Upgrading Unupgradable Middleware Legacy Processes: Misconceptions, Challenges, and a Roadmap

Radhouane B. N. Jrad[1], M. Daud Ahmed[2] and David Sundaram[3]

[1] Carter Holt Harvey, Auckland, New Zealand

[2] Manukau Institute of Technology, Auckland, New Zealand

[3]University of Auckland, Auckland, New Zealand

Abstract. Middleware systems are information systems which live in the middle of business processes, enabling communication between various businesses and their heterogeneous information systems. When it comes to upgrading their legacy processes, middleware systems have their own particular requirements and complexities that standard upgrade roadmaps are incapable of fully addressing. In particular, legacy processes that cannot be upgraded using vendor or market tools due to the lack of knowledge about their mechanisms or due to their complexity represent a challenge to upgrade projects. This paper proposes a roadmap to address this challenge by offering a side-by-side approach to migrate unupgradable legacy processes in middleware systems with minimum business and financial impact.

1 Introduction

Enterprise Messaging Systems (EMS), commonly known as Middleware Systems or Middleware Applications, are sets of published standards and tools that allow architecturally heterogeneous computer systems to exchange information through usage of sector-endorsed structured messages (e.g. XML) and protocols (e.g. SOAP with web services) [10]. This intra-systems communication mechanism is key to establishing Business-to-Business (B2B) electronic trades [14] and can be thought of as the "postman" and the "translator" between various information systems of business partners that cannot communicate directly. Middleware systems have also been used to establish technological relationships between non-business systems such as between a web server and the cloud [14, 17].

A middleware system is part of the business process spectrum, i.e. it does not only play the role of bridging the technological differences between business partners, but its processes are part of the business logic itself. For instance, if a Business Partner 1sends a Purchase Order (PO) to a Business Partner 2 (step 1 in Fig.1 below), the middleware may need to map various fields from the purchase order into meaningful fields for the target systems. This requires understanding of business-process-related logics such as which fields represent the total PO value, delivery address, etc. and what conditions do they need to fulfill (e.g. a total must not be negative). Although

B. N. Jrad R., Daud Ahmed M. and Sundaram D..
Upgrading Unupgradable Middleware Legacy Processes: Misconceptions, Challenges, and a Roadmap.
DOI: 10.5220/0004602201110118
In *Proceedings of the 1st International Workshop in Software Evolution and Modernization* (SEM-2013), pages 111-118
ISBN: 978-989-8565-66-2

the business logic might reside outside the middleware flow, it still needs to be assimilated for the conversion to occur; and it is not uncommon to use the middleware as a validation tool in order to avoid lengthy traffic and delays between end-systems. The middleware system would also be required to inform systems and/or stakeholders of any delivery problems, and relay messages, such as acknowledgements, between end-systems. Fig.1 below illustrates an example on how middleware systems are implicated in a B2B process between 3 business partners: a Customer (Partner 1), a Seller (Partner 2), and a Deliverer (partner 3). The figure illustrates both the complexity that middleware systems add to the entire design but also the level of complexity taken out of end-systems (referred to in Fig.1 as Internal Business Processes) allowing them to focus on their core processes.

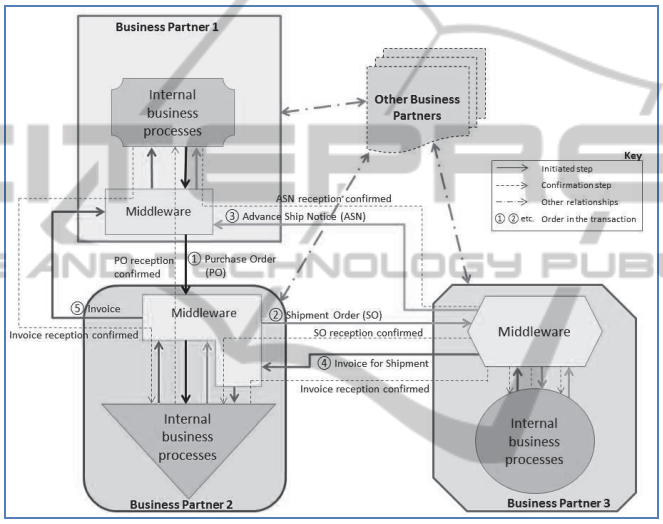

Fig. 1. Middleware as B2B enabler.

