shown in Figure 27 below.

If a meganode in Europe receives a payload which has two million subscribers, one of which is the meganode in Asia which has 10 million subscribers, we can avoid the transfer cost of 10 million requests from Asia and service those locally in a similar manner to the way content servers distribute static content today. The main difference is that we have the capacity to apply different rules, filters and transforms for any of the 12 million total subscribers empowering multiple different views of the same information.

The reactor platform is, at its core, a message broker implemented as a centralized cloud service with which third-party cloud solutions connect to integrate with other third-party solutions without the need to formally 'bind' to those solutions.

A hub-and-spoke integration style allows for governance of transformation, distribution and autonomy, relieving the load from the consuming application.

For the reactor service to be a viable integration solution, it must support the key functionality we have identified from an enterprise service bus.

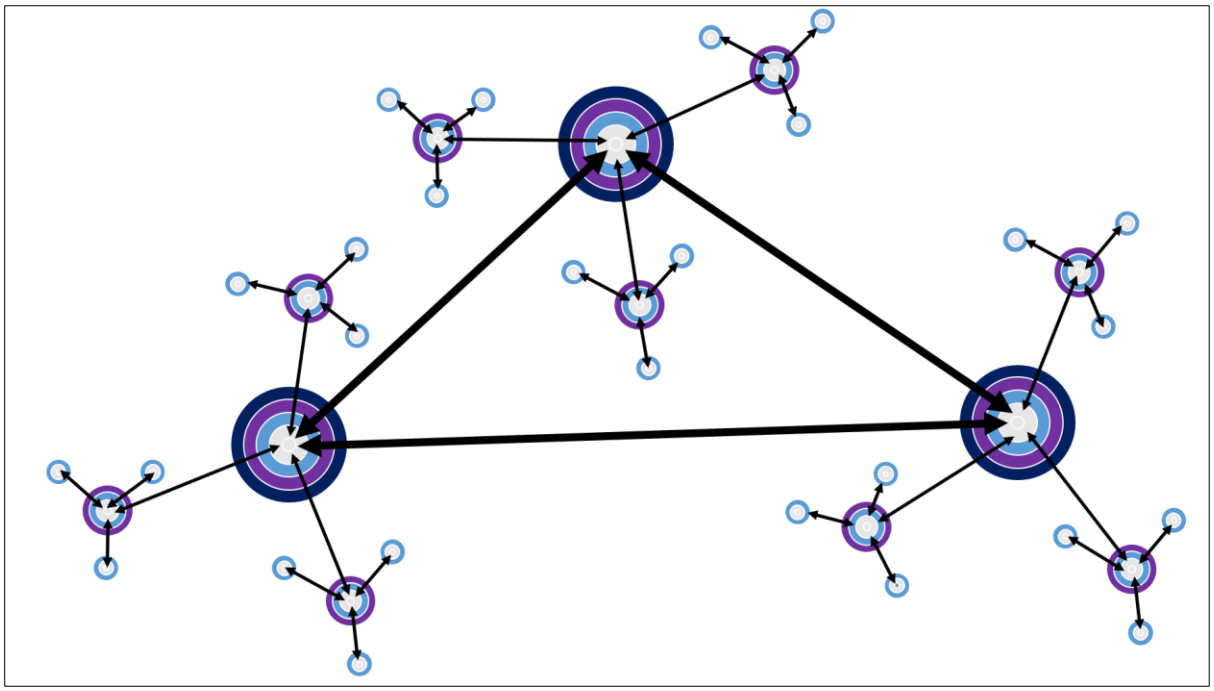


*Figure 27: Reactor-Based Information Superhighway*

## 5.3 Reactor Autonomy Layer

The autonomy layer within the reactor platform allows for an expression language that can support complex event processing, execute business intelligence and, if necessary, apply conditional routing. As the configuration is data driven and easily modified, the platform remains resilient to organizational change.

With advanced needs in mind, the reactor platform has been designed to support container execution. This features execution of compiled code in a safe way as the result of a platform event.

The three-layer network architecture ensures autonomy is balanced throughout the network in the same way that load is distributed. This forms a balance between decentralizing and centralizing autonomy, achieving the best of both worlds.

### 5.3.1 Invocation

The reactor service invokes business functionality in an abstract fashion by sculpting requests to and receiving responses from third-party services. These services form the components of the solution that is fabricated by the sum of functionality provided by multiple applications.

### 5.3.2 Process Orchestration

As a reactor service request is atomic in nature, process orchestration is limited to occurring in one of two ways:

- by establishing a network of routes that execute process based on conditional filtering
- by implementing atomic process within a container.

### 5.3.3 Complex Event Processing

The reactor's support for asynchronous routed messaging lends itself to full support of complex event processing.

Additionally the reactor service abides by a number of key architectural principles.

- *Loose coupling* – where possible the reactor service will reduce loose coupling, e.g. removing the need to consume a WSDL, or forcing a specific model structure. Looser binding between server and client leads to simpler, later development. Changing message structure can occur without changing client code — an advantage for mature systems which are often not being constantly developed.
- *Atomicity* – all messages received into the reactor are considered 'atomic' or self-contained. However, if a message is routed into the business intelligence component, it can result in a chained series of events — although these will remain stateless and unconnected.

## 5.4 Reactor Heterogeneity Layer

### 5.4.1 Messaging

Utilizing the REST protocol, the reactor service receives a message in XML and JSON format via a GET, PUT, POST or DELETE operation and processes the message against a routing table managed by the solution.

Using a REST-based approach has the following advantages.

- *Scalability*: removal of maintained state allows for increased scalability as subsequent requests can hit a different load balanced node of the reactor service array.
- *Resilience*: also connected to absence of state and the atomicity of transactions, a client can lose and regain a connection, and continue to send messages without the need to re-authenticate.
- *Simplicity*: inbound and outbound messages conform to REST protocol and unlike RPC-like approaches, contain a small number of verbs accompanied by a payload — making the reactor service considerably easier to administer.
- *Lightweight*: often RPC-style models are encapsulated in wrapper structure which can add weight to the message during transit. With REST this is not needed and in some cases where the message is simply an event, the payload may only be one or two lines of text. Additionally, encoding of media files is more efficient because it is not necessary to encode within a string before transit.
- *Caching*: because the payload message contains no state, the result set may be cached for use at a later time, which is especially useful if content is being dispersed to a wider audience.

#### 5.4.1.1 Base Message Format

- Header – contains information about the packet, including origin, destination, length and packet number, and an optional sub-header option for each of the reactor layers:
    - architecture – may contain security tokens, master node ID, etc
    - autonomy – may contain input parameters for use within business intelligence
    - heterogeneity – may contain which model/transform to use
    - distribution – may contain which destination node to send to
- Payload (or body) – contains the data that comprises the packet.
- Trailer – indicates the end of the packet and frequently includes error detection and correction information.

### 5.4.2 Mediation

Mediation refers to the ability to transform or translate messages from one format to another. This is a critical part of the integration process as it ultimately unifies disparate business functionality from multiple systems and allows it to be presented in a common structure.

*Direct*

A standard direct transformation enables conversion from a source format to a destination format. Direct transforms from custom models should not be allowed and the reactor should enforce intermediary transforms as per below.

*Intermediary*

An intermediary transform is an important part of promoting a common model ecosystem. Rather than transforming a payload from A to B, it enforces the notion that the transform must apply an A to I (platform and representational neutral intermediary) and then a second I to B transformation.

This has a number of subtle benefits:

- It encourages the provider of model A to transform to standardized and best-practice models. This addresses Hasselbring's heterogeneity problem dimension by promoting a common object model (Hasselbring, 2000).
- It persuades the service provider of model B to maintain the I to B transformation as it ensures client connectivity is in good health when service changes are made. Over time it removes some emphasis on the need for versioning as the service provider can release the transform mapping at the same time it releases the service change.

Over time it encourages the provider of model A to adopt model I as its own model. In doing so, it will be possible to route directly from I to B, a transformation that can be maintained by the service provider of model B.

As models and transformations are data driven within the platform, the platform itself remains model, solution and endpoint agnostic, which ensures it will not break given a change in any model/transformation.

The use of best-practice standards and wrappers encourages wider adoption of the platform by the developer community.

## 5.5 Reactor Distribution Layer

### 5.5.1 Routing

A key component of the reactor service is routing functionality during delivery of messages. Delivery may be determined by different forms of routing logic including content inspection, load balancing, broadcasting, business process or configuration. The act of routing affords the ESB the opportunity to delay delivery and split or combine messages.

The routing table will be configured to deliver payloads of certain types to service endpoints, transforming them in transit if necessary. Additionally, the routing table enables splitting and re-routing of payloads in an autonomous manner.

Hasselbring notes that autonomy is one of the hardest of the problem dimensions to address technically. However, the routing functionality of the reactor service does address communication and provides us with some scope to be able to repoint payloads to another service provider, and control what information reaches what service without changing the internal application. This reduces the

need for engineering when integration solutions change, e.g. switching accounting or payment gateway services.

Routing supports the following options:

#### 5.5.1.1 Direct Routing

Direct routing routes an inbound payload to an outbound service endpoint using REST. In the example below the payload is transformed from inbound format to outbound format.

| Inbound | Outbound | Transform | Service endpoint | Response callback |
|---|---|---|---|---|
| Sale_Acme.XML | Sale_Zero.XML | Sale_Acme_Zero.TR | Endpoint_Zero | Callback_Acme |

*Table 20: Direct Routing Example*

#### 5.5.1.2 Split Routing

Split routing allows for a payload to be routed to multiple destinations. Each route is treated as an atomic transaction and, therefore, there is no rollback should one route succeed and another fail.

| Inbound | Outbound | Transform | Service endpoint | Response callback |
|---|---|---|---|---|
| Sale_Acme.XML | Sale_Zero.XML | Sale_Acme_ Zero.TR | Endpoint_Zero | Callback_Acme |
| Sale_Acme.XML | Sale_Acme.XML | None | Endpoint_Analytics | None |

*Table 21: Split Routing Example*

#### 5.5.1.3 Conditional Routing

An optional condition plan allows for analysis of the payload to determine if the route should be executed. An example may be that you only want to SMS someone if the sale has a value greater than $5000.

| Inbound | Outbound | Transform | Service endpoint | Response callback | Condition |
|---|---|---|---|---|---|
| Sale_Acme.XML | Sale_Zero.XML | Sale_Acme_Zero.TR | Endpoint_Zero | Callback_Acme | Cond_Acme_Sale.CON |

*Table 22: Conditional Routing Example*

#### 5.5.1.4 Route Subscription

With public accounts, third parties may opt in with their own custom route. Functioning in a similar manner to a subscription to an RSS feed, it would automatically route the payload to the third-party endpoint. This would be useful in news syndication scenarios, and combined with a condition, would allow filtering of certain content.

| Account | Route owner | Inbound | Outbound | Transform | Service endpoint |
|---|---|---|---|---|---|
| Acme | Acme | Sale_Acme.XML | Sale_Zero.XML | Sale_Acme_Zero.TR | Endpoint_Zero |
| Acme | Marvin | Sale_Acme.XML | Sale_Zero.XML | Sale_Acme_Zero.TR | Endpoint_Marvin |

*Table 23: Route Subscription Example*

As endpoints are data driven with the reactor platform, the platform itself remains endpoint agnostic. Any relationship to services including service discovery is unnecessary when using a REST interface. Interfaces can still be reused by sharing models, transformations and route configuration through the reactor platform.

## 5.6 Reactor Architecture Layer

### 5.6.1 Adapters

The reactor service supports adapters on a number of levels.

- Connection adapters allow for different formats to be supported. For example, in modern times, open authentication has become popular, which requires a security key to be transferred as part of the payload. Alternatively an open REST-based interface without open auth would not allow for interchange of data in a secure fashion.
- Transformation adapters allow multiple forms of mapping to be supported; e.g. from XML to XML, XML to JSON, JSON to JSON, CSV to XML, etc. Transform adapters can also be employed for proprietary formats.
- Container adapters allow for business intelligence to be executed within the reactor to transform payloads according to some advanced logic. Executing the intelligence in a container enables referencing of external data by web request or complex calculations that may be beyond a standard transform.

### 5.6.2 Security

The platform's security is controlled within the architecture layer and includes account management (e.g. user management, publisher and subscriber access control).

All payloads are encrypted and salted according to both static and random content to ensure they are secure. Account-related settings would also be secure in the event of an attempted hacking.

The reactor service provides security support in a number of ways.

- Due to the nature of the transformation process, a standard transformation must be unencrypted at the time of transformation. Communication to and from the reactor service may be controlled using SSL.
- Message payloads can have values obfuscated if both the sending and receiving applications support it.
- Message payloads can be fully encrypted if both the sending and receiving applications support it, and if the transformation is conducted within a container instance.
- The reactor service should be deployed within a PCI-compliant network.

### 5.6.3 Management

The reactor service supports a single web interface which allows for centralized management of monitoring, logging and configuration, and administration of key information such as:

- *Account* (System Owner) – the account is a core organizational entity that identifies which transformations and routes to use, based on the AccountIdentifier that is included with the request message.
- *Service* – an account will register (0 – many) services that it either provides or consumes for the purpose of integration. An example may be a third-party accounting provider or payment

gateway. The service entry will also contain any keys required for posting to the service. Service endpoints are similar to the application support that the VDM approach provides. However, they are loosely coupled.

- *Operation* – an operation reflects a request URI belonging to a solution, which will be referenced by the route along with a model (payload).
- *Models* – a model represents the structure of a DTO, business entity, logical entity or other payload structure. If the model is marked as XML it is an XML schema. For JSON models it will be a JSON schema. Both the VDM and CORBA solutions proposed centralized management of the models so they could be modified, accessed, shared and executed by stakeholders and interested parties. Unlike the VDM approach, this is fully managed and executed on the server, thus maximizing the reuse potential, minimizing the necessary coupling and leveraging conditional and routing mechanisms.
- *Transformation* – a transformation defines how a source model is transformed into a destination model. In the case of an XML -> XML mapping, this would be an XSLT transform file.

### 5.6.4 Usability & Flexibility

Because the reactor platform is, in essence, a very large API it must be constructed in a way that is familiar to users and be easy to use.

Given that our survey results indicated the majority of integration developers utilized or were familiar with the W3C standards, the adherence to W3C's XML, XSLT and REST standards ensures a sufficient level of familiarity.

The platform also aims to provide familiarity via other recognizable technologies such as peer-to-peer mechanics for distribution and DNS configuration for routing.

