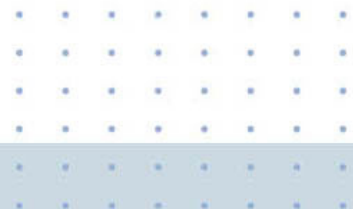


Microsoft®

Describing the Enterprise Architectural Space



patterns & practices

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2004 Microsoft Corporation. All rights reserved.

Microsoft, MS-DOS, Windows, Windows NT, Windows Server, and BizTalk are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

Describing the Enterprise Architectural Space	1
Introduction	1
Enterprise Architectural Space Organizing Table	2
Architectural Viewpoints and Roles (Rows of the Organizing Table)	3
Architectural Space Interrogatives (Columns of the Organizing Table)	6
Combining the Viewpoints, Roles, and Interrogatives into the Organizing Table	6
Using the Organizing Table	8
Using the Table to Organize Patterns	8
Adding Patterns to the Organizing Table	12
Modifying the Table	12
Summary	13
Community	14
References	14
Contributors	14
Key to Pattern Names in Figure 4	14

Describing the Enterprise Architectural Space

Authors: David Trowbridge, Ward Cunningham, Matt Evans, Larry Brader, Microsoft Platform Architecture Guidance; Paul Slater, Wadeware.

Microsoft Corporation

June 2004

Summary: This document presents an organizing table that describes the enterprise architectural space, shows relationships among artifacts in the space, and demonstrates how different roles in your enterprise view enterprise architecture. This document also demonstrates how pattern authors can use the organizing table to organize existing patterns and to identify areas where patterns are not currently documented.

Introduction

Effective organization is critical to help us gain a full understanding of the complex world surrounding us. Standard and consistent organizing systems are used everywhere, from the Periodic Table of the Elements and the Biological Classification of Organisms, to the Dewey Decimal system in libraries. Such systems are also plentiful in the world of Information Technology. For example, the DNS system helps organize computers globally in a meaningful way, and file systems provide a directory structure to organize files in storage.

Enterprise-level software and system architecture are ripe for a similar organizing system. If you ask any group of technologists to describe the architecture of a system, you are likely to hear contradicting descriptions. Each person often has his or her own view of the system, which is accurate but different from the view of other technologists looking at the same system. A consolidated and consistent view of enterprise software-intensive systems could help technologists gain a shared understanding of the enterprise architectural space that is more complete and accurate.

This document presents an organizing table that the Microsoft patterns & practices team has used in pattern development for the past two years on releases such as *Enterprise Solution Patterns Using Microsoft .NET*. This paper exposes the early thinking about a dynamic way to organize and consume patterns. The authors anticipate that the structure of the organizing table will be refined over time and that the table will prove to be useful in many other efforts.

Enterprise Architectural Space Organizing Table

As enterprises respond to specific business initiatives, they produce many artifacts that capture architectural decisions, including: plans, notes, models, scripts, and code. Different roles in your enterprise use these artifacts to view the enterprise architecture in different ways. The potential for reuse increases if you have a meaningful way of organizing these artifacts.

Organizing the artifacts is a complex task, particularly as the enterprise and technologies change. Examining the space for all stable elements and cross-correlating all the relationships between elements would result in a multi-dimensional table that is not easy to display. A two-dimensional table allows you to analyze enterprise software-intensive systems in an intuitive way.

The Enterprise Architectural Space Organizing Table is such a two-dimensional table that captures and organizes business artifacts according to the decisions that produce them. The organizing table defines architectural viewpoints as rows, and interrogatives as columns. Figure 1 shows the basic structure of the organizing table. The particular rows and columns are discussed later in this paper.