Middleware systems also facilitate communication between two or more distributed information systems, without directly being part of a B2B process [17]. This is achieved through bridging the gap of systems heterogeneity and distribution [7]. In this case, the middleware layer sits between applications and lower layers, such as operating systems, network, hardware, etc. [10, 15]. Most notably, middleware systems are used with the cloud technology where companies might either opt for the classic middleware framework, and/or in line with the Software-as-a-Service (SaaS),the Platform-as-a-Service (PaaS), and the integration-as-a-service (IAS) concepts [14]. Complexity of upgrade for these systems is more likely to be associated with the complexity of the systems themselves [8, 18], be it a genuine complexity or due to bad design, than about the obsolesce of the As-Is systems.

It is possible to utilize middleware system both as B2B and technology enablers. The scope of this paper, however, is specific to middleware systems for B2B relationships, and it addresses complex legacy processes. By legacy systems we reflect on “techniques, tools, and processes” that may have once been state-of-the-art but now are nonresponsive to changes that are happening in the business processes or the

technological environment [19]. These legacy systems usually are so outdated that knowledge about their internal processes and characteristics (e.g. contexts in which they were created or modified) are lost with the loss of people who built or customized them. And while some of these legacy systems can be upgraded using standard or specific vendor tools, a good number of them require either full reengineering, or end up being partially or totally left untouched; and in the process engaging the rest of the process to deal with them as black boxes with only inputs and outputs as understandable components. We refer to these black box processes as 'Unupgradable Legacy Processes' because they are considered as such in the scope of the middleware systems upgrade projects. We refer to the act of treating them as black box as a "Wrapping" based on Callaway's definition [2] because usually more code is added around them to get them to work in the new systems.

Businesses expect to use one middleware solution for their entire needs. These one-stop solutions need to be scalable, standardised and capable of integrating with existing technologies, including legacy systems [13]. B2B processes have been around for tens of years and middleware process enabling them have also been around long enough [1] for waves of designers, architects, analysts, and developers who contributed to tailor the excessively standard initial installation to the company's need [11] to retire or move on to different roles and companies leaving behind processes that have now become legacy and unupgradable. And while the simple middleware legacy processes are easy and cheap to reengineer, unupgradable complex legacy systems represent the biggest challenge in middleware systems upgrade projects.

2 Misconceptions about Managing Changes in EMS

Because the middleware environment does not generate income and is usually hidden from business stakeholders, it is common to resist upgrading the middleware systems because "they work fine", unless there is a drive to manage/reduce their costs [6]. From another angle, there is also an implicit assumption that middleware systems are just sub-systems, fitting under the general requirements and frameworks for enterprise systems upgrades. This is apparent in the generalisation of risk analysis in research about implementation of enterprise systems (e.g. Scott & Vessey [16]). This misconception ignores unique characteristics of middleware systems and particular risks and impact associated with their management and upgrade project, which all makes their management significantly different from other Information systems. The difference includes the management of:

Requirements: Middleware systems need to respond to requirements from different information systems and processes that may not use the same definitions or standards for their processes. This makes their design and architecture a challenging exercise.

System: When the middleware system is undergoing an upgrade there is the assumption that B2B end-systems are not affected, with minimum or no downtime, despite the fact that middleware systems outages directly affect the B2B process even if end-systems are up and running. Therefore, middleware systems usually are attributed higher sensitivity, risk, and impact than other B2B information systems in the same

organization, and when one B2B system is down, other B2B systems see the error at the middleware system's level due to its positioning between all the B2B systems.

Life-Cycle: As new business processes and information systems come into the foray, middleware systems need to both accommodate the legacy information systems, as well as the new technologies and standards. During its life, a middleware system would receive many updates until a point where a revamp is required. Due to the high risk and impact associated with them, managing the lifecycle of a middleware system is a consistent challenge both technologically and from business perspective.

Upgrading a middleware system is a unique exercise in IT and requires addressing its challenges in a more specific context than within the broad theories about upgrading information systems. In particular, addressing unupgradable legacy processes within middleware systems requires a particular set of concepts and theories in order to properly face the associated challenges.

