Pro .NET 2.0 Code and Design Standards in C#

Mark Horner

Apress®

Pro .NET 2.0 Code and Design Standards in C#

ISBN (pbk): 1-59059-560-2

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Lead Editor: Ewan Buckingham
Technical Reviewer: Jon Reid
Editorial Board: Steve Anglin, Dan Appleman, Ewan Buckingham, Gary Cornell, Tony Davis,
Jason Gilmore, Jonathan Hassell, Chris Mills, Dominic Shakeshaft, Jim Sumser
Associate Publisher: Grace Wong
Project Manager: Beckie Brand
Copy Edit Manager: Nicole LeClerc
Copy Editors: Freelance Editorial Services and Ami Knox
Assistant Production Director: Kari Brooks-Copony
Production Editor: Ellie Fountain
Compositor: Molly Sharp
Proofreader: Linda Seifert
Indexer: Toma Mulligan
Artist: Kinetic Publishing Services, LLC
Interior Designer: Van Winkle Design Group
Cover Designer: Kurt Krames
Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit http://www.springeronline.com.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit http://www.apress.com.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at http://www.apress.com in the Source Code section.

CHAPTER 5

Design Policy

Chapter 1 contained a discussion of code policy, which is used to manage the important aspects of *code.* This chapter examines policy as it relates to *design* and identifies how *architecture* is used to manage the enterprise.

What

A design policy is a plan that is used to identify and manage the important aspects of architecture—it expresses how an enterprise will be configured to support the technical and functional objectives implicit in an organization's business strategy.

Where

A design policy is applied across the enterprise, and it may extend to stakeholders.

Why

A policy is an effective and efficient way to coordinate the management of a range of interrelated architectures (e.g., enterprise, application, data, and network architectures), each with different dynamics.

How

A design policy, like a code policy, doesn't have a definitive structure; it is developed around how best to coordinate the enterprise, which will vary on a case-by-case basis. A structure may be developed around a set of design objectives and use a design style to implement them. Figure 5-1 illustrates a design policy that licenses the open-source architecture framework (TOGAF) standard and uses a set of in-house standards that complement the architecture framework.

Note TOGAF is a methodology that uses an architectural framework to manage a set of architectures across an enterprise. TOGAF is open-source technology that is available from The Open Group (`www.opengroup.org`) through a public licensing arrangement. Members of The Open Group include IBM, HP, SAP, Intel, OMG, Apple Computer, Oracle, Computer Associates International, Sun Microsystems, NASA, the U.S. Department of Defense, and Citigroup.

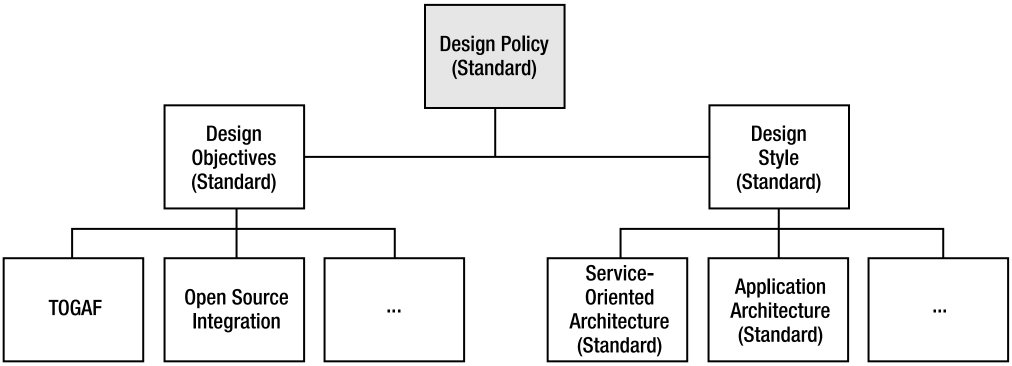
Design Policy (Standard)
Design Objectives (Standard)
Design Style (Standard)
TOGAF
Open Source Integration
...
Service-Oriented Architecture (Standard)
Application Architecture (Standard)
...