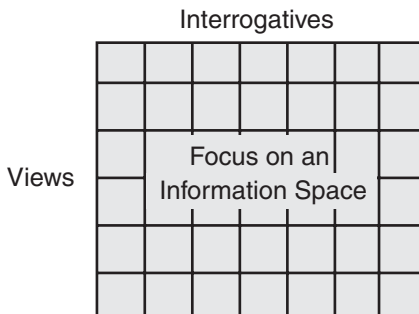


Figure 1
Organizing table structure

As you will see later, the choice of rows for this organizing table provides a particular focus on the enterprise information space. The organizing table is intended to:

- **Provide a comprehensive view of the architecture space.** The description is designed to organize architectural elements and demonstrate the relationships between them.
- **Be specific to enterprise software intensive systems.** You may choose to extend the description to meet the needs of your enterprise.

The organizing table builds on four key pieces of work: The Zachman Framework for Enterprise Architecture [Zach], IEEE 1471 [IEEE], Andersen Consulting's Enterprise Information Architecture [Andersen], and test-driven development. Like the Zachman Framework, this table uses interrogatives as columns and roles as rows. This table, however, shows a higher degree of granularity in the rows and a higher degree of specificity in the artifacts contained in each cell. Also, based on the principles of test-driven development, this table includes an additional column that identifies the test of success for every row. Ideally, for any given row, the artifacts contained in the **Test** column should trace to the artifacts contained in the **Purpose** column for validation.

The organizing table groups rows together in levels of architecture, which shows the influence of the Enterprise Information Architecture framework. This grouping of rows exposes the overall alignment and traceability between business and technology. Within each row are discrete viewpoints, which are influenced by the IEEE 1471 architectural standard description. Together, these elements produce a highly granular map of the enterprise space that is organized by viewpoints and interrogatives.

One of the main strengths of the organizing table is that you can use it to store many different kinds of artifacts. By extracting, distilling, and abstracting the information contained in the traditional artifacts associated with enterprise software-intensive systems, you can produce patterns, practices, and guidance for your enterprise. You can then store these patterns in the organizing table. For more information about storing patterns in the table, see "Using the Table to Organize Patterns," later in this paper.

Architectural Viewpoints and Roles (Rows of the Organizing Table)

The organizing table divides enterprise architecture into five broad enterprise viewpoints. These views appear in the table as rows, which are organized from general to specific as you scan them from top to bottom. The viewpoints are:

- Business architecture
- Integration architecture
- Application architecture
- Operational architecture
- Development Architecture

Although the viewpoints are a useful means of broadly categorizing architectural artifacts, the organizing table goes one step further by dividing the viewpoints according to specific roles that correspond to individuals with a particular skill set. As mentioned earlier, this additional detail allows you to trace the alignment of artifacts across the table.

Note: It is likely that the individuals in your enterprise do not map directly to the roles defined here. In a large enterprise, a role listed here might be assigned to a team (for example, architecture team). Conversely, one person may be responsible for many roles in a smaller enterprise. Nonetheless, most enterprise systems are viewed from the perspective of each of these roles at some point in time, and these roles can influence how you think about architectures as a whole.

The following paragraphs examine the viewpoints and the roles they contain in more detail.

Business Architecture Viewpoint

The business architecture viewpoint provides a basis for the other architectural viewpoints. Enterprise software exists to provide business value to your enterprise and must align with your business objectives. Without a well-defined business architecture in place, any attempts at enterprise architectures are likely to involve reactive, improvised technology decisions.

The business architecture viewpoint consists of the following roles:

- CEO
- General Manager
- Process Owner
- Process Worker

Integration Architecture Viewpoint

The integration architecture viewpoint is concerned with integrating systems that run in the enterprise inside and outside of the firewall. A simple enterprise may need only a few independent applications to run their business, but many enterprises need to integrate their applications both inside and outside the firewall. Inside the firewall, classic enterprise application integration (EAI) approaches are used to integrate systems at the data, function, API, and presentation levels. Often a message broker architecture is developed to perform intelligent routing, transformation, and business rule processing. Outside of the firewall, enterprises are connecting with other enterprises to create extended enterprise networks of trading partners that have cross-enterprise integration needs. This is the domain of business-to-business (B2B) integration servers (BizTalk) and Web services interoperability frameworks, which are designed to integrate multiple businesses at the process level.