3 Challenges of EMS Upgrades – Legacy Processes

Scott and Vessey [16] presented a risk factor model for implementing enterprise systems in general, and Davenport [3] highlighted general challenges associated with upgrading these enterprise systems. While such models and challenges do apply to middleware systems in B2B processes, they have more associated risks and challenges than what these model offer, and there are multiple ways to illustrate this difference. One of the most visible and critical situations is when the middleware system is down. Downtime is when the system is witnessing a partial or total outage and is unable to operate adequately [4]. Let's consider a scenario where two information flows are required going from Enterprise A to Enterprise B (Fig.2). The flows are as follows:

- Flow 1: Process A1→ Middleware1 → Middleware 2 → Process B1
- Flow 2: Process A2→ Middleware1 → Middleware 2 → Process B2

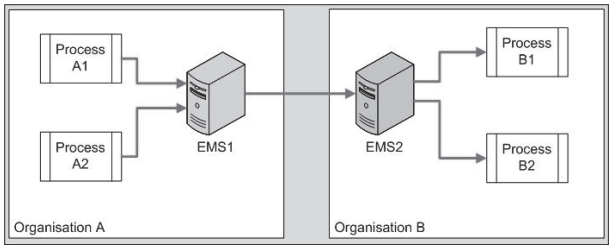

Fig. 2. A simplified representation of B2B information exchange.

If for any reason Process A1, and only process A1, stops, Flow 2 can continue to perform while EMS1 is sending alerts (e.g. emails or paging messages) about A1's unavailability or inaccessibility. However, an outage of EMS1 would disrupt information flow completely between the two enterprises, and processes A1 and A2 would either throw errors or create backlogs while EMS2 would also throw connectivity errors due to non-responsiveness of EMS1. In this simple scenario, middleware out-

age has greater impact than the outage of a B2B end-system. Because of such risks, practitioners and users of middleware systems often face serious challenges to upgrade legacy processes into latest state-of-the-art-technology middleware products. New systems and processes usually adhere to current best practices, frameworks, philosophies, and paradigms. The upgrade becomes complex since middleware systems contain interdependent unupgradable legacy processes (Fig.3).

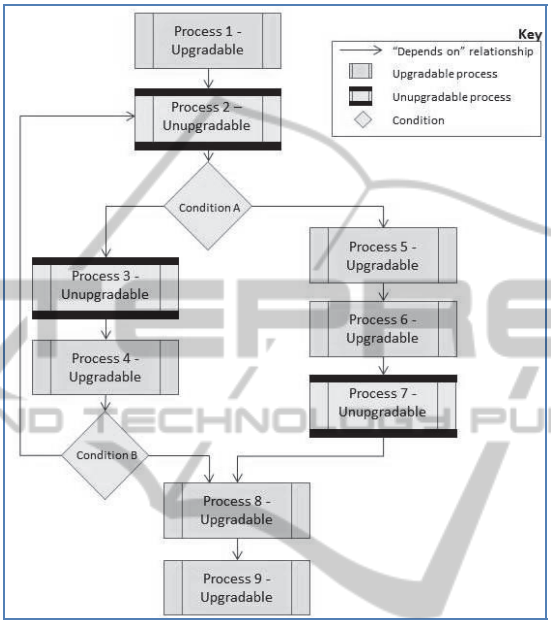

Fig. 3. Interdependence of unupgradable legacy middleware processes.

While an unupgradable process is in itself a challenge, having unupgradable processes referring to each other increases the level of complexity. Fig.3 illustrates how dependency between unupgradable processes can be bidirectional. This is possible when one service's code calls on another service if a certain condition is met. The loop allows the code to flow in any direction between various services, effectively creating a complex dependency net. This complexity is further deepened if these processes belong to different interfaces. Interfaces are sub-mechanisms of the information system that have common logic to be grouped together, and usually represent either a B2B process specific to a partner or a common central processes (e.g. an email interface). The lack of knowledge about some legacy processes in these interfaces puts them in the heart of the upgrade challenges. Interdependency of unupgradable legacy processes implies that migrating one process also requires concurrent migration of all other related unupgradable legacy processes, which may in turn have dependency on other unupgradable legacy processes. The snowball effect can result in a large number of legacy interfaces requiring the upgrade, all at once. This is a Mild or Big Bang approach [3]. While a Big Bang approach does have advantages, it brings time and budgetary constraints to the business and has high risk of business disruption [12]. One commonly used method to avoid the Big Bang approach is to

wrap existing unupgradable legacy processes. Wrapping a process consists of keeping it like a black box and adding header processes to manipulate the outputs to fit the new paradigms. Not only does such an approach present risks due to unknowns associated with it, it also presents the challenge of only delaying the inevitable requirement for reengineering. During subsequent upgrades, wrapped processes become too complex and too inefficient to be double/triple/n-tuple wrapped and the knowledge of what's inside the black box fades with time, making it an even harder exercise to reengineer it.