Most common operations can be performed quickly and, in some cases, templated. Additionally, new users can easily adapt to the software without help and extend working rule sets so platform usage can grow with users' needs in a flexible, usable manner.

### 5.6.5 Maintainability & Testability

Again, because the reactor platform is an API, it is easy to use as all functionality is exposed via an API interface. This means those interfacing with it can also implement specific tools to automate maintenance — a huge advantage when integrating a third-party service.

Testing of the platform can be achieved by omitting a production token from the message payload and special test routes can be established to ensure flow is executed correctly.

### 5.6.6 Availability + Scalability

The combination of a clustered structure for a single node and the three-layered mesh network with peer-to-peer extensions at the meganode layer offers unparalleled scalability and availability.

Failure cases are covered as follows.

- If a server disappears, the node cluster will redistribute the load.
- If a node disappears, asynchronous messages to that node will be queued for a set amount of time before failures are registered.
- If a supernode disappears, asynchronous messages to that node will be queued for a longer period of time before failures are registered. Nodes which have that supernode as a master, will automatically register with a secondary supernode (similar to DNS failover).
- If a meganode disappears, supernodes will re-route to another meganode.

- During high-loading periods, a node cluster will scale out by adding more server capacity.

### 5.6.7 Availability + Reliability

While availability has not been addressed in detail, the three-layered network aims to increase availability and reliability. This relates to how long the system is up and running, and the mean time between failures (MTBF) is known as the availability of a program.

Additional considerations that need to be researched and discussed are as follows.

- How long does the system need to run without failure?
- What is the acceptable length of time for the system to be down?
- Can down times be scheduled?

Typically, a platform of this nature should be increasingly fault tolerant with as close-to-zero down-time as is realistically possible and thus needs a network architecture foundation with hot fail over capability.

### 5.6.8 Extensibility

Extensibility is provided by the platform in a number of ways.

- Models and transform sets can be extended as needed and are data driven so can change as required.
- Additional third-party platform endpoints can be added as needed, enabling messages to be sent to them and received from them.
- End users and third-party developers can extend the system through flow logic and business intelligence via containers.

### 5.6.9 Portability

The platform is portable as it incorporates W3C standards, which are cross platform, and any native language support is executed as a container. Data itself is transported using cross-platform XML via REST interfaces, which are both language and platform neutral.

## 5.7 Reactor Extensions

The primary goal of the reactor platform is to lay out an interface that is simple and extensible. It does this by allowing scripting and dynamic configuration of data, and by linking third-party service providers as participants in the node network.

During the course of this research, we have reviewed the Apache NiFi integration platform and found remarkable similarities between its structure and the internal layers of the reactor node platform. Consequently, given that NiFi has excellent localized autonomy and heterogeneity support, it is possible to envisage its use in place of the node structure as shown in Figure 28, leaving the reactor platform to manage distribution and peer-to-peer mechanics alone, which would extend NiFi to a global audience.

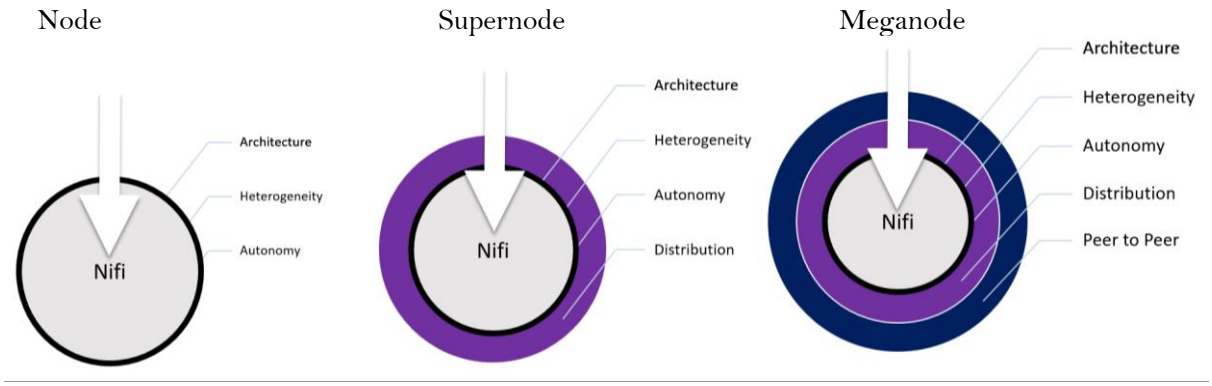



*Figure 28: Reactor Node Design*

| Dimension | Requirement | Rating | Dimension | Requirement | Rating |
|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | Yes | *Distribution* | Scalable distribution | Yes |
| | Business intelligence | Yes | | Proxy services | Yes |
| | Intelligent nodes | Yes | | Endpoint agnostic | Yes |
| | Complex event processing | Yes | | Service discovery | Yes |
| | Service composition | Yes | | Service reuse | Yes |
| | Asset wrapping | Yes | *Architecture* | Reliability | Yes |
| | Virtualization | Yes | | Availability | Yes |
| | Model-driven implementation | Yes | | Security | Yes |
| *Heterogeneity* | Balanced autonomy | Yes | | Portability | Yes |
| | Model agnostic | Yes | | Functionality / extensibility | Yes |
| | Solution agnostic | Yes | | Interoperability | Yes |
| | Structures | Yes | | Usability | Yes |
| | Supported standards | Yes | | Maintainability | Yes |
| | Wrappers | Yes | | Efficiency | Yes |
| | Service | Yes | | Testability | Yes |
| | | Yes | | Reusability | Yes |
| | Messaging | Yes | | Ease of deployment | Yes |
| | Message control | Yes | | Ease of administration | Yes |
| | Message transformation | Yes | | Scalability | Yes |
| | Message security | Yes | | | |
| | Message monitoring | Yes | | | |

*Table 24: Reactor Assessment Table*

## 5.8 Reactor Use Cases

In its initial phase, the reactor has limited support for remote procedure calls, and focuses on ESB-related dataflow functionality. While business intelligence can certainly be added and extended over time, the initial use cases that best suit the reactor are those that disseminate common data to multiple platforms such as:

- News syndication – e.g. a newspaper may be able to publish news articles and have other newspapers worldwide subscribe to it (similar to an RSS feed). The advantage over RSS delivery is that the message would not require polling but instead would arrive as a message payload when available, much the same as an email arriving in an inbox. This is advantageous when attempting to synchronize with hundreds or thousands of content providers.
- Auctioning sites – the motivation to adopt a common model means service providers can share data and action it in real time. With the reactor platform, it would be possible for all auctioning

sites worldwide to notify all other auctioning sites of a listing including its comments, bids and shipping prices to the rest of the world. If a consumer was looking for a special car, they might be able to buy one from another country via their local auctioning site without any additional difficulty.

- Social media sites that currently mine online services such as IMDB will again be able to do so asynchronously. When a new movie is released, the notification will be sent to the reactor, which will, in turn, be distributed and split as required, and delivered to all subscribers. If there are 10,000 interested parties polling every 60 seconds, this would reduce server load considerably and negate the need for huge amounts of unnecessary Internet bandwidth.
- Major players in the peer-to-peer space have begun aggregating sources again, allowing clients to stream directly from a single endpoint. Recently New Zealand consumers overloaded content provider Netflix during its launch, resulting in a network performance issue. The reactor network (and its peer-to-peer support) would encourage the distribution of this load and nearby nodes to pool resources.

## 5.9 Conclusion

In this chapter, we have taken an architectural approach to designing network and platform components, which could participate in a solution for enterprise integration barriers. The platform designed does not intend to be a complete solution, but rather promote discussion on the issues identified and whether certain suggestions can be considered adequate solutions for these challenges.

In the next chapter, we wrap up the final focus group with a review of the reactor platform, and we interview subject matter experts to give a final opinion of the enterprise integration space and whether they believe an overarching solution to integration barriers is achievable.

# 6

# Final Review & Interviews


Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In the previous chapter we laid out the design for the reactor platform, a potential solution to enterprise integration challenges.

This chapter outlines the final review of the focus group including opinions and perspectives on the reactor platform. It also includes interview feedback on the overall process including the focus group output, reactor platform, and additional insights and opinions of the integration landscape that may not have been captured already within the focus groups.

## 6.1 Focus Group 4 (FG4)

The focus group's response to the notion and vision of a global integration platform was overwhelmingly positive — in direct contrast to the initial focus group where the proposition was met with some scepticism.

The value of the focus group process has been to capture and work through concerns at strategic, business and technical levels to the point where a possible implementation is now feasible. That said, the group still registered many residual concerns, which must be considered in any future work.

The group could see the value in motivating developers to use the integration platform by offloading many of the ancillary tasks such as schema and transformation maintenance, and business intelligence construction. The idea that the platform could someday offer an array of off-the-shelf components that could be activated at will was a popular one.

The group felt, however, that for the platform to reach this level of mass adoption it would have to achieve what many open groups such as W3C were struggling to realize. For the technology to remain open-source and available to all, the platform needed support from the commercial world to reach its true potential. On this basis, the group felt a large commercial, open-source-friendly partner, e.g. Google, was needed to drive the initiative forward.

The group noted that value can often be seen in a platform or technology, but when it comes to implementation the level of customization required negates the benefits, so many organizations develop their own solutions. To prevent this, templates provided must require little, if any, customization to become production ready and it would be challenging to find the balance between bespoke requirements and flexibility. There was agreement, however, that modern languages with features such as reflection were making extension metadata easier to achieve, thereby reducing the barrier. Good support for metadata would allow the core ontology to be more abstract and appeal to a wider audience.

The platform would need to have clear separation of concern between its layers (heterogeneity, distribution and autonomy) and not try to achieve all of these in its first version. Any global infrastructure service should have a simple base structure that is capable of being extended by commercial organizations for competitive advantage.

With this in mind, one important step would be to firmly establish a peer-to-peer transport protocol as part of the OCI model. This would require a shift in mind-set by those who still associate the technology with pirating of illegal content and for them to start embracing it as a backbone for asynchronous communications. The group suggested some parallels between this and how telecommunications companies are establishing greater numbers of smaller cabinets in streets to help distribute network loading.

By tackling the distribution challenge in isolation, research may be conducted that addresses security, trust and other distribution concerns with peer-to-peer, as the focus groups raised major concerns with man-in-the-middle attacks.

The group sensed a growing need to improve on SSL as security for data transmissions. SSL was a good solution for website access initially, but in recent years it has been increasingly vulnerable and therefore a different approach is needed — possibly incorporating a binary protocol in addition to stronger levels of encryption and obfuscation.

It was felt the developer community at large was already monitoring what is unfolding in the dark web space and how developments there may flow into the commercial sector.

The group acknowledged the reactor design had been inspired by a number of existing technologies in addition to peer-to-peer in an attempt to create a best-of-all-worlds solution. Static content distribution had been a driver for enabling the download of content off a neighbour in preference to its source. Subscriber and publisher mechanics were similar to those seen by RSS feeds and would remove the need for unnecessary polling of systems. The group felt the combination of peer-to-peer and the subscriber/publisher model would significantly reduce unnecessary Internet traffic.

It was noted that any subscriber/publisher feature which involved registration of routing rules, should have an implemented expiry that must be renewed routinely to remain active. This would ensure there were no stagnant routes that could become problematic over time.

It was thought that established and mature business markets such as the financial sector represented a good opportunity in which to prototype an implementation; while constantly evolving and life-critical markets such as the medical sector should be avoided in the early stages.

Some residual concerns around autonomy remain, e.g. how to manage the centralized node registry and how to prevent network echoes or circular routes. Also, what to do about malicious denial of service attacks.

Finally while discussing the future of integration — which included talking about the remote injection of business intelligence into containers — some parallels were drawn between this approach and sending someone an iPhone application that runs on their phone and requires permission to work. This challenge remains the most sophisticated implementation of a global integration network, one that realizes true distributed intelligence, but requires many stages and a level of adoption to be achieved initially.

## 6.2 Interviews

Initially it was intended that the interviews would feed into the focus group and vice versa as essentially an audit process throughout the research. Unfortunately, it proved too difficult to engage participants throughout the exercise and instead they agreed to be interviewed at the conclusion of the focus group process.

The interviews provided a great deal of value as they helped to validate the findings of the focus group by providing an important sanity check before the results were collated and conclusions drawn. Before discussing the findings with the participants, however, it was useful to consider integration in general to get a feel for a participant's perspective on the landscape and future developments in the field.

Throughout the course of the interviews, participants were asked questions in an open fashion concerning the following areas:

1. Barriers to integration — what were the interviewees' perspectives on the barriers to integration? By asking this first, we attained a purer perspective of the state of play without tainting the feedback with the concept of the primary research question, or the subsequent research output.
2. Primary research question A — did the interviewees believe it was possible to achieve a global communications ecosystem?
3. Review of findings — did the interviewees agree with the findings of the focus group and was there merit in the resulting design?

### 6.2.1 Barriers to Integration

To begin the interview, an interviewee was (re)introduced to Gates' vision of business @ the speed of thought, and asked why we had made minimal progress towards fulfilling that vision.

*"Enterprise environments are extremely complex and often require bespoke solutions"*
*(Interview Participant)*

The primary feedback from the participants was that complexity was a major contributor in the current integration space. They agreed that the complexity of domain knowledge and matching toolsets was significant and that it was difficult to train an individual to reach a level where they were capable of adapting to any integration challenge.

On top of this, participants echoed the sentiment of the focus group that technology was changing so fast that integration solutions also had to evolve quickly. They acknowledged that one peculiarity of integration is that rarely are systems retired, at least not as quickly as technology is evolving, which means the number of integration scenarios continues to grow.

*"Technology changes so fast…there is rarely time to construct long lasting solutions"*
*(Interview Participant)*

This rapidly expanding landscape further cemented the belief that, for integration experts, training is key, understanding the fundamentals is imperative and that soft skills are important so compromises can be reached on integration solutions with a client.

### 6.2.2 Primary Research Question A

Interviewees were next asked if they felt it was possible to achieve a global communications ecosystem, which would make integration easier. Initially most participants were sceptical that such a construct could exist. A single, one-platform-fits-all approach was considered the polar opposite of current industry trends.

> *"A one size fits all solution rarely works in the technology space.."*
> *(Interview Participant)*

Participants were encouraged to think beyond their initial reactions and to consider what would need to occur for this concept to become a reality.

