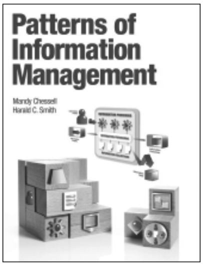
Common Information Models for an Open, Analytical, and Agile World

Mandy Chessell • Gandhi Sivakumar • Dan Wolfson
Kerard Hogg • Ray Harishankar

Foreword by Tim Vincent
IBM Fellow and Vice President
CTO for IBM Analytics Group

Related Books of Interest



Patterns of Information Management

By Mandy Chessell and Harald Smith

ISBN-13: 978-0-13-315550-1

Use Best Practice Patterns to Understand and Architect Manageable, Efficient Information Supply Chains That Help You Leverage All Your Data and Knowledge

In the era of “Big Data,” information pervades every aspect of the organization. Therefore, architecting and managing it is a multi-disciplinary task. Now, two pioneering IBM architects present proven architecture patterns that fully reflect this reality. Using their pattern language, you can accurately characterize the information issues associated with your own systems, and design solutions that succeed over both the short- and long-term.

Beyond Big Data

Using Social MDM to Drive Deep Customer Insight

By Martin Oberhofer, Eberhard Hechler, Ivan Milman, Scott Schumacher, Dan Wolfson

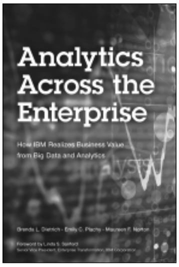
ISBN-13: 978-0-13-350980-9

Enterprises have long relied on Master Data Management (MDM) to improve customer-related processes. But MDM was designed primarily for structured data. Today, crucial information is increasingly captured in unstructured, transactional, and social formats: from tweets and Facebook posts to call center transcripts.

In *Beyond Big Data*, five of IBM’s leading data management experts draw on pioneering experience with IBM’s enterprise customers and show how Social MDM can help you deepen relationships, improve prospect targeting, and fully engage customers through mobile channels.

Business leaders and practitioners will discover powerful new ways to combine social and master data to improve performance and uncover new opportunities. Architects and other technical leaders will find a complete reference architecture, in-depth coverage of relevant technologies and use cases, and domain-specific best practices for their own projects.

Related Books of Interest



Analytics Across the Enterprise

How IBM Realizes Business Value from Big Data and Analytics

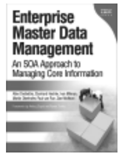
By Brenda L. Dietrich, Emily C. Plachy, Maureen F. Norton

ISBN-13: 978-0-13-383303-4

How to Transform Your Organization with Analytics: Insider Lessons from IBM's Pioneering Experience

Analytics Across the Enterprise demystifies your analytics journey by showing you how IBM has successfully leveraged analytics across the enterprise, worldwide. Three of IBM's pioneering analytics practitioners share invaluable real-world perspectives on what does and doesn't work and how you can start or accelerate your own transformation. This book provides an essential framework for becoming a smarter enterprise and shows through 31 case studies how IBM has derived value from analytics throughout its business.

Whatever your industry or role, whether a current or future leader, analytics can make you smarter and more competitive. *Analytics Across the Enterprise* shows how IBM did it—and how you can, too.

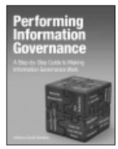


Enterprise Master Data Management

An SOA Approach to Managing Core Information

Dreibelbis, Hechler, Milman, Oberhofer, van Run, Wolfson

ISBN-13: 978-0-13-236625-0

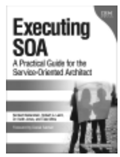


Performing Information Governance

A Step-by-step Guide to Making Information Governance Work

By Anthony David Giordano

ISBN-13: 978-0-13-338565-6

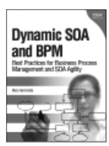


Executing SOA

A Practical Guide for the Service-Oriented Architect

Bieberstein, Laird, Jines, Mitra

ISBN-13: 978-0-13-235374-8

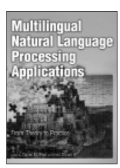


Dynamic SOA and BPM

Best Practices for Business Process Management and SOA Agility

Fiammante

ISBN-13: 978-0-13-701891-8



Multilingual Natural Language Processing

From Theory to Practice

Bikel, Zitouni

ISBN-13: 978-0-13-715144-8

Sign up for the monthly IBM Press newsletter at
ibmpressbooks.com/newsletters

This page intentionally left blank

Common Information Models for an Open, Analytical, and Agile World

This page intentionally left blank

Common Information Models for an Open, Analytical, and Agile World

Mandy Chessell, Gandhi Sivakumar,
Dan Wolfson, Kerard Hogg,
Ray Harishankar

IBM Press
Pearson plc
New York • Boston • Indianapolis • San Francisco
Toronto • Montreal • London • Munich • Paris • Madrid
Cape Town • Sydney • Tokyo • Singapore • Mexico City
ibmpressbooks.com

IBM Press Program Managers: Steven M. Stansel, Ellice Uffer

Cover design: IBM Corporation

Associate Publisher: Dave Dusthimer

Marketing Manager: Stephane Nakib

Executive Editor: Mary Beth Ray

Publicist: Heather Fox

Editorial Assistant: Vanessa Evans

Senior Development Editor: Christopher Cleveland

Managing Editor: Kristy Hart

Designer: Alan Clements

Senior Project Editor: Lori Lyons

Copy Editor: Paula Lowell

Indexer: Publishing Works

Senior Compositor: Gloria Schurick

Proofreader: Katherine Matejka

Manufacturing Buyer: Dan Uhrig

Published by Pearson plc

Publishing as IBM Press

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact international@pearsoned.com.