The integration architecture viewpoint consists of the following roles:

- Enterprise Architect
- Designer
- Developer

Application Architecture Viewpoint

The application architecture viewpoint contains all of the system and software elements necessary to run an executable application, such as databases, Web servers, application servers, networks, presentation frameworks, components (both infrastructure and custom), run-time frameworks, business logic, and the applications themselves. Any applications used to produce business value are really executable knowledge or executable business process. People interact with these applications through various interfaces and perform some portion, or all, of a business process using an automated, executable tool.

The application architecture viewpoint consists of the following roles:

- Enterprise Architect
- Architect
- Designer
- Developer

Operational Architecture Viewpoint

The operational architecture viewpoint is concerned with operating the production system in a stable, secure, scalable, predictable, and managed fashion. This category contains elements related to event, remote and performance management, user administration, backup, monitoring, and tuning.

The operational architecture viewpoint consists of the following roles:

- Systems Architect
- Systems Engineer

Development Architecture Viewpoint

The development architecture viewpoint is concerned with implementing the other architectures. Applications must be built and maintained in a systematic, efficient manner. The development architecture is composed of elements related to this effort, such as design and development tools, repositories, build master utilities, test suites, tracking tools, and other tools.

The development architecture viewpoint consists of the following roles:

- Configuration Management Engineer
- Buildmaster

- Test Engineer
- Developer

Architectural Space Interrogatives (Columns of the Organizing Table)

Although viewpoints and roles provide discrete perspectives into the architecture, you can provide further granularity in the architectural space by categorizing artifacts according to generalized questions that are related to the business initiatives that produce the artifacts. These questions, or interrogatives, form the following columns of the organizing table:

- **Purpose (Why).** What is the reason for the architectural decision made in response to the initiative?
- **Data (What).** What information is required by or will be produced as a result of the decision called for in the initiative?
- **Function (How).** How do the architectural decisions made in response to the initiative work?
- **Timing (When).** When do events or actions called for, or recognized by, the decision take place?
- **Network (Where).** Where do the products of the initiative reside?
- **People (Who).** Who benefits from, or is otherwise affected by, the initiative? What is the interface through which that effect happens?
- **Scorecard (Test).** How do you know that the purpose of the initiative has been achieved?

The order of the columns is arbitrary, although it is useful to have the purpose column on the left, because it defines the reason for the architectural decision.

Combining the Viewpoints, Roles, and Interrogatives into the Organizing Table

Combining the viewpoints, roles, and interrogatives together produces an organizing table for enterprise software intensive systems. Figure 2 shows the organizing table, populated with descriptions of the artifacts it contains.