Figure 5-1. *Design policy structure*

The Standard: Design Policy

> *The standard acknowledges that a design policy is an effective and efficient way to coordinate the management of a range of interrelated architectures that have different dynamics.*

Design Objectives

A design policy relies on a set of objectives to express how it is implemented across the enterprise.

What

Design objectives identify what is necessary to satisfy the requirements of the design policy.

Where

Design objectives form part of the design policy.

Why

Objectives are important because they express how a policy statement will be realized in concrete terms.

How

Each policy statement is considered in terms of architecture—what needs to be done to fulfill a given policy statement. For example, a policy statement might say: “The enterprise is to be designed to maximize loose coupling.” From that statement one considers the implications and prepares a set of objectives, such as the following:

- Objective # 020: Implement a service-oriented architecture (enterprise services)
- Objective # 030: Integrate applications through an application integration layer
- Objective # . . .: etc.

The Standard: Design Objectives

> *The standard acknowledges that design objectives are an essential part of managing the design of an enterprise because they identify in concrete terms what needs to be done.*

Design Style

This section discusses *design style,* which refers to the arrangement of architecture.

What

A design style describes the use of an architecture framework and architectures (e.g., data and network architectures) to style an enterprise.

Where

A design style is used across the enterprise, and it may extend to stakeholders.

Why

A design style coordinates architecture decision making, across all architectures, to ensure that the enterprise is and remains structured in a way that complements an organization's (dynamic) business strategy.

How

Developing a design style involves several key steps: (1) choose an architecture framework (e.g., TOGAF)—a methodology to manage a set of architectures; (2) identify a target architecture—how the enterprise will look in 12 to 24 months; (3) prepare a gap analysis—what needs to be done to get from the present situation to the target or destination architecture; (4) develop a roadmap; (5) develop a migration plan; and (6) implement the roadmap.

The Standard: Design Style

> *The standard acknowledges the use of design style, which seeks to ensure that the enterprise is and remains structured in a way that complements an organization's (dynamic) business strategy.*

Architecture Framework

An enterprise is managed through architecture. An architecture framework orchestrates a set of architectures that manage the specialist technical areas of an enterprise (e.g., software, data, or network development).

What

An architecture framework is a tool that defines and strategically aligns specialist architectures (e.g., network, application, integration, and data architectures). A small set of architecture framework tools are recognized as international standards, including Zachman, C4ISR/DoDAF, FEAF, and TOGAF.

Where

An architecture framework sits across the enterprise.

Why

An enterprise needs to be strategically managed, in line with a business strategy, and an architecture framework is a tool that can be used to regulate the enterprise consistent with the technical and functional requirements implicit in a business strategy.

How

An architecture framework is selected from a set of recognized international standards and implemented. Implementation includes defining a target architecture, undertaking a gap analysis, preparing an architecture roadmap, and applying an architecture development methodology.

The Standard: Architecture Framework

> *The standard acknowledges the use of an architecture framework to strategically use architecture to manage an enterprise.*

Target Architecture

The framework manages architectural change through the definition of its target architecture, which expresses the future structure of the enterprise.

What

A target architecture is the architecture that an organization wants to have at a future point in time, to support a business strategy.

Where

A target architecture is part of the architecture framework.

Why

A design policy has to accommodate the changes in a business strategy that impact functional and technical objectives, and this often means that an architecture needs to adapt. The target architecture encapsulates and coordinates change as a version or iteration of the enterprise and avoids the high risk associated with implementing structural change in an impulsive or ad hoc manner.

How

The architecture framework defines the target architecture and manages the impact of technical and functional change across an enterprise.

The Standard: Target Architecture

> *The standard acknowledges that the target architecture encapsulates and coordinates change as a version or iteration of the enterprise and avoids the high risk associated with implementing structural change in an impulsive or ad hoc manner.*

Architecture Roadmap

To move an existing architecture to a target architecture, the technical team follows an architectural roadmap.

What

An architectural roadmap identifies the *when, what,* and *how* of an existing architecture being migrated to a target architecture.