They agreed that any such platform would need to be open-source for mass adoption. Proprietary networks had come and gone, and in some cases established a niche, but very few had established a monopoly on a solution space because of the resistance of the open-source community to such a scenario.

Participants also believed, however, that an open-source platform alone would have difficulty achieving mass adoption and that it was essential to gain vendor support for the open standards and possibly government support as well.

> *"In order for a solution like this to work, someone big needs to get in behind it. In fact it might take a government initiative to highlight its importance.."*
> *(Interview Participant)*

It was felt that such a network would be a complex beast, and careful separation of concerns and standards would be needed at multiple levels.

Participants were of the opinion that, in recent times, Amazon and Microsoft datacentre economics had led people to soften their stance a little on proximity and cost pressures, thereby relinquishing some of the control of a network where integration is conducted. Additionally, cloud computing had opened up redundancy and scalability enhancements, which may apply additional motivation.

Interviewees, however, believed a common ontology was not a reasonable prospect in their lifetime. It was felt that agreement could not be reached and that a better approach would be to concentrate on establishing a common platform and to have more flexible tooling to manage the ontologies — thus achieving many of the desired benefits.

There was concern that any such platform would have to support extensive data and associated concerns. A platform centred on data flow could become backlogged easily and the method of transporting large amounts of data would have to be carefully thought through.

> *"A platform such as this needs to be extremely fault tolerant as a core internet service. Failure to deliver a payload is not an option"*
> *(Interview Participant)*

Vendor clash was raised as a potential problem — when vendors adopt the open-source standard and then start manipulating it to achieve a competitive gain. This would result in any convergence being reversed again. Additionally the point was made that this type of open networking starts to overlap

with the territory of Aaron Swartz, who was a major contributor to a number of W3C standards, and who helped to promote an open Internet.

One participant agreed with the assertion that an open integration platform could be a significant plus for academia. If academics were to be involved in cataloguing the ontologies, transformations and business intelligence components, they could also help to guide these in a communal way towards fundamentally correct practices, e.g. a university that specialized in medicine could help to research existing standards and define the ontology for a surgical procedure.

> *"I like the idea that universities could be involved in guiding the establishment and maintenance of an open source style ontology. Proprietary solutions just don't seem to work"*
> *(Interview Participant)*

Trust was another major factor raised during interviews. An integration platform of this kind provided a significant opportunity for man-in-the-middle attacks. Not only would the open protocols need to support highly secure encryption algorithms, but also the server nodes themselves would need to be trusted not to bypass or manipulate such protection.

### 6.2.3 Review of Findings

Participants were then given an overview of the focus group discussions and introduced to the reactor platform design.

Overall interviewees commented that the design of the platform showed promise; they made the connection that the platform was inspired by a number of best-practice approaches including static-content distribution, RSS feeds, peer-to-peer, W3C standards and others.

They felt that peer-to-peer was an interesting choice as the transport protocol. It was agreed that, traditionally, data flow had mostly been handled in a synchronous fashion. However, in recent years asynchronous messaging had become popular to reduce server loads and provide an opportunity to balance periods of peak processing. It was agreed that asynchrony helped to make peer-to-peer acceptable for data integration.

It was also agreed that having a platform like the reactor network would help to provide motivation for companies to consolidate and share their ontologies, thus improving the rate of information exchange.

Additionally, the concept of OSI model integration was discussed at length. Interviewees agreed there would be some real advantages to embedding a peer-to-peer transport layer deeper into the OSI, potentially providing the ability to persist locally and letting the network 'sort it out'. However, they were not so convinced that it was possible to achieve this change easily at a network level.

It was further agreed that to achieve this utopian view of integration required a number of slow-moving standards, managed by separate governing bodies, to change mechanisms that had been in play for some time. The consensus was that it was unlikely to happen.

We did discuss, however, how many people might have talked about the initial concept of the Internet in such a way, and agreed that anything was possible if there were a will and a need, which did in this case exist.

## 6.3 Conclusion

This chapter outlined the final review of the focus group including opinions and perspectives on the reactor platform. Overall, the interviews provided new insight and acted as a sanity check for the focus group, which had evolved organically throughout the year. Feeling was mixed as to whether an overarching solution could be found for enterprise integration and that a major player in the market would have to lead the way for it to become a reality.

In the next chapter, Discussion, we look at the results attained throughout the action research process and discuss their implications as well as consider how the results may flow into future work.

# 7

# Discussion

Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

The previous chapter outlined the final review of the focus group including opinions and perspectives on the reactor platform, which wraps up the action research process and provides the platform for discussion. This chapter is an opportunity to collate literature review knowledge with the different perspectives provided throughout the research process, cross-reference it with statistics and discuss the key considerations that have emerged when considering our research questions.

## 7.1 Discussion

One important point that emerged from the research and discussion groups is that Hasselbring's integration problem dimensions remain just as relevant today as when he classified them. Only the change in operating context of the greater Internet demands the addition of a dimension that addresses the enhanced concerns regarding scale (Hasselbring, 2000).

Once added, it can be said that all integration barriers and concerns raised during the research process and by the focus groups fit neatly into one of the four dimensions.

When all is boiled down, it becomes apparent that the main problem we face with integration, both now and in the future, is that of heterogeneity — the ever-diversifying set of models, transformations definitions and standards that result from a world where technology is unbounded and developers are free to engineer solutions that appease their logical and creative sides.

In what could be mistaken for a microcosm of life itself, the Internet community is, on many levels, polarising into two camps of developer.

1. The first camp is for those who embrace definition, standards and controls, and who are willing to invest additional effort and resources to work with them.
2. The second camp is for those who, through commercial pressures, a desire for simplification or merely because of laziness, seek a relaxation of standards and implementation freedoms.

What became apparent when discussing this issue within the focus groups was that both camps perceive benefit in having a more standardized operating environment. Even the most creative architects can bow to function over form when they consider the interoperability of their work within the greater Internet community.

The problem is that the second camp is either unwilling or unable to expend the resources to achieve uniformity. It is an argument of strategic vision versus tactical realities where the pragmatism of having to deliver solutions in a commercial environment overrides the desire to do so in an aesthetically pleasing way.

This overarching dichotomy provides the underlying motivation for establishing a global integration platform, as it will address the following:

1. It provides the necessary operating motivation to establish a global ontology that will evolve and be adopted by the wider community.
2. The platform, coupled with a common ontology, removes the burden of many of the additional development tasks that exist today, including the authoring of models and the maintenance of transformation files.
3. It removes the need to establish one-to-one integration solutions with many parties, instead integrating only with the integration platform itself.

In discussions with integration experts, it became apparent that when planning and implementing an integration solution, alignment between all levels of the business was not an issue. The focus groups agreed the major challenge of integration, once budgetary approval was obtained, was to achieve alignment between the business levels of the two integrating companies.

If the strategy level is considered the brain of an organization, the business level is its heart. The business level, in fact, defines an organization's ontology, model structure and relationships. Although debated at length at strategic levels, and stored and retrieved at a technical level, business defines the ontology.

Just as there is no common business ontology, neither can there be a common data model. This manifests itself in the fact that data models that are essentially the same have different names and different structures, even if they refer to a common entity. Confusion over these discrepancies is time consuming to those involved at all levels.

For this reason, and in consideration of research question D, any feasible integration platform must appeal to business and technology levels alike to motivate evolution of the common ontology and to form inter-level agreement in terms of the resulting ontology.

After extensive research of the existing integration platforms available, it became clear very quickly that the only platform which addresses the needs of both the business and technology levels is Apache NiFi. Its advanced user interface for administering data flow and business intelligence will appeal to both levels, and the consensus was that, in the future, administration would likely be adopted by professionals at the business level.

When considering autonomy there are two key areas. The first is autonomy and control of business intelligence in response to a flow of data. NiFi handles this level superbly and the reactor design in many ways mimics NiFi's approach. The second is in the higher-level autonomy, involved when scaling up a network of integration nodes. As previously mentioned, this is where we hit NiFi's limitation. The

reactor design proposes a smart, three-level network that incorporates mesh mechanics for reliability and peer-to-peer mechanics for performance and distribution.

| Requirement | CORBA | VDM | W3C | ISB | Apache NiFi | Amazon | Reactor |
|---|---|---|---|---|---|---|---|
| Resilient to organizational change | No | Yes | Yes | | Yes | Yes | Yes |
| Business intelligence | Partial | No | No | | Yes | Yes | Yes |
| Intelligent nodes | No | No | No | | No | No | Yes |
| Complex event processing | Yes | Yes | Yes | | Yes | Yes | Yes |
| Service composition | Yes | Yes | Yes | | Yes | Yes | Yes |
| Asset wrapping | Yes | Yes | Yes | | Yes | Yes | Yes |
| Virtualization | Yes | Yes | Yes | | Yes | Yes | Yes |
| Model-driven implementation | Yes | Yes | Yes | | Yes | Yes | Yes |
| Balanced autonomy | No | No | Yes | | No | No | Yes |

*Table 25: Autonomy Dimension Platform Comparison Summary*

NiFi addresses many of the concerns and requirements that have arisen as the result of this research, particularly in the heterogeneity dimension, which was deemed most problematic. NiFi, however, possesses one Achilles heel — it does not have a solution for managing autonomy between nodes and therefore cannot be scaled indefinitely, and certainly in its current form could not be considered as a global integration platform. In fairness to NiFi, it was not authored with this in mind. However, adopting this vision for the product would potentially help to resolve some of NiFi's architectural challenges in this area including its autonomy limitations.

| Requirement | CORBA | VDM | W3C | ISB | Apache NiFi | Amazon | Reactor |
|---|---|---|---|---|---|---|---|
| Model agnostic | No | Yes | No | | Yes | Yes | Yes |
| Solution agnostic | No | Yes | No | | Yes | Yes | Yes |
| Structures | Yes | Yes | Yes | | Yes | Yes | Yes |
| Supported standards | Yes | Yes | Yes | | Yes | Yes | Yes |
| Wrappers | Yes | Yes | Yes | | Yes | Yes | Yes |
| Service | Yes | Yes | Yes | | Yes | Yes | Yes |
| Messaging | Yes | Yes | Yes | | Yes | Yes | Yes |
| Message control | Yes | Yes | Yes | | Yes | Yes | Yes |
| Message transformation | Yes | Yes | Yes | | Yes | Yes | Yes |
| Message security | Yes | Yes | Yes | | Yes | Yes | Yes |
| Message monitoring | Yes | Yes | Yes | | Yes | Yes | Yes |

*Table 26: Heterogeneity Dimension Platform Comparison Summary*

Both NiFi and Amazon performed well in the heterogeneity dimension with NiFi proving to be superior because of its advanced administration tooling and provenance features, which will be critical components of a global integration platform. The solution to a global integration platform does not have to be the responsibility of NiFi alone however.

As discussed within the reactor design, there is the potential for NiFi to interoperate with a carrier mesh network that manages the missing autonomy, as well as scale and distribution, and allows NiFi to do what it does best in handling the heterogeneity aspect of integration. As can be seen in Table 26, NiFi addresses all heterogeneity concerns.

| Requirement | CORBA | VDM | W3C | ISB | Apache NiFi | Amazon | Reactor |
|---|---|---|---|---|---|---|---|
| Scalable distribution | No | No | No | | No | No | Yes |
| Proxy services | Yes | No | Yes | | Yes | No | Yes |
| Endpoint agnostic | No | No | No | | Yes | No | Yes |
| Service discovery | Yes | No | Yes | | Yes | No | Yes |
| Service reuse | Yes | No | Yes | | Yes | Yes | Yes |

*Table 27: Distribution Dimension Platform Comparison Summary*

In a similar way to autonomy, NiFi performs admirably with distribution except at scale. There is no centralized repository for storing publishers or subscribers of data and therefore payloads cannot be

distributed efficiently when more than one node is involved. The solution to this challenge occurs hand in hand with the autonomy challenge. An underlying network that supported global routing of payloads would relieve the NiFi platform of this responsibility.

| Requirement | CORBA | VDM | W3C | ISB | Apache NiFi | Amazon | Reactor |
|---|---|---|---|---|---|---|---|
| Reliability | Yes | Yes | Yes | | Yes | Yes | Yes |
| Availability | No | No | Yes | | Yes | Yes | Yes |
| Security | No | No | Yes | | No | No | Yes |
| Portability | No | No | Yes | | Yes | Yes | Yes |
| Functionality / Extensibility | Yes | Yes | Yes | | Yes | Yes | Yes |
| Interoperability | Partial | Partial | Yes | | Yes | Yes | Yes |
| Usability | No | No | Yes | | Yes | Yes | Yes |
| Maintainability | No | Yes | Yes | | Yes | Yes | Yes |
| Efficiency | No | Yes | Yes | | Yes | Yes | Yes |
| Testability | Yes | Partial | Yes | | Yes | Yes | Yes |
| Reusability | Yes | Yes | Yes | | Yes | Yes | Yes |
| Ease of deployment | No | No | Yes | | Yes | Yes | Yes |
| Ease of administration | No | No | Yes | | Yes | Yes | Yes |
| Scalability | No | No | Yes | | No | No | Yes |

*Table 28: Architecture Dimension Platform Comparison Summary*

Both the NiFi and Amazon platforms performed well architecturally, but stumbled over the same two requirements. The first was scalability, which we have discussed at length.

The second concerned security. Neither Amazon nor NiFi supported a centralized, federated, security model, so in a mesh of integration nodes, security would have to be controlled node by node. This is acceptable in a small LAN environment but not in the context of operating globally.

## 7.2 WSDL / UDDI

One important point to discuss are the attempts that have been made to address the issue of enterprise integration barriers prior to now. As mentioned briefly, at a technical level service oriented architecture has attempted to employ Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI) technologies to allow services to be "registered" and consequently "discovered" on a network. This raises the important question therefore of why the services are not, for the most part, being used in a seamless manner.

WSDL is an XML-based standard which helps describe an underlying service and UDDI promotes the registration of services in something like a service directory which can be queried to locate a specific service similar to how we would locate a phone number in a phone book. The reasons why these technologies have not been as successful as hoped, are closely tied to Hasselbring's problem dimensions.

Firstly, in a distributed environment such as the internet, there is no governing central UDDI Server. In order to discover a service, we tend to need to point a development environment and/or production software toward a known endpoint. Exhibiting this level of manual control over configuring the binding of the service negates the value of the UDDI server somewhat.

Secondly, heterogeneity is only partially resolved with the WSDL specification. While a service can define itself well, and is easily consumed, if change occurs, client software typically requires a rebuild to adjust to the changes in interface offering. Having a client which can magically adapt to changes in a WSDL interface is not something that can be easily achieved.