ENTERPRISE ARCHITECTURAL SPACE ORGANIZING TABLE

		1	2	3	4	5	6	7
Viewpoints		Purpose (Why)	Data (What)	Function (How)	Timing (When)	Network (Where)	People (Who)	Scorecard (Test)
Business Architecture	A CEO	- Strategic scope - Economic intent	Enterprise level dashboard	Enterprise process models	Business events and cycles	Locations where the enterprise operates	Organizational units	Balanced scorecard EVA, ROE, ROA
	B General Mgr	Operational goals and objectives	Operational dashboard	Business unit process models	Business unit master schedule	Business unit logistic network (nodes and links)	Business unit org chart with roles, security permissions and skill sets	P&L DSO
	C Process Owner	Process control objectives	Process control dashboard	Application process models	Process schedule, events, collaborations and state transitions	Communication links and devices needed for process automation	Process level actors with roles and permissions	%Utilization Defect Rates Cycle Times Step Count
	D Process Worker	-Activity objectives	Specific data inputs, processing and outputs	Task and activity based process models	Activity specific events, collaborations and state transitions	Activity specific devices and locations	Specific roles, accounts, passwords and permissions	Activity metrics
Integration Architecture	E Enterprise Architect	- Extended enterprise use cases - EAI use cases	Enterprise level data flows and replication strategies	EAI (API, method, data, user interface) - B2B integration - Global (GXA) - Message broker	Define business process models - BPEL4WS - Rosetta PIPs	-URIs for all trading partners -Integration servers and firewalls	Partners, customers, suppliers, system actors	- EE acceptance test - Performance against SLAs - Onboarding costs
	F Designer	System level use cases	XML Schemas, Half Maps,ETL, Batch feeds	Design level service, component, and subsystem models	Design level collaboration models	- Integration NFR* met	Roles, permissions, security requirements	EE integration tests - Reuse - Transaction metrics
	G Developer	Abbreviated use case descriptions Implement interfaces	Profiles, database instances, stored procs	XLANG, source code, scripts, batch files, executables	XLANG Executable vertical and horizontal slices	Process mapped to processors, links, protocols	Users mapped to roles within organizations	Unit tests, system tests
Application Architecture	H Enterprise Architect	Enterprise level use cases Process refactorings NFR*	Enterprise data models - Data distribution strategies	Enterprise level domain model and logical services	Enterprise level collaboration models (sequence and interaction)	Enterprise level system architecture -Nodes -Links -Locations	Enterprise level user profiles including demographics, psychographics, technographics	-I1 realizations -Cross application integration test -SLA metrics -Reuse
	I Architect	Application level use cases Mechanisms NFR	Application level data models	Application level -Domain models -Analysis model	Analysis level collaboration model	Application level system architecture - Nodes, devices - Links and segments - Processors	Information architecture, interaction maps, story boards, security requirements	I1 realizations Acceptance tests App SLA metrics Reuse
	J Designer	System level use cases Mechanisms NFR*	Tables, indexes, views, queries	Design level classes, component, subsystem and QoS	Design level collaboration models (sequence and interaction)	Design level system architecture -addresses -subnets -processors	Visual designs, wire frames, site maps	J1 realizations Integration tests Interfaces defined Reuse
	K Developer	- Abbreviated use case descriptions - Implement interfaces	Database instances , stored procs, etc.	Source code implementation units, Executables	Executable (vertical / horizontal slices)	-Processes allocated to processors - Production environ - NFR* met	Intuitive, easy to use executable interface relevant to user needs	Unit Tests User Adoption
Operational Architecture	L Systems Architect	Non functional requirements - high level	-Directory Design -Data storage design	-Monitoring and tuning -Remote management	Event management	High level network models Traffic analysis	Users, roles, permissions, security requirements	-Performance against SLA's - Operational metrics
	M System Engineer	Non functional requirements - detailed level	- Directory Implementation - Backup and recovery	Batch files, scripts, utilities	Fault management and recovery	Detailed network models - Network monitoring	User administration	Operational metrics
Development Architecture	N Config Mgmt. Engineer	- Change impact analysis - Rollbacks - Asset retention	-Repository -Dependency maps	Batch files, scripts, utilities	-Restore known configurations -Promote code	Logical and physical device information	Users, roles, permissions, security requirements	Asset management metrics
	O Buildmaster	-Build quality and quantify metrics -Reports	Source code, compile time dependencies, test data, results,	Compilers, build tools, system admin tools, test tools	Development process, events and schedule	Development test, and staging environment locations and accounts	Development team, system admin team, configuration mgmt team	-Build quality and quantify metrics -Reports
	P Test Engineer	-Quality and predictability metrics -Reports	-Test Cases -Test Data -Repository	Automated test suites - integration - acceptance - performance	Test plan with schedule and test cases	Test environment	Development team, buildmaster, Config mgmt team, QA mgmt	-Quality and predictability metrics -Reports
	Q Developer	-Software development efficiency and effectiveness -Reports	- Repository - Test Data	- IDE - Bug Tracking Tools - Debugging Tools - Test Tools - Modeling tools	- Iteration plan with schedule and features - Integration schedule	Development environment including integration machine(s)	Users, roles and permissions	-Software development efficiency and effectiveness -Reports

* Non-Functional Requirements

Figure 2
Enterprise Architectural Space Organizing Table

Note: For a larger version of this table, see the Organizing Table .pdf file.