Where

An architecture roadmap is part of the architecture framework.

Why

A roadmap acts as a guide by which tasks can be readily identified and sequenced.

How

An enterprise architecture or team of architects, in consultation with technical staff (e.g., application developers and network and data administrators), defines a roadmap in which tasks are identified, sequenced to be progressively iterated, as milestones, to eventually arrive at the target architecture.

The Standard: Architecture Roadmap

> *The standard acknowledges the use of an architecture roadmap as a guide to migrating an existing architecture to a target architecture.*

Architecture

An architecture is a structure that organizes artifacts into a cohesive system.

What

An architecture is defined as "the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution" (ANSI/IEEE Std 1471–2000).

Where

An architecture resides within an architecture framework. Common types of architectures presented as standards are as follows:

- Enterprise architecture—defines the technology structure that represents the organization; it may cross stakeholder domains.
- Network architecture—defines the structure of a network of servers within an enterprise.
- Technical architecture—defines middleware software.
- Application architecture—defines application design, development, and integration.
- Data architecture—defines the structure of physical and logical data; how it is stored, accessed, and distributed within the enterprise.
- Deployment architecture—defines the deployment of applications on network nodes within the enterprise, or it may cross stakeholder domains (e.g., remoting).
- Integration architecture—describes how applications that are internal and external of the enterprise share functionality and data.
- Service-oriented architecture—defines a type of integration architecture premised on loose coupling, which distributes functionality and data as published services.
- Business architecture—defines business and technical objectives, and IT governance and business processes.

Why

An architecture is an efficient and effective way to manage the artifacts of an enterprise, as it wraps complexity by presenting a simple interface through which activity may be coordinated.

How

An architecture is prepared in the context of its role (e.g., a service-oriented architecture has an integration-services role) within an architecture framework, which is determined by the technical and functional objectives of a business strategy.

The Standard: Architecture

> *The standard acknowledges the use of architecture to manage enterprise artifacts in line with technical and functional objectives.*

Enterprise Architecture

Defines the structure that represents the enterprise, which may cross stakeholder domains.

What

An enterprise architecture is an architecture that coordinates technology, across the enterprise, in line with a given business strategy.

Where

An enterprise architecture resides within an architecture framework and spans an organization or across organizations.

Why

Technology is complex, particularly when it expands not only an organization's domain but across stakeholder domains. An enterprise architecture is an efficient and effective way to manage technology in line with a business strategy.

How

An enterprise architecture is prepared in the context of its role within an architecture framework, which is determined by the technical and functional objectives of a business strategy.

The Standard: Enterprise Architecture

> *The standard acknowledges the use of enterprise architecture as an efficient and effective way to manage technology, across an enterprise, in line with a business strategy.*

Network Architecture

A network architecture is an artifact of an enterprise architecture.

What

Network architecture defines the structure of a network of servers within an enterprise.

Where

A network architecture resides within an architecture framework and spans an organization or across organizations.

Why

A network of servers needs to be structured and managed in a manner that supports the software, technology, data, and deployment architecture, and this can be done through architecture.

How

A network architecture is designed and maintained on the basis of the support requirements of software, technology, data, and deployment architectures.

The Standard: Network Architecture

> *The standard acknowledges the use of network architecture as a tool to support the strategic requirements of software, technology, data, and deployment architectures, within the context of an architecture framework.*

Technical Architecture

A technical architecture is an artifact of an enterprise architecture.

What

A technical architecture comprises middleware software, which is software that performs the role of an intermediary between two discrete artifacts. Commonly the Model–View–Controller (MVC) design pattern is used as an *architecture design pattern* to implement decoupling between architectures. MVC is discussed in Chapter 9.

Where

A technical architecture resides within an architecture framework and spans an organization or across organizations.

Why

The use of middleware promotes loose coupling and minimizes the risk of duplicating functionality, by abstracting the functionality into a central service to which artifacts may subscribe.