Finally assuming that the above two concerns had been adequately addressed, the overarching consideration is that of autonomy. The purpose of 'WHY' a system would connect to a service and 'WHAT' will be the resultant business logic executed when it does is something that must be designed

and implemented with care. In general, the cloud has not yet evolved to the point where solutions can scan for services and extend business logic accordingly without manual intervention. Such a solution would require greater advances in the field of automation and machine learning.

With these limitations in mind, the value that WSDL and UDDI services deliver are somewhat negated and in recent times, many development houses have abandoned its complexity in favour of light weight protocols such as REST in order to develop more productively.

## 7.3 Conclusion

In this discussion chapter, we review findings from the action research process, and discuss the architectural considerations for resolving enterprise integration barriers. In particular, it was discussed that overarching control of autonomy was an extremely difficult barrier to resolve. Underlying architectural challenges, however, is the need for a common ontology utilized by all businesses. Such an ontology may be idealistic and difficult to achieve.

In the following conclusion chapter, we consider all of the evidence attained during this research and combine it with proposed solutions and discussion in order to render some conclusions. We also discuss limitations of the research and areas for future work.

# 8

# Conclusion

Notes: Figure courtesy of iStockphoto LP. Permission granted under the iStock "standard" content license agreement.

In the discussion section, we have discussed key identified findings from this research and how they may affect enterprise integration in the future. In this section, we make our conclusions and discuss the valuable contributions this research has made to the field of integration. We acknowledge the study's limitations and discuss how these limitations may be addressed in the future, and finally we discuss what future work may follow from this research.

## 8.1 Conclusion

One important point that emerged from the research and discussion groups is that Hasselbring's integration problem dimensions remain just as relevant today as when he classified them (Hasselbring, 2000). Only the change in operating context of the greater Internet demands the addition of a dimension that addresses the enhanced concerns regarding scale.

Before concluding our main research question, we first will address our secondary questions.

### 8.1.1 What proportion of enterprise integration barriers are technical?

As previously mentioned, Lam identified four key integration barriers: strategy, organization, technology and policy. Key to the discussion and research has been determining the technical and non-technical barriers. A barrier that is non-technical in nature may also be exaggerated by the lack of seamless technical solutions (Lam, 2005).

Analysis of our survey results was inconclusive in determining which level of business is most responsible for integration barriers in the enterprise. Instead, it appears there is an additional factor in play — the alignment between the three levels, which may, in fact, be more problematic. Further

research is required to prove or disprove this assertion.

### 8.1.2 What are the current key technical barriers to information exchange? — research question B

Currently web standards that exist for information exchange offer maximum levels of flexibility and freedom to remain cross-platform and technology agnostic. With these freedoms comes a diverse range of solutions and, additionally, underlying technology stacks can promote further diversification by defining a model, contract and interfaces using specific technology-based patterns.

Because the exposed interfaces are diverse, they must be identified, analyzed and implemented to be consumed. Survey data, focus groups and interviews have all helped confirm the assertion that divergence of technology, and therefore of standardization, is a primary barrier to information exchange at a technical level. We can also conclude from discussions that while the landscape is fragmented, applied best-practice approaches are rare.

Additionally, the absence of common open-source ontologies is negatively affecting the way businesses are engaging with each other, and ultimately reducing the amount of information that can be shared.

### 8.1.3 Is seamless information exchange achievable? — research question E

At present, it appears the root cause of many of the technical barriers to enterprise integration is due to two key things.

1. The majority of systems still define their own data model and therefore systems are continually communicating with a different 'language'. The need to use a bespoke data model may be attributed to the fact that the Internet as a whole has not yet evolved to the point where models have been set and formalized, although this may improve over time. There are certainly examples of real-world data models that are in play and continue to be used.
2. Additionally, however, systems are evolving beyond data stores to support business intelligence and that will force even further diversification between custom business solutions. In time that, too, can be addressed through a common business intelligence execution model.

As we understand what causes the absence of seamless information exchange, we can conclude that it must also be possible to achieve it. Seamless information exchange is a by-product of solving these two technical barriers. The consolidation of these two models at both a data and business intelligence level will alleviate non-technical integration barriers, as inter-company communication is also following a similarly consistent approach.

### 8.1.4 How do we establish a global communications ecosystem? — research question A

Finally, with these considerations in mind, this research concludes that a global integration platform as part of a global communications ecosystem is an important proposition that would improve the quality of integration solutions, the rate at which they can be constructed and their rate of adoption.

This research has taken important holistic steps to answering this question, including investigating how to address different business levels (strategy, business and technology), technology concerns, vendor support, global standards and user motivation.

Additionally, this research concludes that a solution needs to be based on firm open standards adopted by a community and backed by vendor support in order to achieve mass adoption. We also conclude that work must be done at the network level to promote a safe and reliable transport for data.

Finally, we conclude that any global communications ecosystem must support the sharing of integration mappings in a social manner that will apply the necessary motivation for the ecosystem to grow and evolve over time.

## 8.2 Contributions

This analysis has taken important steps towards the vision of establishing a global communications ecosystem.

To address integration barriers at scale, this research argues that Hasselbring's three-dimensional integration model should be extended to four dimensions by adding architecture, which addresses scale as well as any other architectural concern that is impacted by scale. This adjustment to the model is important to extend the current thinking on integration barriers beyond the local network environment (Hasselbring, 2000).

The literature review has been extensive, with this research analyzing over 500 academic articles and books as the basis to select the most relevant considerations for the discussion. Integration is a highly complex topic with many factors that need to be taken into account when designing a replacement platform for scale. This analysis takes an important step by providing a foundation for future discussion on the vision of a global communications ecosystem.

Establishing assessment criteria for the ecosystem is an important step to aid analysis and discussion. In the future, these criteria should be continuously scrutinized to the point where they can be used as the scorecard for assessment of integration technology at scale.

By taking an 'enterprise architecture' approach, the initial set of reviews enabled us to establish the set of known integration platforms as the 'current state' view of integration platform solutions. Evolution in this area can happen organically by reflecting on the current state, planning a future state and performing a gap analysis between the two that will help to determine what research is needed to achieve the future state.

The act of designing the reactor network has demonstrated this process and has delivered a future state concept that can now be debated and accepted as the current state, or replaced with a better future state alternative design. Design of the reactor was an attempt to address research question C, What would a global integration platform look like?

Another important contribution this research has made is to continue the debate around why and how to cement peer-to-peer protocols within the OSI model so they are used as a transport by the general Internet community. With lower-level support for peer-to-peer in place, the basis for a global communications ecosystem is set.

## 8.3 Limitations

Although the research has achieved its goals, there were some unavoidable limitations. Because of the subject matter, qualified participants were limited to those professionals with first-hand experience of software integration. With only a fraction of professionals qualifying it made locating valid participants challenging, resulting in a less-than-optimal sample size. With more time available, a larger sample size could be constructed.

The time limit on the research also meant the number of focus groups was restricted to four. Each focus group was therefore tasked with broad-ranging issues to consider over short periods. The research would benefit from having more focus groups with the time to absorb the content material, which evolved significantly over the course of 12 months.

In addition, the research would be improved by conducting a more in-depth study into real-world integration platforms used within organizations. It is expected this would be better achieved over the course of a longer period of research, e.g. a doctorate programme where the overhead for participating organizations would be less disruptive for their business.

Although social media was employed globally to attract participants, due to the point of origin (location of the researcher's network), the majority of participants came from New Zealand. This represents a flaw in the research, as we cannot conclude that integration within New Zealand reflects the broader worldwide context. It could be that New Zealand's smaller scale means integration is conducted in a different way to that of large organizations in the USA. Attempts were made to stratify according to organization size; however, this can act as an indicator only given the current sample size.

## 8.4 Future Work

There is currently a lack of academic research into integration with most studies having been conducted in a pre-cloud Internet environment.

Although the concept of a global integration platform might seem fantastical, thinking at scale causes us to address concerns, which may become problematic in different contexts.

The lack of a common business ontology is a serious problem within the business and Internet communities. A common ontology must not be proprietary and therefore should originate from within academia or the open-source community. Establishing a common ontology should mimic the work done in the field of genetics where modelling of the human genome is being conducted. Likewise, there needs to be research on how to string together the many pocket ontologies that exist globally to form a common ontology which businesses can adopt, leading to greater levels of communication and interoperability.

At the computer science level there is an opportunity to research the implementation of a high-performance, asynchronous binary protocol that can transport data entities more efficiently and be used in conjunction with peer-to-peer mechanics. When we consider the possibility of receiving data from any number of providers, it raises the very real problem of trust and the structure of the network itself needs to be designed from the ground up.

When different network solutions are proposed (this research suggested one possible implementation), they will need to be tested for theoretical limits against current and future loading expectations.

Additionally, we must consider how far away we are from a time when we not only inject data, but also the business intelligence that accompanies it. Containers and virtualization allow us to inject logic without fear of affecting the host system. For example, let us say as concertgoers we want to book tickets only if we can get front row seats. Currently each user or system would have to integrate with each booking agency, query a server for available tickets and then post a purchase order. There are many points of failure. It would be much easier if you could send the order to the integration platform, accompanied by some logic, and receive a ticket response. It may sound far-fetched to sculpt and inject such logic, but in a world where both a common ontology and containers exist, it becomes a very real proposition.

Finally, one challenging area at the heart of this research topic is the issue of motivation. There needs to be much more research into determining how to address motivation, and how to encourage developers to conform to standards. Will a global network that centrally manages models, definitions, transformations and business intelligence actually have the desired effect?

## 8.5 Conclusion

As we end the research, we take this final opportunity to summarize the research and its findings. Enterprise integration is an extremely complex field, fraught with diversity and ever-changing technology. Any proposed solution to integration must be both sophisticated enough to solve identified integration barriers, and presumptuous enough to pretend as though it will hold for unknown technologies in the future.

The only true solution to enterprise integration, and therefore achieving information at the speed of thought, is for all parties to be communicating the same language. Before technical solutions can be constructed businesses must evolve to the point where they can define themselves in a globally consistent manner. Before this occurs, any other proposed technical solution is merely covering up the key underlying issue.

# 9

# References


Airamo, O. Peer-to-Peer Systems Comparison with Respect to Operating Layer.

Al Mosawi, A., Zhao, L., & Macaulay, L. (2006). A model driven architecture for enterprise application. In *Proceedings of the 39th Hawaii International Conference on System Science.*

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). Enterprise application integration. In *Web Services* (pp. 67-92). Springer Berlin Heidelberg.

Amir, Y., Danilov, C., Musuăloiu-Elefteri, R., & Rivera, N. (2010). The SMesh wireless mesh network. *ACM Transactions on Computer Systems (TOCS)*, *28*(3), 6.

Aubert, B., Vandenbosch, B., & Mignerat, M. (2003). Towards the measurement of process integration. *Cahier du GReSI no*, *3*, 06.

Bauknecht, K., Min Tjoa, A., & Quirchmayr, G (2003). Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies, EC- Web 2003, Prague, Czech Republic, September 2003.

Brown, C. V. (1999). Horizontal mechanisms under differing IS organization contexts. *Mis Quarterly*, 421-454.

Buschmann, F., Henney, K., & Schimdt, D. (2007). *Pattern-oriented Software Architecture: On Patterns and Pattern Language* (Vol. 5). John Wiley & Sons.

"Business Process Definition" (n.d.) in Appian. Retrieved May 5, 2015, from http://www.appian.com/about-bpm/definition-of-a-business-process/

Calvert, C. (1997). Regulating Cyberspace: Metaphor, Rhetoric, Reality, and the Framing of Legal Options. Hastings Comm. & Ent. LJ, 20, 541.

Campista, M. E. M., Esposito, P. M., Moraes, I. M., Costa, L. H. M., Duarte, O. C., Passos, D. G., ... & Rubinstein, M. G. (2008). Routing metrics and protocols for wireless mesh networks. *Network, IEEE*, *22*(1), 6-12.

Chappell, D. (2004). *Enterprise service bus.* "O'Reilly Media, Inc."

CIRANO Working Papers #2003s-04, University of Montréal, 2003.

Cisco Systems. (2009, June). White Paper: Cisco Visual Networking Index: Forecast and Methodology, 2008-2013.

Clarke, M., Blair, G. S., Coulson, G., & Parlavantzas, N. (2001, November). An efficient component model for the construction of adaptive middleware. In Middleware 2001 (pp. 160-178). Springer Berlin Heidelberg.

Cooper Berdayes, L., & Berdayes, V. (1998). The information highway in contemporary magazine narrative. Journal of Communication, 48(2), 109-124.

"CRM History" (n.d.) in CRM Switch. "A Brief History of Customer Relationship Management". Retrieved May 3, 2015, from http://www.crmswitch.com/crm-industry/crm-industry-history/.

David, P. A. (2001). The Evolving Accidental Information Super-Highway. *Oxford Review of Economic Policy*, *17*(2), 159-187.

"Digital Nervous System" (n.d.) in Wikipedia. Retrieved May 5th, 2015, from https://en.wikipedia.org/wiki/Digital_nervous_system

Du, Y., Peng, W., & Zhou, L. (2008, December). Enterprise Application Integration: An Overview. In *Intelligent Information Technology Application Workshops, 2008. IITAW'08. International Symposium* (pp. 953-957). IEEE.

"Email" (n.d.) in Wikipedia. Retrieved June 1st, 2015, from https://en.wikipedia.org/wiki/Email

Emmerich, W. (2000, May). Software engineering and middleware: a roadmap. In *Proceedings of the Conference on The future of Software engineering* (pp. 117-129). ACM.

Eshel, E. (2000). Enterprise application integration in financial services. *Financial Services Information Systems*, Auerbach Publications, New York, 469-483.

Ferguson, D. F. (2007). The internet service bus. In *On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS* (pp. 5-5). Springer Berlin Heidelberg.

Foster, I., Kesselman, C., Nick, J. M., & Tuecke, S. (2002). Grid services for distributed system integration. *Computer*, *35*(6), 37-46.

Fowler, M. (2002). *Patterns of enterprise application architecture*. Addison-Wesley Longman Publishing Co., Inc..

Gable, J. (2002). Enterprise application integration. *Information Management*, *36*(2), 48.

Gates, B. (1999). Business @ the speed of thought. *Business Strategy Review*, *10*(2), 11-18.