Library of Congress Control Number: 2015932607

ISBN-13: 978-0-13-336615-0

ISBN-10: 0-13-336615-4

Text printed in the United States on recycled paper at Courier Westford, in Westford, Massachusetts.

First printing: May 2015

To my family and friends,
who have encouraged and supported me over the years, with love:
Sarah, Jane, Ian, Ray, Terri, Ben, Ella, Keith, Kay, Dan, Tim and Chris.
—Mandy Chessell

To my Lord Krishna and Mother Sri Aurobindo,
Who gave me the wisdom in this initiative.
To my dad R. Barathan,
Who constantly followed up and motivated me.
—Gandhi Sivakumar

To my family, especially my wife Danelle, for their understanding and support.
—Dan Wolfson

To my mom and dad Saraswathi and Raman, who have always been my inspiration;
and to my wife Prema and daughters Krupa & Keerthi for their love,
understanding, and support.
—Ray Harishankar

Contents at a Glance

This page intentionally left blank

Contents

Foreword by Tim Vincent

Regardless of whether you are modeling new analytic algorithms to drive business decisions, creating new mobile applications, or building cloud-based systems of engagement, you will need to work with data, and in many cases this data comes from multiple places. The plethora of different types of data, the access mechanisms to the data, and the mixing of batch, real-time, and interactive models into a single paradigm are all leading to new challenges. How are the needs of multiple consumers reconciled with the structure of the original sources, and how do you reliably transform data between these desired structures? Traditional data modeling techniques are well known but are unfortunately seen as onerous and slow moving. However, this does not eliminate the necessity of thinking through what shape you need the data in at each stage of its lifetime.

The push to a self-service model, where business users and application developers can find, access, and consume data from multiple places, is becoming more prevalent. Relentless pressure to improve time to value is creating a demand to free these users from the need to work with central IT, and new tools are emerging for locating and preparing data. Satisfying this self-service paradigm requires new ways of managing data using both business-level and technical-level metadata. The business-level metadata will be used to find information (sometimes referred to as the shop for data paradigm) as well as prepare and transform the data into its final consumable form. Effectively, the concept of data modeling is going to radically evolve to become a key component of the self-service paradigm.

Tim Vincent
IBM Fellow and Vice President
CTO for IBM Analytics Group

Preface

About This Book

IT is going through a major upheaval. So much of modern life is becoming digitized, resulting in a deluge of new information. Mobile devices are ubiquitous, providing many people with their principle mechanism for shopping, banking, finding information, collaborating, and organizing their life and health. Cloud-based delivery is changing the cost models around IT infrastructure. Any one of these changes would be disruptive. All three occurring at once—and feeding the growth of each other—creates a huge gap between an organization's current needs and the services provided by its existing IT systems.

IT teams are striving to be agile, adopting new technologies and focusing on delivery of new capability while trying to ignore the existing systems. However, a limit exists to the success of this approach.

- The existing systems implement the logic that enables an organization to deliver on the promises that it is making through the new capability. So, integration is required between the new capability and the existing systems in order to retrieve information and initiate the processing of new accounts, bookings, orders, payments, and many more processes.
- People expect an organization to behave coherently and appropriately within a specific context. This means it cannot afford to operate as a number of independent units. Each unit needs to use shared information, use analytics to enhance the decision-making, and feed back the results of its actions to enable others to follow on effectively.

This book is about establishing an effective set of standards to efficiently integrate systems together, share information, and manage change. It focuses on how information is represented and understood at different points in the IT systems and the correspondence between them.

Modeling information helps us design, implement, and maintain information systems—no matter if they are simple web APIs being deployed in a new environment or if they are part of an existing, complex web of applications and data. Modeling is useful for individuals as well as for teams, architects, application developers, and data scientists. Information modeling helps us clarify the structure, meaning, and intent of the information we exchange. It plays a key role in today's world of rapidly changing requirements, agile development, and the increasing focus on analytics to make everything better.

Common information models define standard representations for information that can be used in many places, such as:

- Interfaces between applications
- Interfaces between applications and data sources
- Interfaces used to synchronize information between databases
- Interfaces that exchange information between people

Common information models facilitate design, deployment, and evolution of software systems because they document and share the knowledge of how the different kinds of systems should interact, enabling people to analyze the impact of changing requirements and coordinate the design and implementation of changes to accommodate these changing requirements. These techniques increase in importance as the complexity of the systems increases and the number of people who need to share a common understanding increases.

In this book we discuss the role of common information models: what they are, how to build them, how to use them, and how to maintain them. We provide examples from our experience and from industry standards. Our goal is to encourage you to adopt these techniques, to effectively consume existing information models, and to make you more effective in today's agile world.

Intended Audience

This book is intended for enterprise architects, information architects, and solution architects who are responsible for defining how information should be shared between IT systems. It is intended for implementers who translate design into working systems. Finally, it is intended for project managers who need to coordinate the resources needed to deliver and maintain these systems.

Structure of the Book

Common Information Model for an Open, Analytical, and Agile World is divided into the following chapters.

Chapter 1, "Introduction"

In Chapter 1 we illustrate why the modeling of information is critical for meeting today's software engineering challenges, particularly when the organization is focused on mobile applications, cloud deployments, and big data analytics. We also introduce the case study that will be used throughout this book to illustrate the value, approach, and best practices for maintaining a common information model.