How

Functionality is published as a service and not duplicated and coupled to an application's implementation. For example: rather than write similar data functionality in many Web pages that couple the pages directly to the interface of a database, the functionality is abstracted to data service middleware, which acts as an intermediary between the pages and the data source. In the event of change to the interface of the database, the maintenance is confined to the relationship between the middleware and the database, leaving the interface exposed by the middleware to its clients (the Web pages) unchanged. Generally, it costs less to maintain middleware than multiple clients.

The Standard: Technical Architecture

> *The standard acknowledges the use of technical architecture as an efficient and effective way to maximize decoupling and minimize duplication, within the context of an architecture framework.*

Application Architecture

A software or application architecture is an artifact of an enterprise architecture.

What

An application architecture is a blueprint for application design, development, and integration.

Where

An application architecture resides within an architecture framework and spans an enterprise.

Why

An application architecture ensures applications are designed, developed, and integrated consistently, which minimizes the risk of nonconformity and exposure to additional costs of maintenance and reengineering.

How

Application architectures are commonly designed as application frameworks or templates from which applications are developed as implementations of the framework or template. The framework contains prebuilt common enterprise functionality, which reduces development to adding specific application functionality and specializing enterprise functionality, as required.

The Standard: Software or Application Architecture

> *The standard acknowledges the use of application architecture as an efficient and effective way to manage application design and development within the context of an architecture framework.*

Data Architecture

Data architecture is an artifact of an enterprise architecture.

What

Data architecture defines the structure of physical and logical data and how it is stored, accessed, and distributed within and across domains (e.g., nontransactional data is denormalized and stored in a warehouse [OLAP], and it may be published as Web services).

Where

A data architecture resides within an architecture framework and spans an enterprise or across enterprises.

Why

A data architecture is an efficient and effective way to manage and protect the investment in data and information.

How

Data is modeled physically and logically and stored in repositories (e.g., database and warehouse), from where it is distributed in a timely and efficient manner to clients.

The Standard: Data Architecture

> *The standard acknowledges the use of data architecture as an efficient and effective way to support information requirements within the context of an architecture framework.*

Deployment Architecture

A deployment architecture is an artifact of an enterprise architecture.

What

Deployment architecture defines the deployment of applications or functionality and data on network nodes. For example, although it may be logical to deploy functionality and data to a given server, owing to high transaction volumes it may be more efficient (e.g., responsive) to replicate or split functionality and data across multiple servers to better balance peak workloads.

Where

A deployment architecture resides within an architecture framework; it spans a domain network or may cross stakeholder domains.

Why

A deployment architecture plays an important role in defining distribution and accessibility to enterprise functionality and data.

How

A deployment architecture is designed to complement the requirements of an enterprise architecture by deploying functionality in an optimal manner that promotes efficiency, reliability, and security.

The Standard: Deployment Architecture

> *The standard acknowledges the use of deployment architecture as an efficient and effective way to deploy functionality and data throughout an enterprise within the context of an architecture framework.*

Integration Architecture

Integration architecture is a methodology by which artifacts within a domain or across domains are integrated.

What

Integration architecture describes how applications share functionality and data. Commonly, there are two types of integration architecture: synchronous and asynchronous. Hub-n-spoke (including Web services) is synchronous, and integration is tightly coupled to an interface. A message bus is an example of asynchronous integration that is loosely coupled (the consumer of the service does not know the source of the functionality—it only has to know the message protocol and which queue to send the message to).

Where

An integration architecture resides within an architecture framework; it spans a domain network or may cross stakeholder domains.

Why

Integration architecture is critical because it defines how application functionality and data are moved within and across domains. The choice of integration architecture (synchronous and asynchronous) may significantly impact the efficiency and effectiveness of an enterprise to meet the requirements of a business strategy.

How

Asynchronous integration acts through an interim layer or middleware that decouples the client from the provider. By contrast, with hub-n-spoke integration the client integrates directly with the provider. Commonly the Model–View–Controller (MVC) design pattern is used as an *architecture design pattern* to implement decoupling between architectures. MVC is discussed in Chapter 9.