Gibbs, A. (1997). Focus Groups. Social Research Update, 19. University of Surrey.

Gibbs, W. W. (1994). Software's chronic crisis. *Scientific American*, *271*(3), 72-81.

Gligor, G., & Teodoru, S. (2011). Oracle Exalytics: Engineered for Speed-of-Thought Analytics. *Database Systems Journal*, *2*(4), 3-8.

Goel, A. (2006). Enterprise integration EAI vs. SOA vs. ESB. *Infosys Technologies White Paper*, *87*.

Gokhale, A., Balasubramanian, K., Krishna, A. S., Balasubramanian, J., Edwards, G., Deng, G., ... & Schmidt, D. C. (2008). Model driven middleware: A new paradigm for developing distributed real-time and embedded systems. *Science of Computer programming*, *73*(1), 39-58.

Hasselbring, W. (2000). Information system integration. *Communications of the ACM*, *43*(6), 32-38.

Henning, M. (2008). The rise and fall of CORBA. *Communications of the ACM*, *51*(8), 52-57.

Hohpe, G., & Woolf, B. (2003). Enterprise integration patterns: Designing, building, and deploying messaging solutions. Addison-Wesley Professional.

Huat Lim, S., Juster, N., & de Pennington, A. (1997). Enterprise modelling and integration: a taxonomy of seven key aspects. *Computers in Industry*, *34*(3), 339-359.

Irani, Z., & Love, P. E. (2001). Information systems evaluation: past, present and future. *European Journal of Information Systems*, *10*(4), 183-188.

Irani, Z., Themistocleous, M., & Love, P. E. (2003). The impact of enterprise application integration on information system lifecycles. *Information & Management*, *41*(2), 177-187.

Johannesson, P., Wangler, B., & Jayaweera, P. (2000, June). Application and process integration—concepts, issues, and research directions. In *Information Systems Engineering Symposium CAiSE*.

Kamel, M., Scoglio, C., & Easton, T. (2007). Optimal topology design for overlay networks. In *NETWORKING 2007. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet* (pp. 714-725). Springer Berlin Heidelberg.

Kenway, J. (1996). The information superhighway and post-modernity: The social promise and the social price. *Comparative Education*, *32*(2), 217-232.

Kishore, R., Zhang, H., & Ramesh, R. (2006). Enterprise integration using the agent paradigm: foundations of multi-agent-based integrative business information systems. *Decision Support Systems*, *42*(1), 48-78.

Klefstad, R., Krishna, A. S., & Schmidt, D. C. (2002). Design and performance of a modular portable object adapter for distributed, real-time, and embedded CORBA applications. In *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE* (pp. 549-567). Springer Berlin Heidelberg.

Klischewski, R. (2004). Information integration or process integration? How to achieve interoperability in administration. In *Electronic Government* (pp. 57-65). Springer Berlin Heidelberg.

Kon, F., Costa, F., Blair, G., & Campbell, R. H. (2002). The case for reflective middleware. *Communications of the ACM*, *45*(6), 33-38.

Kuck, N., Kuck, H., Lott, E., Rohland, C., & Schmidt, O. (2002). SAP VM Container: Using Process Attachable Virtual Machines to Provide Isolation and Scalability for Large Servers. *Article, SAP AG, Walldorf, Germany*, *2*.

Kühn, H., Bayer, F., Junginger, S., & Karagiannis, D. (2003). Enterprise model integration. In *E-Commerce and Web Technologies* (pp. 379-392). Springer Berlin Heidelberg.

Lam, W., & Shankararaman, V. (2004). An enterprise integration methodology. *IT professional*, *6*(2), 40-48.

Lam, W. (2005). Barriers to e-government integration. *Journal of Enterprise Information Management*, *18*(5), 511-530.

Lam, W., & Shankararaman, V. (2007). *Enterprise architecture and integration: Methods, implementation and technologies*. USA: IGI Global. doi:10.4018/978-1-59140-887-1.

Lankhorst, M. M. (2004). Enterprise architecture modelling—the issue of integration. *Advanced Engineering Informatics*, *18*(4), 205-216.

Lenzerini, M. (2002, June). Data integration: A theoretical perspective. In Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (pp. 233-246). ACM.

Linthicum, D. (1999). *Enterprise application integration*. Massachusetts, USA: Addison-Wesley.

Liu, Y., Liu, Q., Wang, Y., & Qin, Z. (2012, March). Inferring Client Distribution by Using Dyad BitTorrent Network Data. In *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference* (Vol. 1, pp. 615-619). IEEE.

Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., & Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys & Tutorials, IEEE, 7*(2), 72-93.

Lublinsky, B. (2001). Achieving the ultimate EAI implementation. *eAI Journal, 3*(2).

Malone, T. W., Crowston, K., & Herman, G. A. (Eds.). (2003). *Organizing business knowledge: the MIT process handbook*. MIT press.

Menge, F. (2007, August). Enterprise service bus. In *Free and open source software conference* (Vol. 2, pp. 1-6).

Michelson, B. M. (2006). Event-driven architecture overview. *Patricia Seybold Group, 2*.

"Middleware Services" (n.d.) in Technotec. Retrieved May 5, 2015, from http://www.technotec.com/middleware_services.html

Morgan, D. L. (1997). *The focus group guidebook* (Vol. 1). Sage Publications.

Niemi, E. (2008). Enterprise architecture benefits: Perceptions from literature and practice. Evaluation of enterprise and software architectures: critical issues, metrics and practices:[AISA Project 2005-2008]/Eetu Niemi, Tanja Ylimäki & Niina Hämäläinen (eds.). Jyväskylä: University of Jyväskylä, Information Technology Research Institute, 2008.-(Tietotekniikan tutkimusinstituutin julkaisuja, ISSN 1236-1615; 18). ISBN 978-951-39-3108-7 (CD-ROM).

Ortiz, A., Lario, F., & Ros, L. (1999). Enterprise integration—business processes integrated management: a proposal for a methodology to develop enterprise integration programs. *Computers in Industry, 40*(2), 155-171.

Pappa, D., & Stergioulas, L. K. (2008). The emerging role of corporate information systems: An example from the area of business process-oriented learning. *Int. Journal of Business Science and Applied Management, 3*(2).

Paulheim, H. (2009). *Ontologies for user interface integration* (pp. 973-981). Springer Berlin Heidelberg.

Pautasso, C. (2008). Bpel for rest. In *Business Process Management* (pp. 278-293). Springer Berlin Heidelberg.

Peltz, C., 2003: Web Services Orchestration and Choreography, Computer, 36(10). 46-52.

Pérez, H., & Gutiérrez, J. J. (2014). A survey on standards for real-time distribution middleware. *ACM Computing Surveys (CSUR), 46*(4), 49.

Perrey, R., Johnston, A., Lycett, M., & Paul, R. (2004). Value propositions: a new conceptualisation for integration. *Journal of Enterprise Information Management, 17*(2), 142-163.

Puschmann, T., & Alt, R. (2001, January). Enterprise application integration-the case of the Robert Bosch Group. In *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference* (pp. 10). IEEE.

Ring, K., & Ward-Dutton, N. (1999). Enterprise application integration: Making the right connections. *Ovum Ltd, London, UK*.

Rouvoy, R., Barone, P., Ding, Y., Eliassen, F., Hallsteinsen, S., Lorenzo, J., ... & Scholz, U. (2009). Music: Middleware support for self-adaptation in ubiquitous and service-oriented environments. In Software engineering for self-adaptive systems (pp. 164-182). Springer Berlin Heidelberg.

Samtani, G., Healey, M., & Samtani, S. (2002). *B2B integration: A practical guide to collaborative e-commerce*. London: Imperial College Press.

Schmidt, J., Morrison, A., & Gruneus, E. (2001). *U.S. Patent Application 09/824,690*.

Schmidt, M. T., Hutchison, B., Lambros, P., & Phippen, R. (2005). The enterprise service bus: making service-oriented architecture real. *IBM Systems Journal*, *44*(4), 781-797.

Schmidt, R., Lyytinen, K., & Mark Keil, P. C. (2001). Identifying software project risks: An international Delphi study. *Journal of management information systems*, *17*(4), 5-36.

Schulte, R. (2004). Event-driven architecture: The next big thing. In *Application integration & web services summit*. Los Angeles, USA: Gartner.

Shankararaman, V., Tan, K. W., Thonse, S., Gupta, M., & Deshmukh, N. (2007). Aligning IT Solutions with Business Processes: A Methodological Approach.

Shankararaman, V., & Megargel, A. (2013). Enterprise Integration: Architectural Approaches. *Service-Driven Approaches to Architecture and Enterprise Integration*, 67.

Thekkath, C. A., & Levy, H. M. (1993). Limits to low-latency communication on high-speed networks. *ACM Transactions on Computer Systems (TOCS)*, *11*(2), 179-203.

Themistocleous, M., & Irani, Z. (2001). Benchmarking the benefits and barriers of application integration. *Benchmarking: An International Journal*, *8*(4), 317-331.

Themistocleous, M., Irani, Z., & O'Keefe, R. M. (2001). ERP and application integration: exploratory survey. *Business Process Management Journal*, *7*(3), 195-204.

Tovée, M. J. (1994). Neuronal Processing: How fast is the speed of thought?. *Current Biology*, *4*(12), 1125-1127.

Trastour, D., Preist, C., & Coleman, D. (2003, September). Using semantic web technology to enhance current business-to-business integration approaches. In *Enterprise Distributed Object Computing Conference, 2003. Proceedings. Seventh IEEE International* (pp. 222-230). IEEE.

Tse, W. Enterprise Application Integration. UCL Computer Science. http://www. cs. ucl. ac. uk/staff/ucacwxe/lectures/3C05-03-04/EAI. pdf. (Accessed 2015)

Van de Maele, F., & Díaz, A. (2008, January). Towards a Scalable and Collaborative Information Integration Platform and Methodology. In *On the Move to Meaningful Internet Systems: OTM 2008 Workshops* (pp. 220-229). Springer Berlin Heidelberg.

Van der Aalst, W. M., Ter Hofstede, A. H., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and parallel databases*, *14*(1), 5-51.

W3C Web Standards (n.d.) in W3C. Retrieved June 10, 2015, from http://www.w3.org/standards/

W3C WSDL Standard (n.d.) in W3C. Retrieved June 11, 2015, from http://www.w3.org/TR/wsdl

Woods, D. (2003). *Enterprise services architecture*. "O'Reilly Media, Inc."

Yu, Q., Liu, X., Bouguettaya, A., & Medjahed, B. (2008). Deploying and managing Web services: issues, solutions, and directions. *The VLDB Journal—The International Journal on Very Large Data Bases*, *17*(3), 537-572.

Zahir Irani, P. E. (2000). The propagation of technology management taxonomies for evaluating investments in information systems. *Journal of Management Information Systems*, *17*(3), 161-177.

Ziegler, P., & Dittrich, K. R. (2004, August). Three decades of data integration — All problems solved? In *IFIP congress topical sessions* (pp. 3-12).

# A

# Appendix A – Survey Construction

| Question | Purpose |
|---|---|
| ***A1. What is your email address?** | Asked solely to ensure uniqueness of respondent, thus maintaining the integrity of the survey. |
| ***A2. What is the size of your company?** | It is important to understand the scale demographic of the participant as scale (or lack thereof) may play a part in determining an integration approach and/or attitudes towards integration in general. We will look to use this factor to cross-analyze for patterns. |
| ***A3. Does your company (or on behalf of a client) integrate information with a third-party system?** | To determine whether the integration conducted by the participant is done solely within the organization or with a third party. |
| ***A4. Compared to five years ago, do you integrate with fewer, more or the same number of systems?** | To determine whether more integration is occurring now than previously. The assumption is that the 'cloud' has been a catalyst for integration and so this question aims to confirm or refute that assumption. |
| ***A5. Compared to five years ago is it less, more or equally difficult to integrate systems/solutions?** | There is an assumption that modern web standards have made integration easier. However, there are also many more ways to integrate than previously, so this question aims to determine the 'net' difficulty involved in applying integration solutions in today's operating environment. |
| ***A6. Are you less, more or equally concerned (the same) with where your information is stored than you were five years ago?** | There is an assumption that the cloud has trained people to be less concerned about where their data sits. If this is true it has a significant bearing on accepted integration solutions. This question aims to confirm or refute that assumption. |
| ***A7. Which level of business do you most strongly associate with?** | To identify which level of business (according to the EA3 framework) an individual lies within, and to determine which secondary section the participant will answer |

| | |
|---|---|
| | (Section B - Strategy, Section C - Business or Section D - Technology). |
| ***A8. From your perspective, how often is 'strategy' a barrier to integration?** | The next three questions aim to determine which business level represents the greatest frequency of barriers to integration. Specifically this will be tested against a participant's own business level to see if any correlation exists. This question specifically tests the opinion on strategy as a barrier to integration. |
| ***A9. From your perspective, how often is 'business' a barrier to integration?** | To specifically test the opinion on business as a barrier to integration. |
| ***A10. From your perspective, how often is 'technology' a barrier to integration?** | To specifically test the opinion on technology as a barrier to integration. |
| ***A11. Please rank the business levels in order from the greatest overall barrier to integration to the smallest barrier to integration. (Consider the scenario that the business needs to integrate immediately with a third-party system to win a tender. What prevents you from achieving that goal in a matter of moments?)** | To ask the participant to reflect on and summarize the overall ranking of each business level in terms of its impact on integration. |
| ***A12. Please rank the following three challenges in order from most difficult to least difficult to address.** | To address Hasselbring's integration problem dimensions by asking the candidate in language they can relate to whether distribution, autonomy or heterogeneity represents the greatest challenge for them. |

Section B – Strategy

| | |
|---|---|
| ***B1. Impact of strategic barriers on integration** | To determine the strategy participant's opinion on where the key strategic areas of obstruction are towards integration. |
| **Ability to share goals and objectives** | The ability to share goals and objectives will depend on the relationship between the two integrators. A loosely bonded relationship where each party has a different agenda may cause integration problems. |
| **Ambitious milestones** | If the project objectives are too ambitious, e.g. time, scope and resource are not aligned correctly, then the integration is likely to fail. |
| **Lack of ownership** | An integration project may be instantiated but if no one is driving it, and if other projects are given higher prioritization, the project may fail. |

| | |
|---|---|
| **Lack of implementation guidelines** | An organization may have a preferred approach or policies for integration. However, if these are not supplied or do not exist then implementation will have to choose an approach which may not be consistent with an overarching vision. |
| **Funding** | A lack of suitable financial commitment to an integration project will result in shortcuts, which introduces risk to the project. Over a large number of integration projects this becomes problematic. |
| ***B2. Are there other strategic factors which present a barrier to integration?** | To determine the possibility of other strategic barriers to extend the result set. |