Chapter 2, "Inside the Common Information Model"

Chapter 2 explores the content of the common information model. Although we refer to the common information model as if it were a single artifact, it is made up of multiple parts. Each common information model has:

- **Scope:** The subject areas, or topics, that the models cover
- **Perspectives:** The types of models it contains
- **Depth of detail:** How close to the physical implementation it specifies

We explain each of these aspects so you can select which types of artifacts are relevant for your organization.

Chapter 3, "Structural Patterns for the Common Information Model"

In Chapter 3 we provide more guidance on how to structure a common information model through five software design patterns that illustrate different approaches.

Chapter 4, "Modeling Best Practices"

For many people, the act of creating models and common definitions is a mystery. There are books that describe the different modeling notation and syntax of different data structure definitions, but very little detail on the thinking process behind the modeling process. In Chapter 4 we share some of the best practices we have found useful in our work.

Chapter 5, "Governance"

A common information model is an asset common to many IT initiatives undertaken by an organization. Chapter 5 describes the broader governance context and how governance of the common information model "fits in." The chapter then goes on to identify and discuss the key elements that make up governance of a common information model.

Chapter 6, "Moving Beyond the Hammer"

In Chapter 6 we cover the types of tools you need to manage a common information model. The trick is to create a master copy of the model and deploy the contents into the tools of the target consumers.

Chapter 7, "System Characteristics"

In Chapter 7 we explore the nonfunctional behaviors of software (such as availability, maintainability, scalability, performance, and security) and how they can affect the design of the common information model.

Chapter 8, "Building Business Value"

In Chapter 8 we discuss how the business value of using a common information model can be developed through a systematic approach that expands both the scope and usage of the common information model while ensuring there is freedom to innovate.

Chapter 9, "Real-World Deployment Study"

Chapter 9 describes a large service-oriented project that made extensive use of a common information model.

Chapter 10, "Looking Forward"

In our concluding chapter we reflect on our experiences in using common information models in our work, and how the role of common information models is changing, as well as our perspectives on its future.

Appendixes

The appendixes include a glossary of technical terms and a bibliography.

What Is Not Covered

This book is an introduction to a broad topic. Information modeling is an active, ongoing area of research and development with more than 30 years' worth of techniques, tools, and modeling languages, with many excellent publications that we could never hope to summarize (our bibliography reflects some of the favorites that reside on our bookshelves). Common information models (CIMs) have been developed through the efforts of many standards bodies and open source organizations; again, this book is not an exhaustive inventory of such standards, but does discuss a representative few.

It is important to note that the term *common information model* or *CIM* can itself be confusing. As the notion of a common information model has become so important across many different industries and functions, different standards bodies have created their own models that are referred to as a CIM for their particular needs. There is a CIM for computer management systems, a CIM for the electrical power industry, and probably several more. This book is not exclusively about these existing CIM models, but it does discuss ideas and techniques that can help you to apply these models to your organization.

Acknowledgments

The content of this book has been gathered from our collective experiences working with IBM colleagues and clients over many years. In particular, we would like to recognize and thank Dougal Watt, Pat O'Sullivan, Steve Lockwood, and Tim Vincent for their help in pulling together this material.

We would also like to thank the members of Pearson who have expertly guided us through the publishing process.

About the Authors

Mandy Chessell CBE FREng CEng FBCS

Mandy is an IBM Distinguished Engineer, Master Inventor, and member of the IBM Academy of Technology. Her current role is the Chief Architect for Information Solutions in the IBM Analytics Group CTO office. She leads the design of common information management patterns for different industries and solutions. This includes the Data Reservoir, Next Best Action solution, and the strategy for information governance.

In earlier roles Mandy led the development of new features for the CICS®, Encina, TxSeries, WebSphere®, and InfoSphere® products. She has more than 50 issued patents worldwide in the fields of transaction processing, event management, business process management, and model-driven development.

Outside of IBM, Mandy is a Fellow of the Royal Academy of Engineering and a visiting professor at the University of Sheffield, UK. In 2001, she was the first woman to be awarded a Silver Medal by the Royal Academy of Engineering, and in 2000, she was one of the "TR100" young innovators identified by MIT's *Technology Review* magazine. Mandy also has been granted an honorary fellowship of the Institution for Engineering Designers (IED) and an honorary doctorate of science by the University of Plymouth.

Mandy's recent publications include:

- "Patterns of Information Management" by Mandy Chessell and Harald Smith, http://www.informit.com/store/patterns-of-information-management-9780133155501?WT.mc_id=Author_Chessell_PoIM
- "REDP5120: Governing and Managing Big Data for Analytics and Decision Makers," http://www.redbooks.ibm.com/redpieces/abstracts/redp5120.html?Open
- "REDP5012, Smarter Analytics: Information Architecture for a New Era of Computing," http://www.redbooks.ibm.com/abstracts/redp5012.html?Open
- "REDP5043, Smarter Analytics: Taking the Journey to IBM Cognitive Systems," http://www.redbooks.ibm.com/abstracts/redp5043.html?Open
- "REDP4888: Smarter Analytics: Driving Customer Interactions with the IBM Next Best Action Solution," http://www.redbooks.ibm.com/abstracts/redp4888.html?Open

For more information see Mandy's LinkedIn page (http://www.linkedin.com/pub/mandy-chessell/22/897/a49) and Wikipedia page (http://en.wikipedia.org/wiki/Mandy_Chessell).

Gandhi Sivakumar

Gandhi Sivakumar is an IBM® Senior Certified Architect and service-oriented architecture (SOA) solution designer. Gandhi possesses 23 years of experience in the industry and has demonstrated technical leadership in large and complex programs of IBM including Telecom, Transport, Banking, and Human Services industries. Gandhi has been a technical champion in leading solution, integration, information, and infrastructure architectures across complex programs.