4 Roadmap to Upgrading Unupgradable Legacy Processes in EMS

To address the issue of upgrading complex unupgradable legacy processes in middleware systems, we propose a roadmap that is invasive to the existing process [9] and based upon the side-by-side approach that ensures business continuity but also a softer upgrade process[20]. It is indeed intentional to step away from Mild and Big Bang approaches due to the inherent risks and costs associated with them [12], while acknowledging that reengineering all unupgradable legacy systems might not be possible due to budget and time constraints. As Fig.4 shows, the proposed roadmap can be combined with various known practices in the market, including wrapping processes where/if necessary for transition phases.

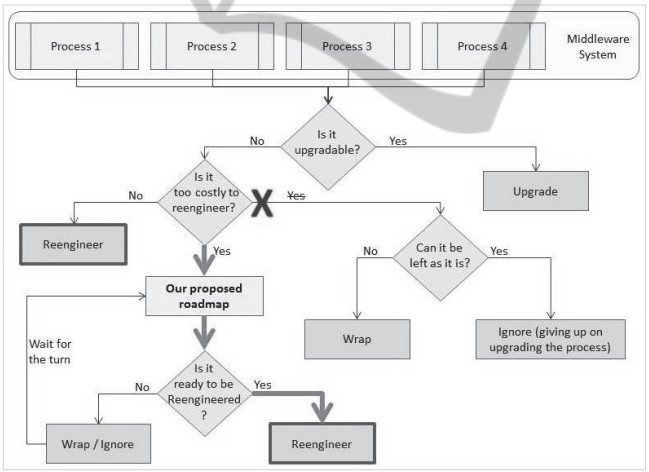

Fig. 4. Decision flow for upgrading unupgradable legacy processes.

Fig. 5 shows how an existing process ("S2 →B2 → B3 → B4 →A4→S4) can be upgraded to process ("S2→C1→Standard Process→S4") without affecting other processes and flows. In the existing process, both interfaces A and B use custom (i.e. not vendor-provided) code to process transactions, while in the new process the new interface C transforms data in a format that the new Interface D can handle in a standard way. Interface D offers the same standard process for all interfaces of similar nature, such as converting all Purchase Orders from different partners into one canon-

ical format before sending it to the organisation's B2B end-system. While interfaces A and B remain available, the two new interfaces (C and D) cater for the newly upgraded process. The step-by-step approach would gradually move processes into the new interfaces until a point where all processes have shifted to the new paradigm, and the old interfaces can be decommissioned –it is also possible to decommission interface as they get migrated to the new system. This approach is agnostic to the details in the existing processes, which minimizes downtime for the process and the entire middleware system, and reduces the burden of handling the black box by eliminating the need for tracking known dependencies or discovering hidden dependencies [5].

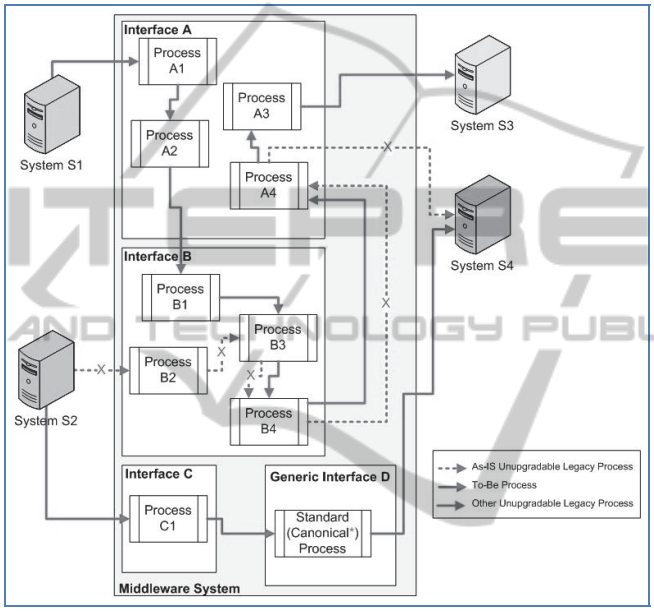

Fig. 5. Side-by-side upgrade of an unupgradable legacy process in a middleware system.