Section C – Business

| | |
|---|---|
| ***C1. Impact of business barriers on integration** | To determine the business participant's opinion on the key business areas of obstruction to integration. |
| **Data ownership** | Does data ownership cause a problem? For example, if a third party manages a data source and they change the schema during or after the implementation of a project. |
| **Policy evolution** | Does change of policy cause a problem with implementation and maintenance? For example, a policy changes midway through a project. |
| **Lack of business readiness** | Is the business ready to consume the data provided as a result of the integration project? Although technically complete, the project may still be registered as a failure. |
| **Pace of integration adoption** | Can the business keep up with the rate of change that integration brings? For example, new data sets arriving and retiring, and available in a timely manner. |
| **Absence of champion** | Is there someone within the business who truly believes in the integration project and who will drive it internally? |
| **Common terminology** | Are the two integrating parties talking the same language or are communication issues slowing the integration process? |
| ***C2. Are there other business factors which present a barrier to integration?** | What other business factors may be problematic when implementing an integration project? |

Section D - Technology

| | |
|---|---|
| ***D1. Impact of technology barriers on integration** | To determine the technology participant's opinion on the key technology areas of obstruction to integration. |
| **Architectural interoperability** | Are the two architectures involved similar enough in structure that they can be integrated easily? |
| **Incompatible data standards** | Are the data standards close enough that they can be integrated easily, e.g. schemas can be applied to both and transformed? |
| **Incompatible technical standards** | Are the technical standards close enough that they can be integrated easily, e.g. one party can provide a REST interface and the other can consume a REST interface? |
| **Different security models** | Are the security models compatible, e.g. one party provides OAuth support and the second party can use OAuth, or both parties support an Access Control Layer (ACL) approach? |
| **Inflexibility of legacy systems** | Do legacy systems cause a problem as they are unable to be upgraded to use modern standards? |
| ***D2. Are there other technology factors which present a barrier to integration?** | What other technology factors may be problematic when implementing an integration project? |
| ***D3. How often do you consider industry standards (e.g. ISO or W3C, etc.) when implementing a technical solution?** | One of the problems we have with integration in some cases is that people do not adhere to standards, e.g. due to commercial pressures or vendor guidance. The only way forward to achieve *business @ the speed of thought*, however, is through a common standardized approach. |
| ***D4. Do you use a third-party or proprietary integration platform/framework/technology, e.g. CORBA?** | To determine whether the candidate is using a stand-alone technology to broker integration. This is an important question as it establishes both the level of maturity and willingness to work with such platforms. |
| ***D5. Do you develop custom or bespoke integration solutions in-house?** | This is the counter question to the above — whether the participants develop their own integration solutions in-house. Note that |

| | |
|---|---|
| | these two questions are not mutually exclusive. |
| ***D6. Do you use a platform/framework to distribute data?** | Addresses Hasselbring's integration problem dimension of distribution. It is more focused on routing data, e.g. via some organized queueing framework. |
| ***D7. Do you use a rules engine, e.g. Amazon Lambda, to execute business intelligence?** | Addresses Hasselbring's integration problem dimension for autonomy. Where does the business intelligence and the overall application control lie? If it lies within scripted logic, then that logic can be redistributed and scaled more easily. |
| ***D8. Do you use a transformation technology, e.g. XSLT, to transform between source and destination data formats?** | To determine how advanced the integration level is. Use of a transformation mechanism means change of data model and/or message format requires little change to accommodate. However, there are some overheads in reaching this level of maturity. |
| ***D9. Do you use XML as the basis for one of your message formats?** | There are two main movements with message formats in modern technologies. XML has a long history and is still popular for data exchange. However, the proliferation of JavaScript in web applications and MVVM as an architectural model means that JSON is now also a significant contender. |
| ***D10. Do you consume or publish REST services?** | There is an industry trend in which systems are utilizing REST, an old standard for simple data exchange, over something more structured such as SOAP. REST is easy to use and therefore integration implementation has a shorter turnaround. |
| ***D11. Have you implemented middleware within your organization, or on behalf of another organization, to orchestrate solutions?** | Orchestration relates to Hasselbring's autonomy dimension. Where does the control lie? Without autonomy you sit with a point-to-point implementation which is problematic at scale. |
| ***D12. Have you implemented a service-oriented architecture within your organization or on behalf of another organization?** | A service-oriented architecture provides many benefits, not the least of which is separation of concern. This componentized thinking is conducive to interoperability with the cloud and other third-party service providers. |
| ***D13. Have you implemented asynchronous technologies within your organization or on** | There is a current shift of thinking where architects are realizing that a lot of system |

| | |
|---|---|
| **behalf of your customers? For example, queue technologies such as RabbitMQ.** | overheads are due to the requirement of synchronous connectivity. As the number of connected systems rises, an asynchronous model reduces system overheads. In many cases the slight delay in time of delivery is acceptable. |
| ***D14. Do you integrate with the 'cloud', i.e. connect and share data with a service that runs in the cloud?** | A high-level question designed to see if cloud adoption has been considered. This will be correlated with some of the other technology questions to see if the cloud is promoting certain trends. |
| ***D15. If there was a best-practice, universally accepted, non-proprietary (or open-source) data model (e.g. an ISO standard for financial transactions) for your industry would you adopt it?** | This is the single most important question relating to Hasselbring's heterogeneity dimension. Would developers adopt a best-practice model if one existed for their domain? |
| ***D16. Please rank the following two statements in order of your level of agreement with them (rank = 1 means you agree with it more than rank = 2).**<br>• Requirements of applications vary greatly ranging from data that can be well stored in spreadsheets, to data that does indeed require DBMS storage.<br>• Data must always be stored and managed in DBMS systems. | To test a participant's attitude towards traditional architecture with a backend database server versus the more recent thinking where data is more mobile and can be stored in different states at different locations. |
| ***D17. Please rank the following two statements in order of your level of agreement with them (rank = 1 means you agree with it more than rank = 2).**<br>• The database must always provide for and manage the structure and semantics of the data through formal schemas.<br>• The database can be nothing more than intelligent storage. Data could be stored generically and rendered accordingly by the client. | To further test the traditional n-tier thinking where each tier has a function (e.g. presentation, application, entity, persistence) versus a lighter two-tier structure where the client can manage more of the MVC functionality — e.g. MVVM approach. |
| ***D18. Please rank the following two statements in order of your level of agreement with them (rank = 1 means you agree with it more than rank = 2).**<br>• Managing multiple schemas from independent sources and interrelationships | Focuses specifically on attitude towards schema management. It relates to the traditional persistence management of having a database server with a fixed schema versus having a more dynamic form of persistence — e.g. via an ORM or NoSQL approach. |

| | |
|---|---|
| between them, i.e., 'schema chaos' is inevitable and unavoidable.<br>• Any imposition of schema can be done by the clients, as and when needed by applications. | |

# B

# Appendix B – Actual Survey

## Introduction

**This Survey is conducted in conjunction with *Massey University, Auckland New Zealand.***

**Important!! - Please do not continue this survey unless you have:**
**1. Worked in an organization that has implemented a software integration project (either for your company or for a client). This may include the implementation of a new system, connection to a cloud service, or data sharing with other partner organizations.**
**2. Have been involved with integration project for a period no less than 1 year**

**The Survey**

**Bill Gates once penned a book called "Business at the speed of thought". Although we have made numerous advances with business information exchange, it still remains a complex and challenging process to share data in a real-time fashion with other business entities.**

**Please help us to understand why this is the case. For the purpose of the survey ahead the following terms will be used as follows:**

**Strategy: CxO level strategic governance including CEO, CIO, CTO etc.**
**Business: Day to day business operation, including Sales, Marketing, Business Analysts etc.**
**Technology: Delivery of Software and Information Solutions including Solution Architect, Systems Administrator, Developer etc.**

## General

**General details regarding integration which apply to all levels of business**

* 1. what is your email address ? (Please note, this is to ensure uniqueness only. Your email address will NOT be used in any way other to contact you if you win a prize. You will not be contacted otherwise and your email will NOT be given to any other party)

* 2. What is the size of your company?

* 3. Does your company (or on behalf of a client) integrate information with a 3rd party system?

* 4. Compared to 5 years ago, do you integrate with less, more or the same number of systems?

| Less | The same | More | I Don't Know |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

* 5. Compared to 5 years ago is it less, more or equally difficult to integrate systems/solutions?

| Less | The same | More | I Don't Know |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

* 6. Are you less, more or equally concerned (the same) with where your information is stored than you were 5 years ago?

| Less | The same | More | I Don't Know |
|---|---|---|---|
| ○ | ○ | ○ | ○ |

* 7. Which level of business do you most strongly associate with:

- ○ Strategy (e.g. CxO including CEO, CIO, CTO etc)
- ○ Business (e.g. Business Analyst, Sales, Marketing, Product or Program Manager)
- ○ Technology (e.g. Architect, Developer, Integration, Systems Administrator)

* 8. From your perspective, how often is "Strategy" a barrier to integration?
(This may include Goals, Objectives, Milestones, Governance, Implementation Guidance, Funding etc.)

| Never | Rarely | Occasionally | Frequently | Very Frequently | I Don't Know |
|---|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ | ○ |

* 9. From your perspective, how often is "Business" a barrier to integration?
(This may include Data Ownership, Policy Evolution, Business Readiness, Absence of a Champion, Lack of common business terminology)

| Never | Rarely | Occasionally | Frequently | Very Frequently | I Don't Know |
|---|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ | ○ |

* 10. From your perspective, how often is "Technology" a barrier to integration?
(This may include Standards, Platforms, Frameworks, Proprietary formats, Legacy Systems, Technologies etc.)

| Never | Rarely | Occasionally | Frequently | Very Frequently | I Don't Know |
|---|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ | ○ |

* 11. Please rank the business levels in order from the greatest barrier to integration to the smallest barrier to integration.
(Consider the scenario that the business needs to integrate immediately with a 3rd party system in order to win a tender. What prevents you from achieving it in a matter of moments?)

| | |
|---|---|
| | Strategy integration barriers |
| | Business integration barriers |
| | Technology integration barriers |

* 12. Please rank the following three challenges in order from most difficult to least difficult to address.

| | |
|---|---|
| | Ability to distribute data to other systems |
| | Ability to transform data once acquired |
| | Ability to action data in intelligent ways |

## Strategy

13. Impact of strategic barriers to integration

| | Never | Rarely | Occasionally | Frequently | Very Frequently | I don't know |
|---|---|---|---|---|---|---|
| Ability to share Goals and Objectives | ○ | ○ | ○ | ○ | ○ | ○ |
| Ambitious Milestones | ○ | ○ | ○ | ○ | ○ | ○ |
| Lack of Ownership | ○ | ○ | ○ | ○ | ○ | ○ |
| Lack of Implementation Guidelines | ○ | ○ | ○ | ○ | ○ | ○ |
| Funding | ○ | ○ | ○ | ○ | ○ | ○ |

14. Are there other strategic factors which present a barrier to integration?

## Business

15. Impact of business barriers to Integration

| | Never | Rarely | Occasionally | Frequently | Very Frequently | I don't know |
|---|---|---|---|---|---|---|
| Data ownership | ○ | ○ | ○ | ○ | ○ | ○ |
| Policy evolution | ○ | ○ | ○ | ○ | ○ | ○ |
| Lack of business readiness | ○ | ○ | ○ | ○ | ○ | ○ |
| Pace of integration adoption | ○ | ○ | ○ | ○ | ○ | ○ |
| Absence of champion | ○ | ○ | ○ | ○ | ○ | ○ |
| Lack of common terminology | ○ | ○ | ○ | ○ | ○ | ○ |

16. Are there other business factors which present a barrier to integration?

## Technology

17. Impact of technology barriers to Integration

| | Never | Rarely | Occasionally | Frequently | Very Frequently | I don't know |
|---|---|---|---|---|---|---|
| Architectural Interoperability | ○ | ○ | ○ | ○ | ○ | ○ |
| Incompatible data standards | ○ | ○ | ○ | ○ | ○ | ○ |
| Incompatible technical standards | ○ | ○ | ○ | ○ | ○ | ○ |
| Different security models | ○ | ○ | ○ | ○ | ○ | ○ |
| Inflexibility of legacy systems | ○ | ○ | ○ | ○ | ○ | ○ |

18. Are there other technology factors which present a barrier to integration?

19. How often do you consider industry standards (e.g. ISO or WC3 etc.) when implementing a technical solution?

| Never | Rarely | Occasionally | Frequently | Very Frequently | I Don't Know |
|---|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ | ○ |

20. Do you use a 3rd party or proprietary integration platform/framework/technology e.g. Corba?

If so what is it ?

21. Do you develop custom or bespoke integration solutions in-house?

22. Do you use a platform/framework to distribute data

23. Do you use a rules engine e.g. Amazon Lambda to execute business intelligence

If so what is it ?

24. Do you use a transformation technology e.g. XSLT to transform between source and destination data formats

If so what is it ?

25. Do you use XML as the basis for one of your message formats ?

26. Do you consume or publish REST Services ?

27. Have you implemented middleware within your organization, or on behalf of another organization to orchestrate solutions ?

28. Have you implemented an SOA Architecture within your organization or on behalf of another organization?

29. Have you implemented asynchronous technologies within your organization or on behalf of your customers e.g. queue technologies such as RabbitMQ

30. Do you integrate with the "Cloud" i.e. connect and share data with a service that runs in the cloud

31. If there were a best practice universally accepted non-proprietary (or open source) data model (e.g. an ISO standard for financial transactions) for your industry would you adopt it ?

32. Please rank the following two statements in order of your level of agreement with them (rank = 1 means you agree with it more than rank = 2)

- [ ] Data must always be stored and managed in DBMS systems
- [ ] Requirements of applications vary greatly ranging from data that can well be stored in spreadsheets, to data tha does indeed require DBMS storage.

33. Please rank the following two statements in order of your level of agreement with them (rank = 1 means you agree with it more than rank = 2)

- [ ] The database must always provide for and manage the structure and semantics of the data through formal schemas
- [ ] The “database” can be nothing more than intelligent storage. Data could be stored generically and rendered accordingly by the client

34. Please rank the following two statements in order of your level of agreement with them (rank = 1 means you agree with it more than rank = 2)

- [ ] Managing multiple schemas from independent sources and interrelationships between them, i.e., “schemachaos” is inevitable and unavoidable
- [ ] Any imposition of schema can be done by the clients, as and when needed by applications.