IBM honored Gandhi as Master Inventor for her innovations in addition to multiple Outstanding Technical Achievement Client Awards for the impact she created in client engagements. Gandhi has filed more than 70 patent applications spanning networks, infrastructure, data, integration, and others with IBM, and she holds a number of honorary roles within and outside IBM. Gandhi has published a number of papers and articles in IEEE and developerWorks. Gandhi served as a board member of the Australian Computer Society and has been portrayed as one of the leading technical women in IBM.

Dan Wolfson

Dan is an IBM Distinguished Engineer and the chief architect/CTO for the InfoSphere segment of the IBM Analytics Group. He is responsible for architecture and technical leadership across the rapidly growing areas of information integration and quality for big data including information quality tools, information integration, master data management (MDM), and metadata management. Dan is also Chief Architect for Hybrid Cloud Integration, working closely with peers throughout IBM.

Dan has more than 30 years of experience in research and commercially distributed computing, covering a broad range of topics including transaction and object-oriented systems, software fault tolerance, messaging, information integration, business integration, metadata management, and database systems. He has written numerous papers and blogs, and is the co-author of "Enterprise Master Data Management: An SOA Approach to Managing Core Business Information" and "Beyond Big Data: Using Social MDM to Drive Deep Customer Insight." Dan is a member of the IBM Academy of Technology and an IBM Master Inventor. In 2010, the Association of Computing Machinery (ACM) recognized Dan as an ACM Distinguished Engineer.

Kerard Hogg

Kerard is an Executive IT Architect at IBM. He has more than 30 years of experience in information technology, primarily as an IT architect. Kerard has provided technical leadership and governance on many large and complex IT projects, primarily in the telecommunication industry. Many of these engagements have used a common information model as a basis for complex systems integration solutions.

IBM has awarded Kerard a Global Excellence Award for outstanding achievement as an IT architect. Kerard has presented on service-oriented architecture outside IBM including the Australian Computer Society and ACM Research Conference.

Ray Harishankar

Ray Harishankar is an IBM Fellow and Vice President of Technology & Innovation within IBM Global Business Services®. Ray defines and operationalizes strategies for IBM to have a strong portfolio of solutions and assets, and assists clients in adopting and benefiting from these assets. Ray collaborates with IBM Research to identify opportunities for development of innovative capabilities that drive value to our clients.

Ray is an industry expert on systems of engagement that include mobile, analytics, social, and cloud computing technologies. Ray is actively engaged with customers across multiple industries and with a focus on banking, insurance, retail, and Smarter Cities®. Ray is on the technology advisory council for selected customers and universities and provides guidance to them on business-related technology decisions. Ray is currently a member of the Strategy Council for the College of Engineering at The Ohio State University.

Ray has been with IBM since 1999. He was nominated as a Distinguished Engineer in 2003 and as an IBM Fellow in May 2006. Appointment to IBM Fellow is the highest honor that an employee can receive for technical innovation. In 2009 Ray was named an Asian American Engineer of the Year. In October 2013, Ray was honored with a Distinguished Alumnus award by The Ohio State University College of Engineering. Ray holds seven patents and has filed several more.

Ray holds a master's degree in computer science from The Ohio State University. Ray can be reached at harishan@us.ibm.com.

This page intentionally left blank

This page intentionally left blank

CHAPTER 3

Structural Patterns for the Common Information Model

"14. To work our way towards a shared and living language once again, we must first learn how to discover patterns which are deep, and capable of generating life."

Christopher Alexander, "The Timeless Way of Building."
New York, Oxford University Press (1979): xii

In This Chapter:

We now know that the common information model is not a single artifact. It is made up of many parts—this content is tuned to the needs of the organization. In this chapter we provide more guidance on how to structure a common information model. To do this we describe five software design patterns that illustrate different approaches to structuring your common information model.

Introduction

Software design patterns[1] are a useful way to explain alternative design choices. They use simple English statements to cover

- The **context** of the design decision
- The **problem** that needs to be solved
- An **example** scenario that illustrates the problem

1. *Patterns of Information Management* by Mandy Chessell and Harald Smith, IBM Press, 2013, ISBN 0-13-315550-1

- The **forces** that make the problem difficult to solve
- A description of the **solution**
- An illustration of the solution showing how the **example** scenario was **resolved**
- A description of the **consequences** of using the solution, expressed as both benefits and liabilities
- Pointers to **known usages** of this solution
- Pointers to **related patterns**

This section covers five basic high-level structural patterns for a common information model. The first pattern is the basic definition of a common information model. It establishes the need for the common information model and is summarized in Table 3.1.

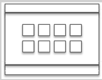
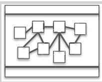
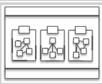
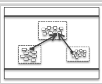
Table 3.1 Generic design pattern for a common information model

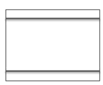
Icon	Pattern Name	Problem	Solution
	**COMMON INFORMATION MODEL**	An organization is struggling to integrate its IT systems and business operations. This integration may be required to increase its efficiency, embrace new technology, expand its business, and/or improve its customer service.	The organization should develop a shared understanding of its information's terminology, meaning, and structure in order to facilitate agile and effective integrated operations.

The subsequent patterns are specializations of this first common information model pattern that each support a particular business need. A specific implementation of a common information model may focus on one pattern, or may combine these patterns if the stakeholders want to address multiple challenges simultaneously.