The Standard: Integration Architecture

> *The standard acknowledges the use of integration architecture as an efficient and effective way to manage the integration of functionality and data throughout an enterprise within the context of an architecture framework.*

Service-Oriented Architecture

Service-oriented architecture (SOA) is premised on providing or publishing functionality or data as a service against which clients subscribe.

What

SOA is synonymous with Enterprise Application Integration (EAI) and is a type of integration architecture that is loosely coupled and through middleware supports asynchronous and synchronous communication between publisher and subscriber.

Where

SOA resides within an architecture framework; it spans a domain network or may cross stakeholder domains.

Why

SOA offers an enterprise the opportunity to integrate using either asynchronous or synchronous communication. In domains where there are large transaction volumes or complex transactions, SOA's ability to support asynchronous communication has a lower risk profile than synchronous transaction support.

How

SOA may be designed as follows: hub-n-spoke, Web service, or message bus. The hub-n-spoke model (Figure 5-2) is the traditional synchronous model used to share functionality, where the hub is a server and the entities are client applications. Although functionality is reused, access is limited to a LAN/WAN architecture. The entities (clients) are coupled to the interface of the server (hub); using a hub in peak times may impact performance.

A Web service model (Figure 5-3) is a variation of the hub-n-spoke model, as it, too, is a synchronous model; however, it is not restrained by LAN/WAN, as it leverages Internet technology to distribute services. The entities (clients) are coupled to the interface of the Web service.

