Preface

We have embarked on a exciting era of developing enterprise applications based on assembling business components into electronic business processes that interact via the Internet. This contemporary paradigm for developing enterprise applications is rapidly gaining momentum in the business community. Enterprise applications built using earlier software-development paradigms focused almost exclusively on internal business processes. However, enterprise applications built using this emerging development paradigm are progressively encoding cross-organizational business processes. Coupled with the Internet, business component computing is believed to be able to create a class of enterprise applications that is flexible and relatively easy to maintain, upgrade, and integrate. This up-and-coming breed of enterprise applications brings a realm of tantalizing possibilities for highly competitive business models, for instance, integrated virtual supply chains that treat business software components as business processes, and vice versa.

While trying to move to this arena, many brick-and-mortar companies are hindered by their heritage of stovepipe[1] enterprise information systems, which are typically large, heterogeneous, and mission-critical systems. Unfortunately, it is commonplace that these inherited enterprise systems are strongly intertwined with existing business processes and policies. Although this allows efficient operations, it severely hinders their alignment with novel, integrated business processes. The replacement and renovation of these systems, most of which have become legacy systems with the passage of time, has proved to be a complicated, time-consuming, expensive, and risky endeavor (Brodie and Stonebraker 1995). These systems are tightly coupled to the business workflow and suffer from a rigid and contrived architecture that results from many years of ad hoc patching and fixing, and that offers very limited openness to other systems.

Given this situation, organizations are facing an unenviable dilemma. On the one hand, they want to get rid of their legacy systems as soon as possible in order to be able to align their enterprise systems with new processes. On the

other hand, legacy applications and repository systems are considered important enterprise assets, harboring significant investments in enterprise data and policies already at hand. Business processes often critically depend on the permanent availability of these legacy systems. Hence, one cannot annihilate the investments and simply replace legacy systems with new business applications.

Motivation for This Book

Since the client/server days of the 1980s, several legacy evolution strategies have been developed to overcome the legacy dilemma. The four most prominent ones are encapsulation/integration, data warehousing, "Cold Turkey" (replace at once, or rip and replace), and the gradual migration approach (Umar 1997; Brodie and Stonebraker 1995; Wilkes 1999).

The encapsulation/integration strategy, sometimes referred to as the access/integration in-place strategy, is the strategy of choice when legacy systems hold valuable and up-to-date assets that should be leveraged in the next generation of enterprise information systems. This strategy is empowered by contemporary distributed software component technologies allowing legacy systems to be componentized.

In particular, component wrappers, also called wrappers or legacy components, are used to encapsulate legacy systems in an interface layer exposing internal legacy programs, transactions, and data to external software components (Comella-Dorda et al. 2000). In this way, component wrappers enable sharing and conservation of the legacy systems and at the same time allow them to be aligned with new business process requirements.

Despite the fact that existing legacy wrapper approaches and tools have been applied successfully in integrating existing legacy applications and repositories with new application components, they suffer from several major drawbacks:

- They are oriented toward legacy systems, but not business processes. In today's practice, most component-wrapping approaches exploit programming logic and data that are encapsulated in legacy enterprise systems, largely neglecting requirements imposed by new business processes in the business domain. These approaches are typically oriented toward realizing technical integration between new applications and legacy systems with virtually no regard for the new business process requirements that the legacy systems are supposed to facilitate.
- Legacy code and data cannot be reused as is. Existing component-wrapping solutions are typically based on the assumption that new processes and applications may be developed in terms of legacy systems with minor adjustments

to the legacy data and functionality to make them accessible to external resources. This assumption is not realistic because legacy systems are geriatric systems that implement partly outdated business processes and policies. In addition, their large-grained nature make them inappropriate building blocks for composition into consistent and cohesive business processes.

- Existing component-wrappting approaches usually seek to integrate legacy systems with new business applications in a purely bottom-up manner without taking into consideration alignment with new top-down engineered business process requirements.
- Finally, and most important, any effective approach to aligning new business processes with componentized legacy systems must be methodology-oriented. A symmetric, component-based alignment methodology that combines bottom-up reverse engineering of legacy systems according to the encapsulation/integration strategy with top-down design of business processes is required.

Goals of This Book

The alignment of business processes with information technology has often been studied from a rather abstract vantage point that is believed to provide insufficient guidance to architects and designers of business processes and applications (Henderson and Venkatraman 1993; van Eck, Blanken, and Wieringa 2004). It is the main aim of the methodological framework presented in this book to offer a body of pragmatic methods and techniques to assist in aligning business processes and legacy systems. This methodological framework resulted from an extensive review of the scientific literature, and it has been tested in various experiments and realistic case studies, which are drawn from a large real-world project at the Dutch DoD.

The methodological framework addresses the drawbacks of many contemporary techniques. In particular, it is designed to combine bottom-up reverse engineering approaches with top-down business process engineering methods, introducing a meet-in-the-middle approach that is capable of aligning new business processes with legacy systems. This alignment is facilitated by a sophisticated process of comparison and adaptation to ensure the processes and systems (continue to) fit.

Hence, alignment starts with a thorough understanding of the relationship between business requirements and available information technology capabilities. One important aspect of alignment is to identify gaps and matches, after which new software may need to be constructed to fill gaps and to integrate

with reusable information technology resources. Both issues are addressed in the methodological framework.

Ulrich (2002) discusses management-related issues in the alignment of legacy systems with new business processes, which concern governance and communications across line-of-business organizations and information technology organizations. The communication and governance techniques discussed by Ulrich may be used to embed the methodological framework into an organization.