Table 3.2 summarizes the four specialized patterns, and the details of each of the patterns follow the table.

Table 3.2 Specialized structural patterns for a common information model

Icon	Pattern Name	Problem	Solution
	**CONCEPT BEADS**	An organization needs common definitions of the simple concepts that are used in many applications to improve the efficiency and consistency of the work of its developers. However, it does not have a strong skill base in modeling.	Create a common information model that defines a clear meaning for each concept and a simple structure for how to record information about an instance of this concept.
	**CONTINUOUS FABRIC**	An organization needs to integrate related information from a wide range of information sources.	Create a common information model that defines the meaning, structure, and relationships among all the core concepts.
	**ENCAPSULATED VIEWS**	An organization needs to exchange information for multiple purposes.	Create a common information model that defines small clusters of related concepts that can be used as structures for exchanging information.
	**UNIFYING CONTEXT**	An organization needs to unify operations that have been independent in the past.	Create a common information model that maps existing terminology and definitions to a set of definitions that represents a canonical view of the subject area.

Common Information Model 

Context

An organization is looking to improve the methods it uses to develop and integrate IT systems into its operations.

Problem

An organization is struggling to integrate its IT systems and business operations. This integration may be required to increase its efficiency, embrace new technology, expand its business, and/or improve its customer service.

Organizations today have many IT systems that have been developed over time. They are struggling to innovate while trying to maintain and manage existing IT systems. Market forces are driving organizations to become more integrated, to support new channels (such as mobile), exploit new platforms (such as cloud and big data), and support a more social way of working both inside and with external parties such as customers and business partners. What can they do to provide a stable foundation for maintaining a state-of-the-art IT capability?

Example

A fictitious travel agent, GKDMR Travel, has many systems that have been acquired over the years as the business has grown. Its systems are getting old and it wants to make better use of modern approaches such as mobile, social media, cloud-based deployments, open source software, and analytics. The company's budget is not large, but it anticipates having to change some of its systems to accommodate its vision. However, many of the existing systems will remain and there is a need to interface them to the new systems.

Forces

- Existing systems provide key capability to the organization. They are expensive to replace and must continue to operate while any changes are made to them.
- An organization's information is distributed among its existing IT systems, in people's heads, in unmanaged files on employee laptops, in paper documents, and externally on the Internet, or in business partner systems.
- The information within an existing system is rarely as good a quality as the organization that owns it believes.
- An organization will use inconsistent terminology across its many departments, professional disciplines, and internal fiefdoms. Sometimes the same term is reused for different purposes, or even when the meaning is consistent, assumptions about its timeliness, precision, and accuracy will differ among the different groups inside the organization.
- The database schema of an application does not document all the information available through the application interface. Some key values are derived. For example, GKDMR Travel's booking system records all the trips a customer has booked with the company to the database. This information is visible in the database schema. However, GKDMR Travel operates a loyalty scheme, which

provides discounts to customers who book trips regularly with it. The loyalty scheme was added to the booking system after it went into production. As such, to save time, the loyalty status was not added to the database schema. Instead it is calculated in the application logic whenever a new holiday is booked. As a result, an important piece of information about a customer that could be useful for analytics or another customer service application is locked in a single application's logic.

Solution

The organization should develop a shared understanding of its information's terminology, meaning, and structure in order to facilitate agile and effective integrated operations.

Information is at the heart of an organization's ability to service its customers, deliver on its promises, and collect the expected rewards for its services. The types of information that an organization holds are stable, although the scope of information available has been growing recently with the digitization of many aspects of our lives.

The organization can develop a set of common information definitions that captures:

- The meanings of key concepts, facts, events, and activities used by the organization
- The preferred structures that should be used to store and exchange this information
- The types of values that must be captured to describe them

These common information definitions are implemented as a collection of models and definitions. This collection covers the portion of the organization's information that needs to be shared and synchronized. It must represent many perspectives on this information and be consumable in different programming environments. Collectively, these models and related artifacts are called the common information model.

An organization can choose which types of models to create, and to which level of detail. The focus will depend on where change happens most often and where the cost of change is high or notoriously error prone.

The different types of models may not be entirely consistent. However, the closer they are, the less transformation is required as information flows around the enterprise.

Consequences

Benefits:

- If an organization can understand the information it uses, then it is better able to assess which systems are important, how best to manage the information it has, what types of information need to be made available to new applications, and where special care must be taken to protect valuable or sensitive information.
- A well-formed common information model creates an adaptable definition of how information should be represented and shared at key places where the organization needs to synchronize or control its operations. Without it, information sharing can be ad hoc and developed as a number of inflexible point-to-point solutions.
- When a new project is started, the common information model is an invaluable planning tool for identifying what type of information is needed, where it is located, and how it should be structured in the new capability.
- When an existing application built to the guidelines of the common information model must be maintained, the development team has the common information to guide its understanding of the existing code.

Liabilities:

- For a common information model to deliver value it must be treated as an asset, with an owner who is responsible for its ongoing maintenance and executive support to ensure it is properly governed and to encourage use of the common information model content.
- The contents of a common information model must be easily consumable by the teams that are building and maintaining the IT systems. Ideally, physical artifacts such as interfaces and schema would be generated from the common information model and included in the developers' working toolset. Thus it becomes easier for the development team to use the common model rather than create its own. The common information model will be welcomed and adopted when it saves the development team time and effort.
- In some organizations, where data modeling is not widely understood, the generated physical artifacts are the only part of the common information model that many developers see. In this case, these physical artifacts should include comments and annotations covering the semantic definitions of the model.