5 Conclusions

Middleware systems for B2B are information systems which by essence live in the middle of business processes, enabling seamless communication between various business partners and their heterogeneous information systems. Upgrading unupgradable legacy processes in middleware systems is a challenging exercise that standard upgrade procedures are not sufficient to deal with. Although some of these processes can be reengineered, complex processes are usually treated like a black box and wrapped towards the new standards with focus on inputs and output. Such exercise becomes more tedious as subsequent upgrade projects are run. The fact that these unupgradable processes may be interdependent adds yet another level of complexity. This paper proposes a roadmap to address these challenges by offering a side-by-side timely-controlled migration for unupgradable legacy processes in EMS with minimum business and financial impact. This roadmap has been conceptualized from the

synthesis of academia and practices of a highly specialized middleware integration team in a large corporate. Though the roadmap has been used in a number of projects and platforms, it is still in its incipient stage, and requires further refining and enhancement.

References

1. Bye, P.: What is middleware and where is it going?, in S. Nilsen "A CORBA service for the OSA+ Real-Time Middleware". (2003).
2. Callaway, E.: Enterprise resource planning: Integrating applications and business processes across the enterprise. Computer Technology Research Corporation, Charleston, South Carolina (1999).
3. Davenport, T.H.: Mission critical: realizing the promise of enterprise systems. Harvard Business School Press, Boston (2000).
4. Dhillon, B.: Engineering Maintainability: How to Design for Reliability and Easy Maintenance. Gulf Pub. Co., Houston, Texas (1999).
5. Dumitraş, T. et al.: No more Hot Dependencies: toward dependency-agnostic online upgrades in distributed systems. a Carnegie Mellon University Pittsburgh publication. (2007).
6. Dutta, K., VanderMeer, D.: Cost-based decision-making in middleware virtualization environments. European Journal of Operational Research. 210, 2, 344–357 (2011).
7. Hasselbring, W.: Information system integration. Communications of the ACM. 43, 6, 32–38 (2000).
8. Herbert, L. et al.: SaaS Adoption 2010 : Buyers See More Options But Must Balance TCO , Security , And Integration. www.forrester.com. June, 1–7 (2010).
9. Jingyong, L. et al.: Middleware-based Distributed Systems Software Process. International Journal of Advanced Science and Technology. December, 13, 27–48 (2009).
10. Kajan, E., Stoimenov, L.: Toward an ontology-driven architectural framework for B2B. Communications of the ACM. 48, 12, 60–66 (2005).
11. Krishna, A. et al.: Context-specific middleware specialization techniques for optimizing software product-line architectures. ACM SIGOPS Operating …. (2006).
12. Markus, M., Tanis, C.: The enterprise systems experience–from adoption to success. In: Zmud, R.W. (ed.) Framing the domains of IT management: projecting the future... through the past. pp. 173–207 Cincinnati, OH: Pinnaflex Education Resources Inc. (2000).
13. Mehling, H.: B2B Takes the Next Step Forward -- New Internet middleware products bring together key e-business technologies. VARbusiness. October, 117 (2000).
14. Mitchell, R.L.: Wrapped in Complexity, (2009).
15. Pinus, H.: Middleware: Past and present a comparison. University of Maryland Baltimore County publication. October, 1–5 (2004).
16. Scott, J., Vessey, I.: Managing risks in enterprise systems implementations. Communications of the ACM. 45, 4, 74–81 (2002).
17. Taiani, F. et al.: What is middleware made of?: exploring abstractions, concepts, and class names in modern middleware. The 11th Workshop on Adaptive and Reflective Middleware1th Workshop on Adaptive and Reflective Middleware. (2012).
18. Wailgum, T.: SaaS's Troubled Adolescence: Three Signs of Immaturity, http://www.cio.com/article/595654/SaaS_s_Troubled_Adolescence_Three_Signs_of_Immaturity.
19. Warren, I., Avallone, D.: The Renaissance of Legacy Systems. Springer (1999).
20. Welti, N.: Successful SAP R/3 implementation: Practical management of ERP projects. Addison-Wesley Longman Publishing Co., Boston (1999).