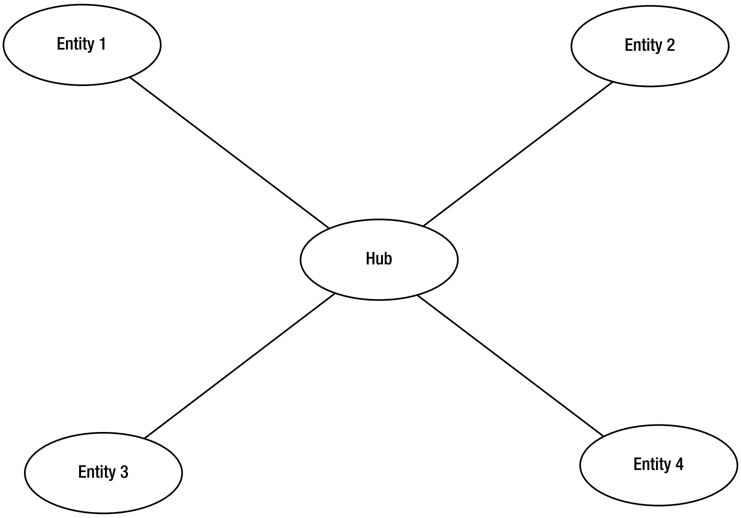

Figure 5-2. *Hub-n-spoke*

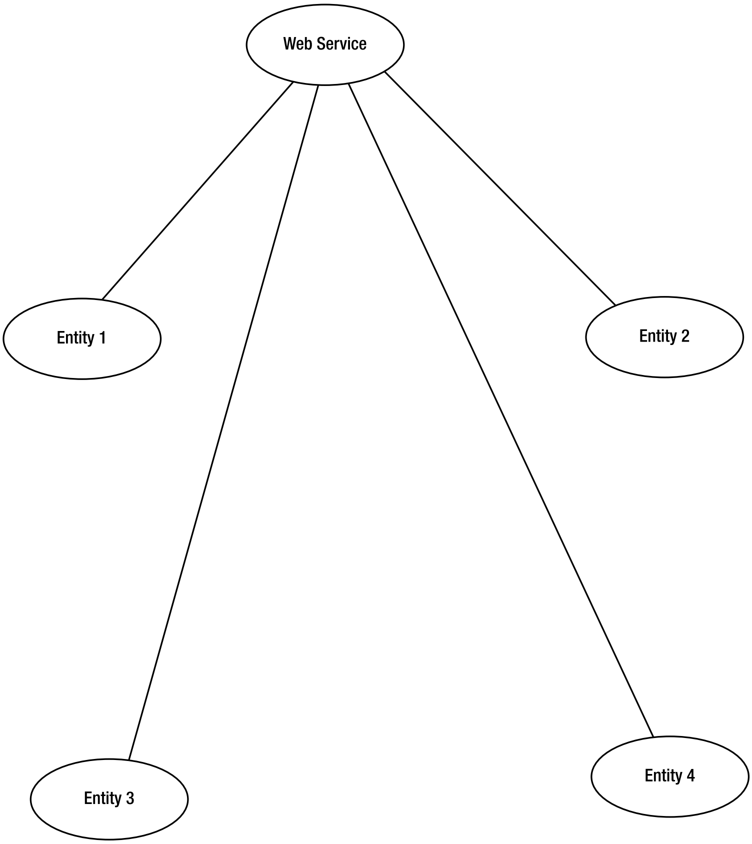

Figure 5-3. *Web service*

The message bus (Figure 5-4) is an asynchronous model which leverages queues and messages, where functionality is accessed by sending and receiving messages via a given message queue. The entities (clients) are not coupled to an interface—they do not know the source of the functionality, they only know about a message format and a queue.

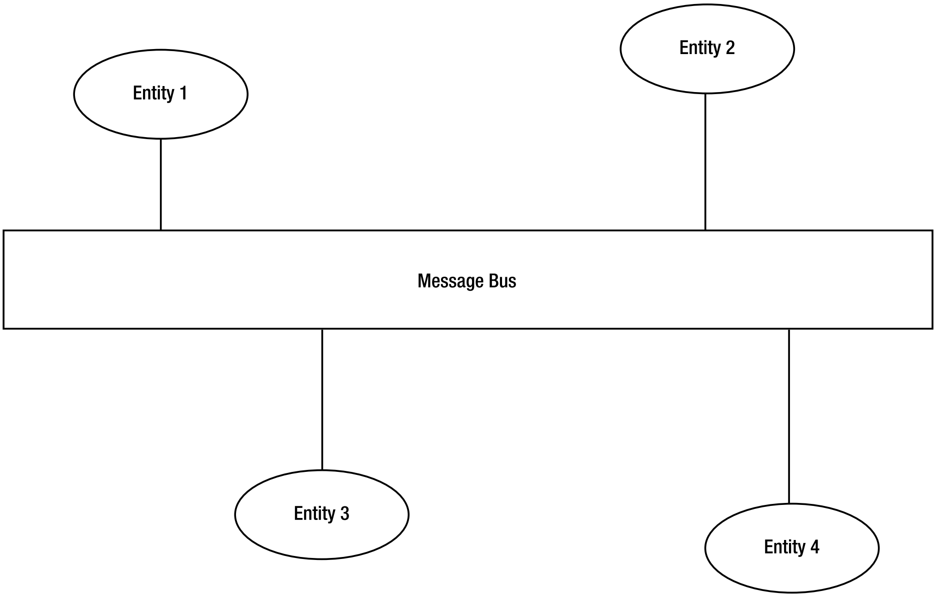

Figure 5-4. *Message bus*

The Standard: Service-Oriented Architecture

> *The standard acknowledges the use of service-oriented architecture as an efficient and effective way to manage the publication and subscription of functionality and data throughout an enterprise within the context of an architecture framework.*

Business Architecture

The business architecture identifies business and technical objectives, and defines IT governance and business processes.

What

A business architecture describes the business and technical artifacts that are expressed through a hierarchy of architectures.

Where

A business architecture resides within an architecture framework.

Why

A business architecture identifies the strategic purpose that unites and coordinates the set of architectures used to manage and safeguard the enterprise.

How

A business architecture is developed by a software development committee, an architecture committee, and an IT governance committee. The software development and architecture committees are responsible for supervising the implementation and management of the business architecture; however, they are accountable to the IT governance committee for its performance.

The Standard: Business Architecture

> *The standard acknowledges the use of business architecture to identify a strategic purpose to unite and coordinate a set of architectures to manage and safeguard the enterprise.*