Example Resolved

GKDMR Travel has three major projects that will be assisted through the creation of a common information model:

- It is building smartphone and tablet applications for its customers and customer-facing staff. These applications will be specialized to the needs of the users being served and will integrate with the existing systems that GKDMR Travel has running. The people using these applications will also receive printed documentation and may use the company website, too. GKDMR Travel needs to develop these applications quickly, but at the same time believes the information these applications display should be consistent with the other information these people see.
- It wants to offer real-time alerts and actions to support its traveling customers with relevant information, rebooking when weather and other factors disrupt travel and offering other relevant offers. This capability is going to need information from existing systems that is integrated with external information.
- It wants to improve the management reporting on the state of the business and the trends they see. The aim is to optimize the operations of the business.

The plan is to create a common information model that includes the following:

- A glossary of terms[2] that describes the meaning of the different types of information that will be used by customers and staff who are using the new capability. This includes the information displayed by the new applications along with the existing systems.
- A model of the objects that will be used in the mobile applications, ensuring they display consistent structures and each field in the structure has a consistent name so it can be matched with printed information and the information on the company website.
- A model of the service interfaces that define how information will be exchanged between the existing systems and the new ones.
- A persistence model that describes how data should be consolidated and linked together.

2. A glossary of terms is a conceptual semantic model, as described in Chapter 2.

The consistency that occurs between these models will speed up the collaboration between the different teams by implementing new capability and extending and enhancing the existing systems, which will then enhance the quality of the resulting applications.

Known Uses

Some of the key uses of a common information model within an organization are

- Creating a consolidated repository of information from different sources, such as a data warehouse, operational data store, and master data management hub.
- Defining canonical information services for exchanging information in a service-oriented architecture, or for consumption by a business process management (workflow) engine.
- Identifying the information that needs to be supplied to the corporate reporting platform.

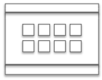
Related Patterns

The following patterns are specializations of the common information model pattern. Each supports a different purpose and style of usage for the common information model:

- **Concept Beads pattern:** Supports simple, fine-grained models of core concepts. Suitable for an organization that needs to quickly assemble simple applications that would benefit from consistency in implementation since they use many common concepts.
- **Continuous Fabric pattern:** Supports an organization where there is high value in understanding the relationships between people, assets, events, and activities; this provides a blueprint for how the landscape of information links together.
- **Encapsulated Views pattern:** Supports an organization that wants to develop common interfaces to information irrespective of how or where the information is stored. This style is called information virtualization and is a common approach in service-oriented architectures.
- **Unifying Context pattern:** For an organization that has historically operated as multiple independent units and wants to become more consistent and integrated in its use of information.

The sections that follow describe these patterns in more detail.

Concept Beads



Context

An organization is looking to improve the methods it uses to develop and integrate IT systems into its operations.

Problem

An organization needs common definitions of the simple concepts that are used in many applications to improve the efficiency and consistency of the work of its developers. However, it does not have a strong skill base in modeling.

Some organizations do not have a strong skill base in data modeling. This is often true where developers are using agile development, because their focus is on rapid prototyping and delivering solutions incrementally. Without constant refactoring, the code developed using this approach can become bloated and inconsistent. As a result, mature agile development teams have a strong emphasis on standards and consistent approaches.

How is it possible to bring the benefits of a common information model to a team of agile developers who do not want to spend time creating, or learning, a large data model?

Example

The mobile development team at GKDMR Travel is working on three new applications:

- A customer application that will work on a smart phone or tablet and enable people to maintain their customer details, see their loyalty status and any outstanding offers they have, and browse the holiday destination catalog, in which they can view other customers' feedback, book holidays, comment on their experiences, and pay their balance.
- A tablet application for the staff working at the holiday destinations allowing them to receive the guest lists and details of the meeting points, receive instructions, report incidents, and interact with guests while they are on holiday through text messages or email.
- A tablet application for the staff in the travel agent's office for helping with customer enquiries, booking holidays, documenting issues, noting stock levels of brochures, and reviewing the current promotions.

These applications need to be generated quickly, with iterations to allow the potential users to provide feedback on the suitability and ease of use of the applications. How does the team ensure consistency in the way these applications use the enterprise data around personal customer details, stores, products, orders, packages, and payments?

Forces

- Employing a common information model takes time. The team needs to learn about the existing content, develop new definitions, and correct errors.
- Agile development is often used where the requirements are evolving with the code. This is particularly common when the project is exploring a new technology, capability, or approach.
- The iterative nature of agile development means there is not time to wait for new model content to be developed, reviewed, and approved before it is used in the code.

Solution

Create a common information model that defines a clear meaning for each concept and a simple structure for how to record information about an instance of this concept.

The concept beads common information model is defined as a set of discrete concepts, as illustrated in Figure 3.1.

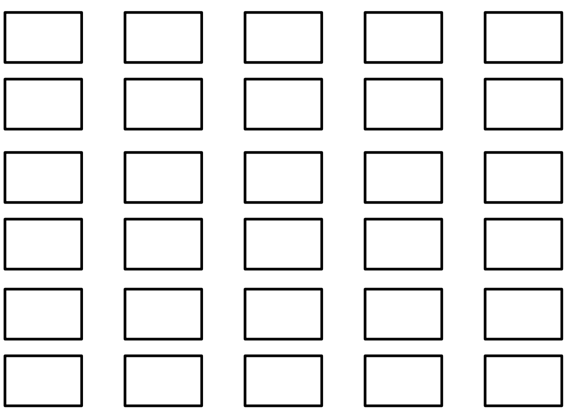


Figure 3.1 The structure of a concept beads common information model

Each concept definition has a business language description and preferred structures, possibly shown as a logical model for design documentation and then physical artifacts for different programming environments. These physical artifacts could be XML structures, JSON interfaces, Java object definitions, or any others that are commonly needed by developers, as illustrated in Figure 3.2.