Using the Organizing Table

The table provides a way of understanding the enterprise architectural requirements of an enterprise from multiple viewpoints and roles. You can use the table in a number of ways, including:

- **To understand the full extent of solutions in the enterprise.** Few people grasp the full extent of solutions required in the enterprise. You can identify specific artifacts within the frame and explore nearby cells to broaden your understanding of enterprise architecture.
- **To identify relationships between artifacts.** By examining subsections of the table, you can identify relationships between cells. Within cells, you can gain an understanding of artifacts that are related to one another. For example, an artifact that a particular roles uses may be related to another artifact that another role uses.
- **To cross-check elements in an enterprise for consistency and traceability.** For example, you can use the test interrogative to check the validity of the purpose interrogative.

Using the Table to Organize Patterns

The organizing table organizes the artifacts of enterprise software-intensive systems, and therefore provides a view of the overall architecture. You can also use the traditional artifacts of the architecture to create patterns, practices, and guidance, as Figure 3 shows.

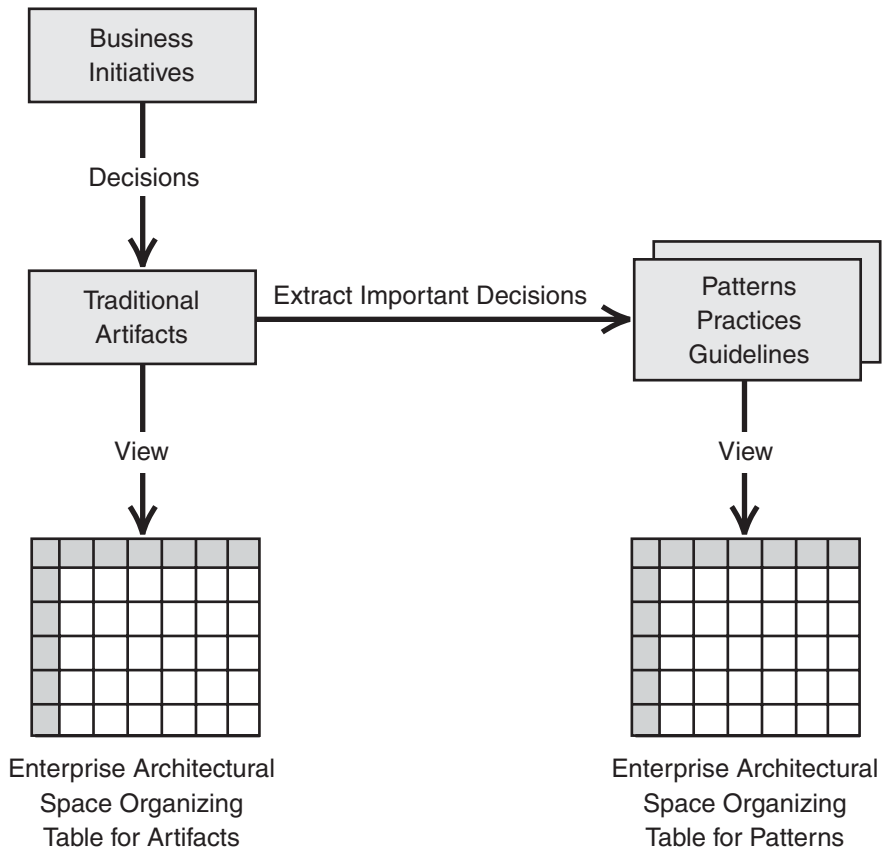


Figure 3
Typical uses of the organizing table

As the figure shows, you can extract important decisions and distill them into patterns, practices, and guidance that capture the collective learning of the enterprise. Placing this new learning in the organizing table provides particular benefits to pattern authors. As a pattern author, you can use the table in a number of ways, including:

- **To understand the context of patterns throughout the enterprise.** You can identify specific patterns within the frame and explore nearby cells to broaden your understanding of the enterprise architecture.
- **To identify areas of the table that do not contain patterns.** Areas of the table that do not currently contain patterns may reveal areas that the patterns community has not yet addressed.
- **To identify relationships between patterns.** By examining subsections of the table, you can identify relationships between cells and patterns within the cells. For example, a pattern that a particular role uses may be related to another pattern that another role uses.
- **To move toward a more granular and consistent classification of patterns, according to their different characteristics and uses.**

The Microsoft patterns & practices team uses the organizing table to categorize existing patterns and to identify areas where new patterns are emerging. Figure 4 shows the organizing table, with recent Microsoft patterns releases plotted on it. Each pattern appears as a number. To see patterns these numbers refer to, and the publications in which they appear, see “Key to Pattern Names in Figure 4” at the end of this paper.