Another approach that is related to the methodological framework is outlined by Seacord, Plakosh, and Lewis (2003). This entails a risk-managed modernization approach, including activities such as defining a business case, understanding the legacy system and target technology, devising a deployment strategy, and working toward a modernization plan that helps to mitigate risks during the actual modernization activities. This approach is complementary to the methodological framework of this book in that it may assist in those activities that take place *before* the actual modernization and alignment of legacy systems and business processes.

The global architecture underlying the methodological framework is depicted in figure 1. It is made up of three building blocks: reverse engineering, forward engineering, and alignment.

Reverse engineering is concerned with gaining an understanding of legacy programs and repositories, and defining component wrappers that encapsulate them into a software layer so that they look and behave like components. Wrappers resulting from this phase are depicted as parallelograms in the bottom tier of figure 1. Forward engineering derives a set of business components from the requirements of new business processes. These components are depicted as chevrons in the top tier of the figure.

Alignment of to-be business processes and as-is legacy systems implies that legacy components that have significant overlap with new business requirements (e.g., required services) are mapped to new business components, signifying that these business components reuse the functionality of the legacy components. Remaining legacy components can be ignored.

In figure 1, mappings between business and legacy components are illustrated with dotted lines connecting components in the top and bottom tiers. The mappings often include some adaptation to make the interfaces of legacy components and business components compatible, for instance, overcoming a semantic interoperability problem.

In the figure, harmonization of processes with systems is realized by aligning four out of six required business components with preexisting legacy components (dark gray chevrons in alignment layer). Business components that cannot

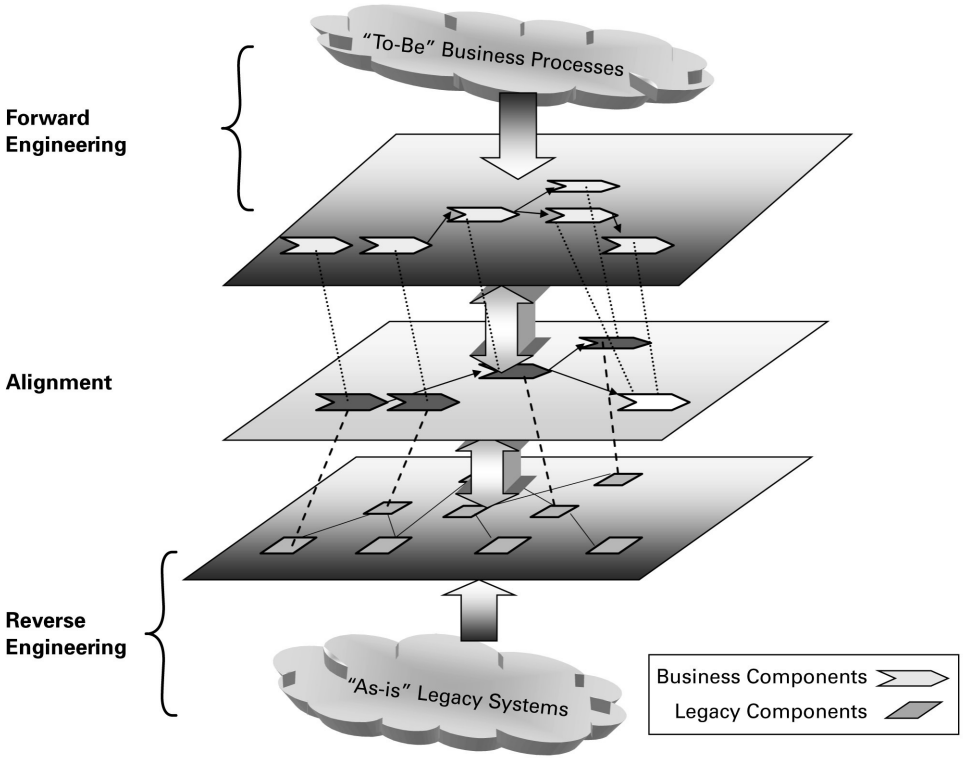

Figure 1
Alignment methodology adopts a meet-in-the-middle paradigm.

be aligned with legacy systems may alternatively be supported by packaged software solutions or commercial off-the-shelf (COTS) components. For example, the white chevron in the alignment layer represents two business processes that were fused to allow implementation by a packaged software solution. In this case, alignment was achieved by changing the two business processes to meet the reality of the services provided by the packaged solution.

Organization of This Book

This book covers a wide range of multidisciplinary topics from the fields of software engineering, component-based development, reverse engineering, information retrieval, and business process modeling.

Chapters 1–3 establish the theoretical foundations of the methodological framework. Chapter 1 introduces component based development in general

and business component development in particular. Chapter 2 elaborates the rationale of the framework, surveying existing legacy evolution strategies and technologies. It seeks to familiarize readers with legacy components, and it introduces a case study that is used throughout the book as a running example to illustrate and explore the methodological framework. Chapter 3 assesses integration problems for single-enterprise or cross-enterprise business processes and explains how integration is used to achieve alignment within the framework.

Chapters 4–6 describe and develop the methodological framework. The main workings of the framework are illuminated in chapter 4. The next two chapters concentrate on two critical phases: matching enterprise models of to-be processes with models of as-is legacy wrappers (chapter 5) and adapting business components and legacy components to align them (chapter 6).

Chapter 7 summarizes the main findings and remaining open issues regarding the methodological framework, and outlines a research agenda for the future.