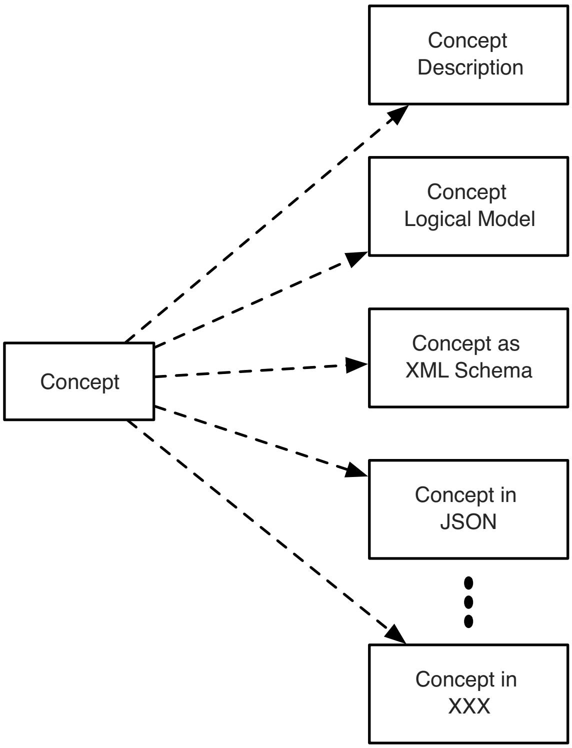


Figure 3.2 Adding detail to a concept beads common information model

The concept beads style of model is used like a Chinese menu. Developers select the concepts they need for their code, link them together as appropriate, and include them in their code. Developers will adopt the standards defined in the concept beads common information model rather than create their own if the standards are easier to use, so the common information model artifacts should be placed so they are easy to search and download into the developers' tools.

Changes to the definitions are developed and iterated on using an agile governance approach. Because the common information model is very modular, changes are localized and can be developed, approved, and deployed in a short period of time.

Consequences

Benefits:

- The concept beads common information model can be developed incrementally as part of an agile development process. New elements can be developed as required by the development team and integrated into the model within the sprint (iteration) that identifies a new concept is required.
- The concept beads model is easy to consume, particularly by developers, because only a small part of it must be understood for it to be used, and it can be simply translated in programming artifacts for direct consumption into code.
- The concept beads model is flexible because concepts can be linked and combined in whichever way the developer wants.

Liabilities:

- The concept beads common information model does not provide any guidance on how concepts that are frequently linked together should be related. This means there will be inconsistency in the relationships that are coded. This can be particularly problematic when trying to aggregate information from across sources for analysis.
- There will be situations where the attributes of a single concept bead are not well matched to the needs of a particular project. There needs to be rapid, responsive processes to discuss whether the common concept bead should be updated, or whether the project is given an exception. These processes are covered in more detail in Chapter 5, "Governance."

Example Resolved

GKDMR Travel begins its definition of a common information model by defining a collection of common concepts that are used repeatedly by its developers. Figure 3.3 shows a selection of these concepts.

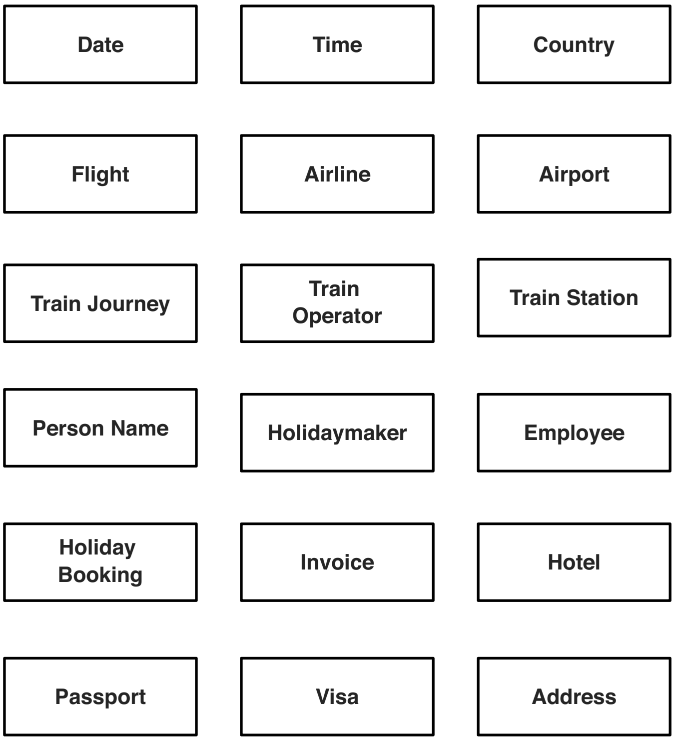


Figure 3.3 A selection of concepts for GKDMR Travel's common information model

Each concept is then expanded with a textual description, a logical model definition, and an optional number of physical implementation examples. Figure 3.4 expands the address concept to show these different definitions.