		Purpose	Data	Function	Timing	Network	People	Scorecard
Business Architecture	CEO							
	General Manager							
	Process Owner							
	Process Worker							
Integration Architecture	Enterprise Architect		65 66 67 68 69 81 85 86 87 88	82 84 89 90 91 92 93		79		
	Designer		70 71 72 73 74 75	52 94 95 96 97 99 100 101 102 103 105 107 109		80		
	Developer		76 77 78	83 98 104 106 108 110 110				
Application Architecture	Enterprise Architect							
	Architect			14 15 47		16 17 18 34		
	Designer			2 4 6 8 10 12 19 22 25 27 29 35 36 37 38 39 40 41 42 43 44 46 48 49 50 51 55 56 57 59 60 61 62 63		31 32 33 58		
	Developer			3 5 7 9 11 13 20 21 23 24 26 28 30 45 53 54 64				
Operational Architecture	Systems Architect							
	System Engineer							
Development Architecture	Config Mgmt Engineer							
	Buildmaster							

Figure 4
 Microsoft patterns releases plotted on the organizing table

Adding Patterns to the Organizing Table

In the future, the authors of this paper want to add patterns from key pattern authors. By sharing the organizing table with the broader patterns community, we believe we can provide an extensive and more coherent view of all available patterns. By combining efforts, the patterns community can increase pattern usage and better meet the needs of developers and architects who use them.

You can contribute to this effort, by adding your own existing patterns to the organizing table.

► To add patterns to the organizing table

1. Examine the problem and context of the pattern.
2. Locate the most likely row and column where the pattern belongs.
3. Read the contents of this and nearby cells and place the pattern in the cell with the best match.

Modifying the Table

In some cases, you will find it useful to modify the table, to meet your specific needs. For example, you may choose to increase the granularity of the table by adding additional rows or columns to more precisely define artifacts or patterns. Members of the patterns & practices team have themselves modified the table in this way in certain publications. For example, Enterprise Solution Patterns introduced a deployment column, as shown in the following subsection of the organizing table.