## Optional Integration Story

35. Optional - Tell an integration horror story. What when wrong and why in your opinion. Please note we are specifically interested in scenarios which caused significant delay. (Hint, best story will be awarded a prize)

# C

# Appendix C – CORBA (OMG)

The 'Common Object Request Broker Architecture' (CORBA) is a standard defined by the Object Management Group (OMG) and provides users with a language and platform-neutral remote procedure call (RPC) specification.

There is a lot to like about what CORBA achieves, as indicated in Table C1, when addressing Hasselbring's heterogeneity and autonomy problem dimensions as it elegantly handles the mapping of objects and provides interfaces and controlled functionality across language boundaries to the outside world.

| Dimension | Requirement | Rating | Dimension | Requirement | Rating |
|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | No | *Distribution* | Scalable distribution | No |
| | Business intelligence | Partial | | Proxy services | Yes |
| | Intelligent nodes | No | | Endpoint agnostic | No |
| | Complex event processing | Yes | | Service discovery | Yes |
| | Service composition | Yes | | Service reuse | Yes |
| | Asset wrapping | Yes | *Architecture* | Reliability | Yes |
| | Virtualization | Yes | | Availability | No |
| | Model-driven implementation | Yes | | Security | No |
| | Balanced autonomy | No | | Portability | No |
| *Heterogeneity* | Model agnostic | No | | Functionality / extensibility | Yes |
| | Solution agnostic | No | | Interoperability | Partial |
| | Structures | Yes | | Usability | No |
| | Supported standards | Yes | | Maintainability | No |
| | Wrappers | Yes | | Efficiency | No |
| | Service | Yes | | Testability | Yes |
| | | | | Reusability | Yes |
| | Messaging | Yes | | Ease of deployment | No |
| | Message control | Yes | | Ease of administration | No |
| | Message transformation | Yes | | Scalability | No |
| | Message security | Yes | | | |
| | Message monitoring | Yes | | | |

*Table 29: CORBA Assessment Table*

Unfortunately, CORBA suffers from a number of critical failings as an integration platform, and additional barriers that prevent it from scaling indefinitely (Henning, 2008).

Autonomy

Highly complex entity definitions result in significant code changes when models are altered. When the ORB is forced to change, the integration solution is broken on both sides, so even if the providing party resolved the mapping on its side, the subscribing party might still be broken. A global integration

solution must have a middle transformation format that is resilient to change and tolerant of restructuring, including full support for change management.

Heterogeneity

CORBA provides no support for change management and therefore does not scale well. In recent years we have seen the pace of standards and technologies increase, and a global integration platform would only exacerbate this situation. The platform — in accordance with Hasselbring's heterogeneity problem dimension — must be capable of supporting change in a number of ways including change management, and a model change should be data driven, avoiding any redeployment of integration technology.

CORBA's API set is large and complex. As we know from Hasselbring's research, this spells immediate disaster for integration, as lightweight simple standards have proven more resilient to change; whereas the opposite, heavily complex interfaces are prone to change and thus cause integration to break regularly, requiring additional development and maintenance.

CORBA's implementation of interface definitions is extremely verbose, often requiring more lines of code for description than the underlying functional implementation itself and complicating interactions with the CORBA runtime.

Architecture

Because CORBA's mappings are tied to the C++ language, they are difficult to maintain and suffer from the language's inherent limitations. It seems peculiar that an integration platform, which tried valiantly to introduce a neutral model structure, also bound itself to technology platforms so tightly as illustrated in Figure C1. Today the adoption of more modern technologies that encourage loose coupling makes integration infinitely easier to maintain.

CORBA's configuration is considered so complex that it requires third-party tooling to use it. A platform developed for mass integration must have very simple, hot-configurable administration, which is not bound to any underlying technology and which requires little training.

Additionally a global platform must not alienate any technology platform. It must embrace interoperability in a simple and elegant way.

CORBA has a number of security flaws which are unacceptable in today's operating environment. Importantly, the traffic is unencrypted, which under a locked-down local network may be acceptable in some cases — but in a wide-area-network environment, encrypted traffic is mandatory.

Additionally, because CORBA uses custom ports in many cases for communication, firewall access to the integration server is problematic.

Due to dynamic CORBA encoding with the absence of compression, and heaviness in protocol implementation, CORBA's performance is poor and thus will not scale well at high volumes.

Portability is a challenge due to the extensive support for types in language-specific implementations. As certain types are not necessarily supported by all languages, portability can become an issue.

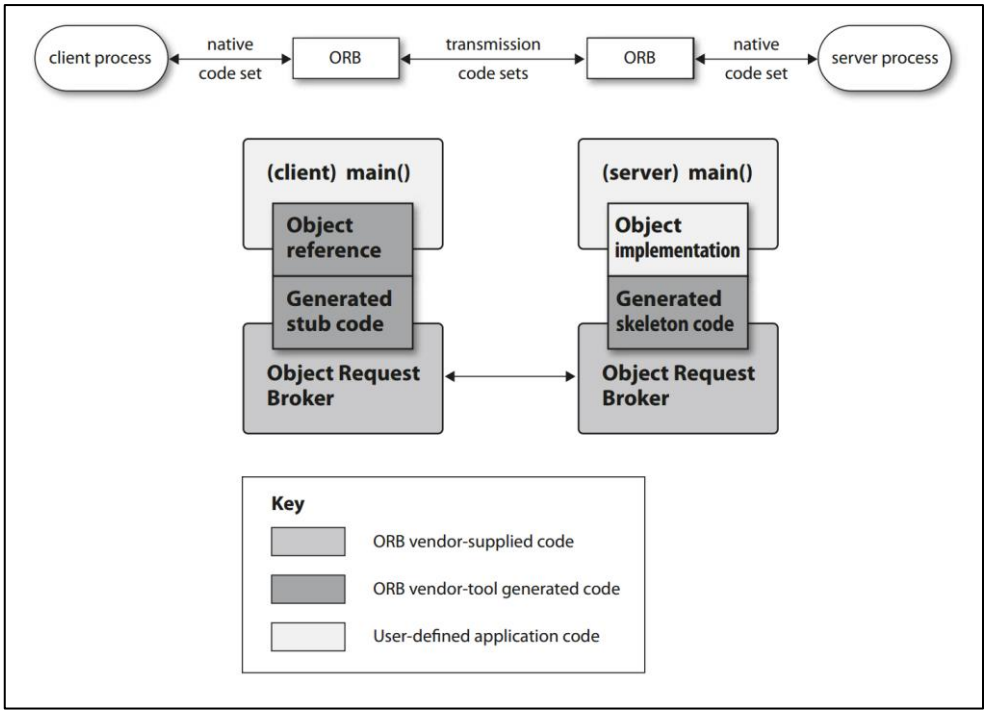



*Figure 29: CORBA Request Structure*

Distribution

CORBA does not provide progress towards the distribution problem dimension because to implement a two-way CORBA solution seamlessly, both components involved in the integration scenario must consume CORBA interfaces. In a cloud-based environment, it cannot be guaranteed that a solution will be willing to implement a CORBA integration solution.

CORBA does not offer full support for asynchronous messaging. In periods of high loading/high volumes this is a critical failing as asynchronous processing is required for reliability and availability.

Also, CORBA requires language mappings to operate successfully. Currently, many modern technologies such as C# and .Net are unsupported.

Ultimately CORBA is too tightly coupled with certain technologies to be considered as a global integration platform. While it might work within a LAN environment, for a WAN operation it tries to exert too much control over integration instead of positioning itself as a technology-neutral piece of middleware between integrating platforms. Consequently, the proprietary structure of the platform acts as a repellent and discourages organizations from building integration logic within it.

In keeping with our objective of creating an information superhighway, CORBA falls short. However, it offers some significant contributions when constructing our integration platform, as discussed later.

# D

# Appendix D – Mapping Platform

Felix Van de Maele's mapping platform is insightful and helps to analyse and break down the usefulness of different forms of integration.

| Dimension | Requirement | Rating | Dimension | Requirement | Rating |
|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | Yes | *Distribution* | Scalable distribution | No |
| | Business intelligence | No | | Proxy services | No |
| | Intelligent nodes | No | | Endpoint agnostic | No |
| | Complex event processing | Yes | | Service discovery | No |
| | Service composition | Yes | | Service reuse | No |
| | Asset wrapping | Yes | *Architecture* | Reliability | Yes |
| | Virtualization | Yes | | Availability | No |
| | Model-driven implementation | Yes | | Security | No |
| | Balanced autonomy | No | | Portability | No |
| *Heterogeneity* | Model agnostic | Yes | | Functionality / extensibility | Yes |
| | Solution agnostic | Yes | | Interoperability | Partial |
| | Structures | Yes | | Usability | No |
| | Supported standards | Yes | | Maintainability | Yes |
| | Wrappers | Yes | | Efficiency | Yes |
| | Service | Yes | | Testability | Partial |
| | | | | Reusability | Yes |
| | Messaging | Yes | | Ease of deployment | No |
| | Message control | Yes | | Ease of administration | No |
| | Message transformation | Yes | | Scalability | No |
| | Message security | Yes | | | |
| | Message monitoring | Yes | | | |

*Table 30: Van de Maele Assessment Table*

He highlights that semantic integration is normally a process that outputs a set of mappings based on two input heterogeneous models and therefore provides a means to transform from one object to the other. However, he continues in his argument to assert that the mapping process would be more valuable if the mappings could be shared in a communal way, thus providing an environment where evolution of objects and their mappings could occur.

Results in Table D1 show that there are some similarities with OMG's platform. Like the model-driven architecture paradigm established by OMG, Van de Maele reiterates the importance of transforming via a platform-independent intermediary model — a critical requirement for a globally accessible integration platform. Both Van de Maele and OMG argue that the usefulness of the platform-independent model (PIM) is in its ability to be transported in its platform-neutral form, so it may be transformed elsewhere to suit its new environment.

Unlike CORBA, which translates from platform-neutral to platform-specific implementations, Van de Maele's mapping platform uses mappings which are language-, application- and paradigm-neutral end

to end, as illustrated in Figure D1. The true value of reaching this level of neutrality, when combined with a centralized cloud-based mapping server, is in establishing a foundation for a common ontology where both service consumers and service providers will be required to transform to and from the model.

As discussed in depth by the focus groups and with support from the survey data, we can surmise that the industry is in need of a common ontology, but the motivation to create/adopt one is lacking. Van de Maele raises the possibility that there could not only be a centralized integration platform but also mappings could be shared and applied according to context in a community-based manner — in line with social media movements which increase the motivation to establish and extend a centralized ontology.

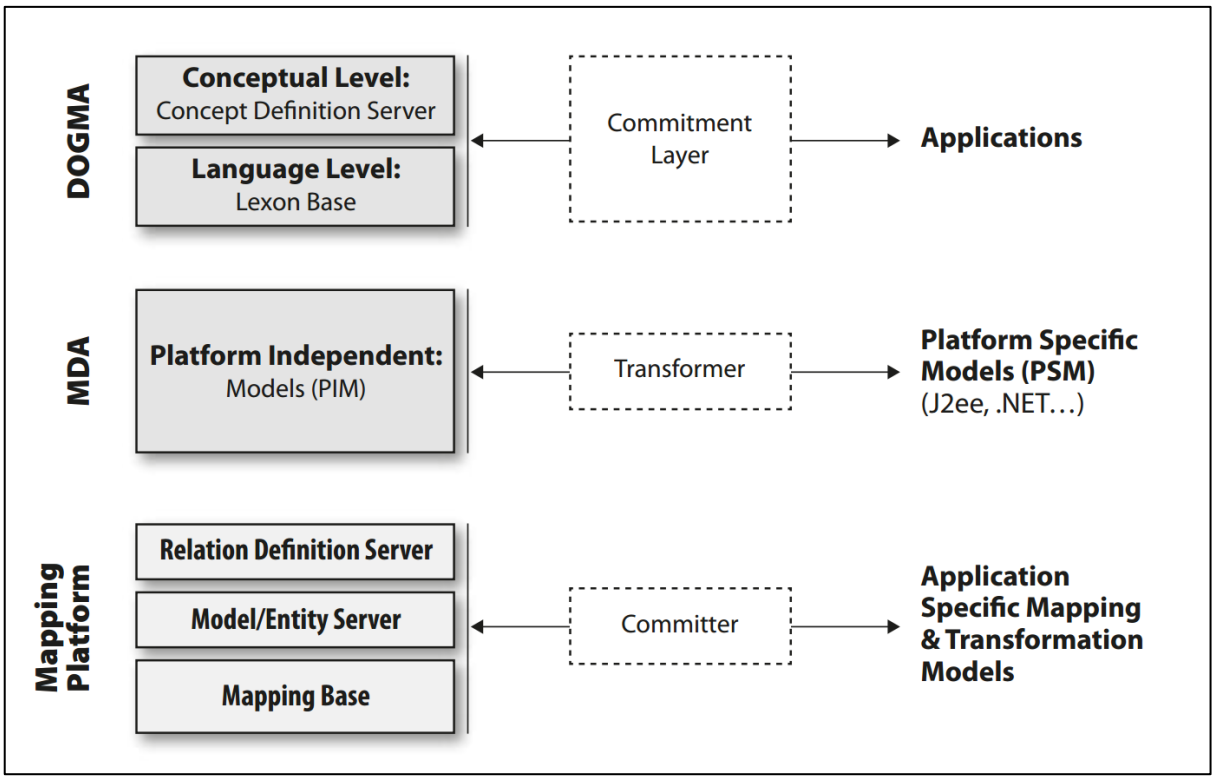



*Figure 30: Comparison Between DOGMA, OMG and Van de Maele's Semantic Integration Mapping Platform*

Autonomy

The fundamental limitation when considering the Van de Maele platform in the context of global integration is that autonomy is poorly handled. The VDM platform does a superb job of ontology mapping, but access to the mapping and therefore autonomy of platform execution lies squarely on the client side of the ledger. It makes sense why this approach has been taken — it tasks the system that understands the mapping technology with applying the mapping. However, in doing so the opportunity to orchestrate data flow centrally is lost with each node in the network being responsible for routing its own traffic in a point-to-point style.