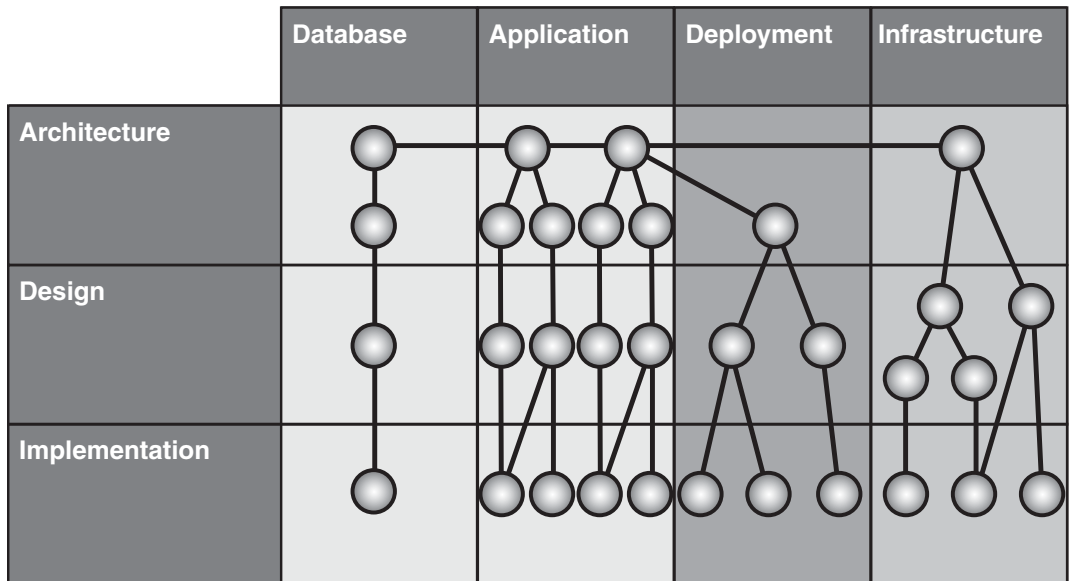


Figure 5
Enterprise Solution Patterns Organizing Table

This organizing table, which Enterprise Solution Patterns refers to as the Pattern Frame, is a subset of the Application Architecture row from the Enterprise Architectural Space Organizing Table. The Deployment column in Figure 5 added granularity to the Network column from the organizing table in Figure 2.

If you compare the Pattern Frame to the organizing table more carefully, you will notice that the column names were relabeled. The authors of Enterprise Solution Patterns customized the column names for the audience and to account for the constraints that they placed on this subset of the organizing table.

Summary

A comprehensive description of enterprise architecture artifacts helps you to reason and communicate about enterprise architectures. By using an organizing table containing viewpoints, roles, and interrogatives, you can help workers within your enterprise to understand the scope of enterprise software-intensive systems and improve business and technology alignment. You can also organize patterns, practices, and guidelines derived from these artifacts in the table. Organizing them allows you to categorize them, identify relationships between them, and determine areas where new patterns and practices may be appropriate.

Community

Questions? Comments? Suggestions? The authors of this paper urge you to join the Enterprise Architecture Space community on the Channel 9 Web site (<http://channel9.msdn.com/wiki/default.aspx/Channel9.EnterpriseArchitecturalSpace>). You can use this community forum to provide feedback on this paper and to contribute to the Enterprise Architecture Space Organizing Table.

References

[Andersen] Andersen Consulting, Goodyear, Mark. *Enterprise System Architectures: Building Client/Server and Web-based Systems*. CRC Press, 2000.

[IEEE] IEEE-Std-1471-2000, *Recommended Practice for Architectural Description of Software-Intensive Systems*.

[Zachman] Zachman, John. The Zachman Institute for Framework Advancement, <http://www.zifa.com>.

Contributors

Many thanks to the following reviewers who provided invaluable assistance and feedback: Wojtek Kozaczynski, Microsoft Platform Architecture Guidance; Phil Teale, Microsoft Corporation; Harry Pierson, Microsoft Developer and Platform Evangelism Architecture Strategy Team; Richard Sears, Sears and Associates; Bill McDonald, Ascentium Corporation; Blaine Wastell, Ascentium Corporation; United Kingdom Architect Council – Patterns Working Group.

Key to Pattern Names in Figure 4

The following table shows the patterns that Figure 4 refers to by number. The table indicates the names of the patterns, and the patterns & practices publications in which they appear—*Enterprise Solution Patterns Using Microsoft .NET* (ESP), *Data Patterns* (Data), or *Integration Patterns* (Integration).

Table 1: Key to Pattern Names in Figure 4

Pattern #	Pattern Name	Publication
2	Model-View-Controller	ESP
3	Implementing Model-View-Controller in ASPNET	ESP
4	Page Controller	ESP
5	Implementing Page Controller in ASPNET	ESP