With this structure, mapping is served through a service layer, capable of being load balanced and scaled in a stateless manner, and responding to requests from the client. The lack of intelligent routing prevents events from passing through business intelligence layers and being broadcast to a wider audience in a 'digital nervous system' style.

Heterogeneity

On the flip side, the VDM platform has excellent support for models and transformation between them. The removal of platform-specific bindings is a major improvement on CORBA and consequently Hasselbring's heterogeneity problem dimension is managed effectively.

Distribution

Distribution is not handled well by the VDM platform. Although there is some sharing of transformation mappings, there is no method for publishing and subscribing of data — forcing the network to operate in a polling-type structure. To operate as an autonomous mesh, capable of distributing data objects throughout, each node would need to be aware of connected nodes and know if they were interested in the particular data object in question — client/server structures do not support this type of approach. In a mesh node, endpoints should be known and loosely coupled enabling distributing rules (routes) to determine if a data object should travel a specific network path.

Architecture

Van de Maele addresses scalability by discussing both horizontal and vertical scaling options. However, he also acknowledges that the approach will not scale endlessly and the limits are quickly reached. Neither VDM nor CORBA is designed specifically for the cloud and therefore they have not addressed mesh mechanics, which would coordinate and distribute load efficiently. This would need to be dealt with in a global integration platform.

Although many of the architectural features have been addressed, a number of them also fail to operate at scale. The mapping platform may be easy to deploy in isolation but not when thousands of instances are required to generate a super network. Likewise, development tools have been addressed that help with modelling entities and mappings. However, there is no consideration given to applying these tools across many millions of data providers with billions of data models — the eclipse approach is not usable at scale.

# E

# Appendix E – Web Services

The World Wide Web Consortium (W3C) is an international community that develops open standards for the World Wide Web, which help to promote its growth and evolution. The W3C has a vision for the web and provides many standards that constitute best-practice. However, the consortium is powerless to enforce these standards and thus proprietary standards often compete with and derail the W3C vision.

In the web services space, W3C describes a web service as an agent which implements a defined set of functionality and executes that functionality by sending a response message as the result of a request message. The functional definition of what the service can provide is published using a standard called the Web Services Description Language (WSDL); however, the method of implementation remains hidden.

Table E1 highlights the areas where web services perform well, specifically in areas of scalability. It should be noted, however, that the table reflects a best-case scenario. Web services are scalable depending on the caching policy they employ (or lack thereof).

| Dimension | Requirement | Supports | Dimension | Requirement | Supports |
|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | Partial | *Distribution* | Scalable distribution | No |
| | Business intelligence | No | | Proxy services | Yes |
| | Intelligent nodes | No | | Endpoint agnostic | No |
| | Complex event processing | Yes | | Service discovery | Yes |
| | Service composition | Yes | | Service reuse | Yes |
| | Asset wrapping | Yes | *Architecture* | Reliability | Partial |
| | Virtualization | Yes | | Availability | Yes |
| | Model-driven implementation | Yes | | Security | Yes |
| *Heterogeneity* | Balanced autonomy | Yes | | Portability | Yes |
| | Model agnostic | No | | Functionality / extensibility | Yes |
| | Solution agnostic | No | | Interoperability | Yes |
| | Structures | Yes | | Usability | Yes |
| | Standards | Yes | | Maintainability | Yes |
| | Wrappers | Yes | | Efficiency | Yes |
| | Service | Yes | | Testability | Yes |
| | | | | Reusability | Yes |
| | Messaging | Yes | | Ease of deployment | Yes |
| | Message control | Yes | | Ease of administration | Yes |
| | Message transformation | Yes | | Scalability | Yes |
| | Message security | Yes | | | |
| | Message monitoring | Yes | | | |

*Table 31: Web Services Assessment Table*

The WSDL provides information on message structure, datatypes used by the message and any transport-related configuration that might be necessary when processing the payload. In summary, the service description is a binding agreement to which the client and server must conform for the client to make a request to the agent.

At first glance it would appear the more structure a web service can build, the more robust the service will become. By increasing the semantics used to define behaviour of the interaction, we can more explicitly determine what a service will do when we call it.

As descriptiveness increases, the chance that multiple service providers implementing a similar service will deliver a greatly different interface to the outside world will also increase, which may make switching to an alternative service provider more difficult.

For example, consider the following ways of ordering a pizza:

1. OrderPizza("Margherita", 11 Some Place, 022 2222222);

versus

2. OrderFood("Margherita Pizza", Bob Jones, Takapuna, 08:00pm);

By necessity, definitions such as these require a change in code, and with a service description of hundreds to thousands of method calls, this would not be a simple process.

Conversely, as the web service becomes more descriptive, services are more likely to be automatically consumed by third-party services that can read the description and take appropriate integration approaches. A balance must therefore be struck, and context should be considered when determining the descriptive nature of the web service interaction.

Distribution

As Hasselbring identified, simple non-changing interfaces and payloads are the way forward when establishing a core infrastructure service such as FTP, SMTP, POP and, potentially, the global integration platform. For that reason, the integration platform should have a representational state transfer interface (REST), which would allow a payload to be posted in raw format to the integration platform.

REST interfaces have been around for some time, but have recently become popular as a simple method for requesting data from a server. In turn, this promotes a greater degree of distribution as subscribers are able to integrate more easily.

Even with a REST interface, a web service still adheres to a client/server approach, which forces each node to know about every node it integrates with. This approach follows a point-to-point pattern, does not scale well, and describes the current state of play with cloud-based web services.

Autonomy

Worse still, in addition to distribution challenges with a web services structure, autonomy is once again left on the fringe, as each node also needs to understand the business intelligence executed behind each integrated endpoint and when to execute it.

While web services are not necessarily dumb, and may be a façade for a more intelligent service or set of services, distributed autonomy remains a challenge due to the number of layers that would break given a structural change. This can be addressed, in part, by using versioned interfaces, but also requires tight coupling between provider and subscriber, and both parties need to be aligned.

Heterogeneity

Similarly, when using a REST interface, models are not defined using service definitions as they are using SOAP, for example. Therefore the subscriber must either be informed of the structure via a third-party service, or must realize it through some other means. A better approach would be to have an intermediary which brokers the model transformations, enabling a subscribing platform to post its own model (or communal model) to the intermediary, rather than consuming the subscriber model.

This approach simplifies integration, moves all mapping and transformation considerations for all integration into a centralized location and, like the VDM platform, motivates the community to share data models and ontologies.

Architecture

Despite being a force in the cloud space, web services alone suffer the same fate as the VDM when attempting to scale. Web servers (with the exception of federated networks) have not addressed mesh mechanics to scale infinitely. The primary reason is that most web servers are stateless or cache static content only and have little need to know the state of another node in their cluster.

To distribute load effectively in a global integration network, each node would need to have clear responsibilities and know which other nodes it must partner with to broadcast events or receive traffic.

Despite these limitations, it should be acknowledged that, in general, web services are easy to scale, administer and use, and form the backbone of today's World Wide Web.

# F

# Appendix F – Apache NiFi

Review – Apache NiFi and the HPCS

The NiFi platform is described by its makers as an easy-to-use, powerful, and reliable system to process and distribute data. After investigating the product and reviewing the findings in Table F1, it is fair to say this summary accurately reflects the product's capabilities.

Distribution

The NiFi platform from Apache combines integration theory with modern-day technologies and standards. NiFi's contribution to the integration space centres on a dataflow approach where data is input, processed and output by the NiFi engine. Because of the componentized approach NiFi takes, it is possible to daisy chain multiple NiFi servers (or even clusters) together to form a high-volume integration network.

Autonomy

At the centre of NiFi is its expression language, a platform-independent approach to business intelligence that can execute against data payloads as they pass through the NiFi engine. This is a significant advantage over CORBA and VDM, which mostly transform without the opportunity to centralize intelligence. The result is a hub-and-spoke solution as opposed to the point-to-point solution employed by CORBA and VDM.

Heterogeneity

NiFi supports a REST interface offering a simple way to inject data payloads into the engine. Dataflow is data driven so the NiFi interface and payload remain simple — satisfying Hasselbring's heterogeneity concerns — and the platform remains solution, model and endpoint agnostic, meaning that changes outside of NiFi will not require NiFi development and redeployment.

| Dimension | Requirement | Rating | Dimension | Requirement | Rating |
|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | Yes | *Distribution* | Scalable distribution | No |
| | Business intelligence | Yes | | Proxy services | Yes |
| | Intelligent nodes | No | | Endpoint agnostic | Yes |
| | Complex event processing | Yes | | Service discovery | Yes |
| | Service composition | Yes | | Service reuse | Yes |
| | Asset wrapping | Yes | *Architecture* | Reliability | Yes |
| | Virtualization | Yes | | Availability | Yes |
| | Model-driven implementation | Yes | | Security | No |
| | Balanced autonomy | No | | Portability | Yes |
| *Heterogeneity* | Model agnostic | Yes | | Functionality / extensibility | Yes |
| | Solution agnostic | Yes | | Interoperability | Yes |
| | Structures | Yes | | Usability | Yes |
| | Supported standards | Yes | | Maintainability | Yes |
| | Wrappers | Yes | | Efficiency | Yes |

| | | | | | |
|---|---|---|---|---|---|
| | Service | Yes | | Testability | Yes |
| | | | | Reusability | Yes |
| | Messaging | Yes | | Ease of deployment | Yes |
| | Message control | Yes | | Ease of administration | Yes |
| | Message transformation | Yes | | Scalability | No |
| | Message security | Yes | | | |
| | Message monitoring | Yes | | | |

*Table 32: NiFi Assessment Table*

Architecture

The biggest problem with the previously reviewed platforms is scalability. For both CORBA and VDM, although many of the architectural considerations were addressed, the solutions failed at scale. NiFi is considerably better in this regard due to its centralized autonomy, which can be distributed amongst multiple server nodes.

A clustered formation allows a single endpoint to support many NiFi servers, and clusters can be daisy chained together using an appropriate connection configuration. Unfortunately, there is a known limitation with this approach. Inputs that are generated server side cannot currently be distributed within the cluster set automatically and instead must be manually configured on a single node within the cluster. Apache acknowledges this is a significant limitation and is working to distribute the internal tasks automatically.

In addition, however, there is no automatic control of NiFi clusters to interoperate with each other in a cluster-of-clusters approach (or mesh of clusters). Although this structure can be manually simulated there is no automated control, which would be required should NiFi be adopted as a global integration platform.

Deployment and administration of the NiFi platform are excellent with graphical user interface tools controlling the design of the dataflow process components and process groups, including support for templates, which can be shared. Again, however, a lack of scalability is evident. There is no centralized ACL or security layer capable of interoperating with multiple NiFi clusters. As such, the potential for a greater community to drive the platform forward is reduced.

# G

# Appendix G – Amazon (AWS) Stack

The Amazon stack (AWS) is a collection of services that form a cloud-computing platform. By combining virtualization technology to host virtual servers and web services with core Internet services such as domain name management and email management, Amazon's AWS provides a full stack that can be used to deploy complete business systems in the cloud.

One of the services Amazon provides is Lambda, which executes code in response to events. As we have previously discussed, the need to execute business intelligence is a critical requirement for a global integration platform, and Table G1 highlights the performance of the AWS stack + Lambda in doing so.

| Dimension | Requirement | Rating | Dimension | Requirement | Rating |
|---|---|---|---|---|---|
| *Autonomy* | Resilient to organizational change | Yes | *Distribution* | Scalable distribution | No |
| | Business intelligence | Yes | | Proxy services | No |
| | Intelligent nodes | No | | Endpoint agnostic | No |
| | Complex event processing | Yes | | Service discovery | No |
| | Service composition | Yes | | Service reuse | Yes |
| | Asset wrapping | Yes | *Architecture* | Reliability | Yes |
| | Virtualization | Yes | | Availability | Yes |
| | Model-driven implementation | Yes | | Security | No |
| | Balanced autonomy | No | | Portability | Yes |
| *Heterogeneity* | Model agnostic | Yes | | Functionality / extensibility | Yes |
| | Solution agnostic | Yes | | Interoperability | Yes |
| | Structures | Yes | | Usability | Yes |
| | Supported standards | Yes | | Maintainability | Yes |
| | Wrappers | Yes | | Efficiency | Yes |
| | Service | Yes | | Testability | Yes |
| | Messaging | Yes | | Reusability | Yes |
| | Message control | Yes | | Ease of deployment | Yes |
| | Message transformation | Yes | | Ease of administration | Yes |
| | Message security | Yes | | Scalability | No |
| | Message monitoring | Yes | | | |

*Table 33: Amazon Stack Assessment Table*

Autonomy

Two considerations to autonomy need to be discussed. Firstly, the control of scaling virtual instances is very good technically. However, a centralized control point is still lacking, which would manage requests according to their content. Load balancing blindly shares the load, whereas in the context of a global integration platform, control of content distribution would have to be much smarter.

The second consideration is the powerfulness of the Amazon Lambda execution. Lambda was originally Amazon's defence against container technology, enabling business intelligence to be executed in the

cloud. Any deficiency in this critical piece would render the stack untenable for global integration. Fortunately, as Lambda is based on Node.js and therefore provides functional development support, it is more than up to the task. Lambda is able to combine this support with functionality from the Amazon stack including persistence and resource management.

Heterogeneity

Similar to the platforms discussed above, AWS is capable of serving web services including REST interfaces. As a technology stack, providing infrastructure, it is up to the developer what integration platform is implemented on top. What is interesting, however, is that the integration interface can be significantly lighter because business intelligence is handled within Lambda. Predominantly, the web services can be used as an entry point for the inbound messages, which are passed to Lambda.

Distribution

Again, as Amazon itself is a technology stack, it is difficult to penalize the platform in terms of distribution. An argument can be made that endpoint and service management can be data driven out of a connected AWS data store, but it is not enough. The fact is, considerable technology needs to be engineered to manage these concerns, but it is not currently available.

Architecture

The architecture for Amazon lends itself well to providing a foundation for a global integration platform. Currently the structure of AWS allows for accounts, which own and manage Amazon services; but in the case of a global integration platform, services must be managed by the public. It may, again, appear unfair to label Amazon unscalable, but it is important we continue to acknowledge the operating context. Amazon does not provide (out-of-the-box) a way for members of the public to administer Lambda rules against a single Lambda network with degrees of access control and the ability to distribute those rules appropriately within a network of Lambda nodes.

Amazon has been designed to support many large fault-tolerant deployments, rather than a mega-deployment — which would be required for a global integration platform.