Pattern #	Pattern Name	Publication
6	Front Controller	ESP
7	Implementing Front Controller in ASPNET Using HTTP Handler	ESP
8	Intercepting Filter	ESP
9	Implementing Intercepting Filter in ASPNET Using HTTP Module	ESP
10	Page Cache	ESP
11	Implementing Page Cache in ASPNET Using Absolute Expiration	ESP
12	Observer	ESP
13	Implementing Observer in .NET	ESP
14	Layered Application	ESP
15	Three-Layered Services Application	ESP
16	Tiered Distribution	ESP
17	Three-Tiered Distribution	ESP
18	Deployment Plan	ESP
19	Broker	ESP
20	Implementing Broker with .NET Remoting Using Server-Activated Objects	ESP
21	Implementing Broker with .NET Remoting Using Client-Activated Objects	ESP
22	Data Transfer Object	ESP
23	Implementing Data Transfer Object in .NET with a DataSet	ESP
24	Implementing Data Transfer Object in .NET with a Typed DataSet	ESP
25	Singleton	ESP
26	Implementing Singleton in C#	ESP
27	Service Interface	ESP
28	Implementing Service Interface in .NET	ESP
29	Service Gateway	ESP
30	Implementing Service Gateway in .NET	ESP
31	Server Clustering	ESP

(continued)

Pattern #	Pattern Name	Publication
32	Load-Balanced Cluster	ESP
33	Failover Cluster	ESP
34	Four-Tiered Distribution	ESP
35	Abstract Factory	ESP
36	Adapter	ESP
37	Application Controller	ESP
38	Assembler	ESP
39	Bound Data Control	ESP
40	Bridge	ESP
41	Command(s)	ESP
42	Decorator	ESP
43	Facade	ESP
44	Gateway	ESP
45	Implementing Data Transfer Object in .NET with Serialized Objects	ESP
46	Layer Supertype	ESP
47	Layers	ESP
48	Mapper	ESP
49	Mediator	ESP
50	MonoState	ESP
51	Observer	ESP
52	Naming Service	ESP
53	Page Data Caching	ESP
54	Page Fragment Caching	ESP
55	Presentation-Abstraction-Controller	ESP
56	Proxy	ESP
57	Remote Facade	ESP
58	Server Farm	ESP
59	Special Case	ESP
60	Strategy	ESP
61	Table Data Gateway	ESP

Pattern #	Pattern Name	Publication
62	Table Module	ESP
63	Template Method	ESP
64	Utility Component	ESP
65	Maintain Data Copies	Data
66	Application-Managed Data Copies	Data
67	Move Copy of Data	Data
68	Data Replication	Data
69	Extract-Transform-Load (ETL)	Data
70	Master-Master Replication	Data
71	Master-Slave Replication	Data
72	Master-Master Row-Level Synchronization	Data
73	Master-Slave Snapshot Replication	Data
74	Capture Transaction Details	Data
75	Master-Slave Transactional Incremental Replication	Data
76	Implementing Master-Master Row-Level Synchronization Using SQL Server	Data
77	Implementing Master-Slave Snapshot Replication Using SQL Server	Data
78	Implementing Master-Slave Transactional Incremental Replication Using SQL Server	Data
79	Topologies for Data Copies	Data
80	Master-Slave Cascading Replication	Data
81	Entity Aggregation	Integration
82	Process Integration	Integration
83	Implementing Process Integration with BizTalk Server 2004	Integration
84	Portal Integration	Integration
85	Data Integration	Integration
86	Shared Database	Integration
87	Maintain Data Copies	Integration
88	File Transfer	Integration
89	Functional Integration	Integration
90	Distributed Object Integration	Integration

(continued)

Pattern #	Pattern Name	Publication
91	Message-Oriented-Middleware Integration	Integration
92	Service-Oriented Integration	Integration
93	Presentation Integration	Integration
94	Broker	Integration
95	Direct Broker	Integration
96	Indirect Broker	Integration
97	Message Broker	Integration
98	Implementing Message Broker with BizTalk Server 2004	Integration
99	Message Bus	Integration
100	Broadcast Message Bus	Integration
101	Switch-Based Message Bus	Integration
102	Publish/Subscribe Message Bus	Integration
103	Publish/Subscribe	Integration
104	Publish/Subscribe with BizTalk Server 2004	Integration
105	Pipes and Filters	Integration
106	Pipes and Filters with BizTalk Server 2004	Integration
107	Gateway	Integration
108	Implementing Gateway with Host Integration Server 2004	Integration
109	Content-Based Routing	Integration
110	Content-Based Routing with BizTalk Server 2004	Integration