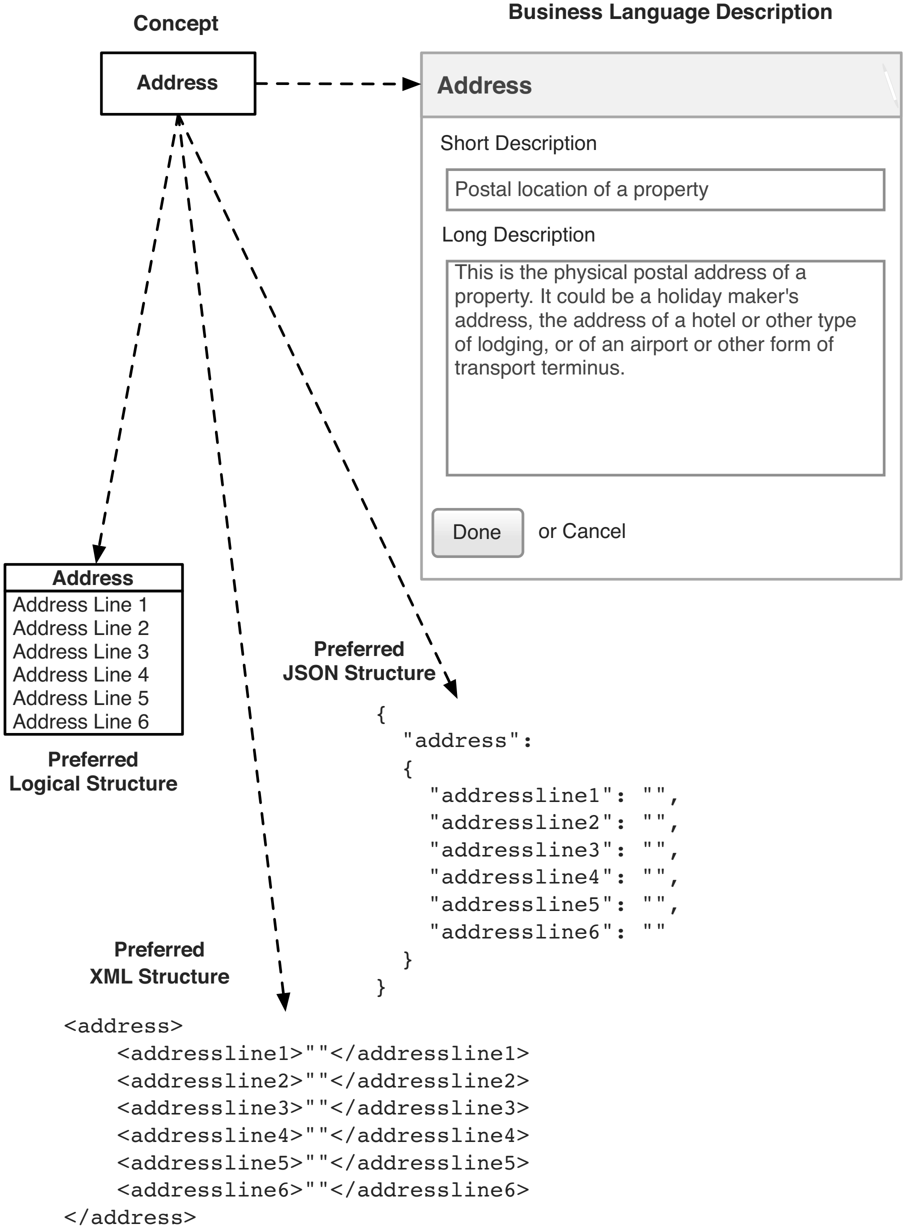


Figure 3.4 Concept bead definition for Address

The developers can download the standard formats directly into their workbench. The back-end applications that they need to call to connect to the business applications use similar structures. The developers who use the standard formats are far more productive than those who do not.

Known Uses

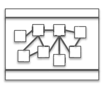
The AS 4590-2006 standard from Standards Australia is an example of a concept beads model that is a technical standard for addresses.

Systems of Engagement–based "born on the web" companies may benefit from this as their starting point.

Related Patterns

If the relationships between concepts are important, and benefit exists in having consistency in how they are represented, then consider either the continuous fabric pattern or the encapsulated view pattern.

Continuous Fabric



Context

An organization is looking to improve the methods it uses to develop and integrate IT systems into its operations.

Problem

An organization needs to integrate related information from a wide range of information sources.

The operation of an organization is typically supported by multiple applications, each supporting a unique aspect of the business. To understand how the organization is performing as an aggregate, pulling information from each of these applications and consolidating it in a single repository is necessary. Often the information from different applications is related and needs to be linked together in a consistent manner. However, each application supports different data structures, identifiers (keys), quality rules, terminology, precision, and currency (timeliness).

How should the aggregated view of this data be represented? These definitions should cover the concepts of the business and the relationships between them. It must include historical detail for analytical mining and up-to-date information for understanding the current state of the business.

Example

GKDMR Travel wants to offer real-time alerts and actions for customers when they are traveling and for the staff who are supporting them. Some of these alerts will be triggered by situations that develop while the customers are traveling, and some actions come from predictive analytics models that use historical information to predict the next best action the company should take to improve the service to a particular customer.

The company will need to create a repository of information that has historical and current information linked together and combined with information from external sources such as weather and location websites, customer comments, and other feedback, along with products and services from business partners.

Forces

- Many people find large data models intimidating and difficult to understand.
- The relationships between concepts can be as important to the business as the details of the concepts themselves.

Solution

Create a common information model that defines the meaning, structure, and relationships between all the core concepts.

A continuous fabric common information model is one where all the concepts are linked together into a continuous network structure. It is built by defining the concepts that need to be represented (in a similar way to the concept beads), and then relationships are added that link the concepts together in meaningful ways. See Figure 3.5.

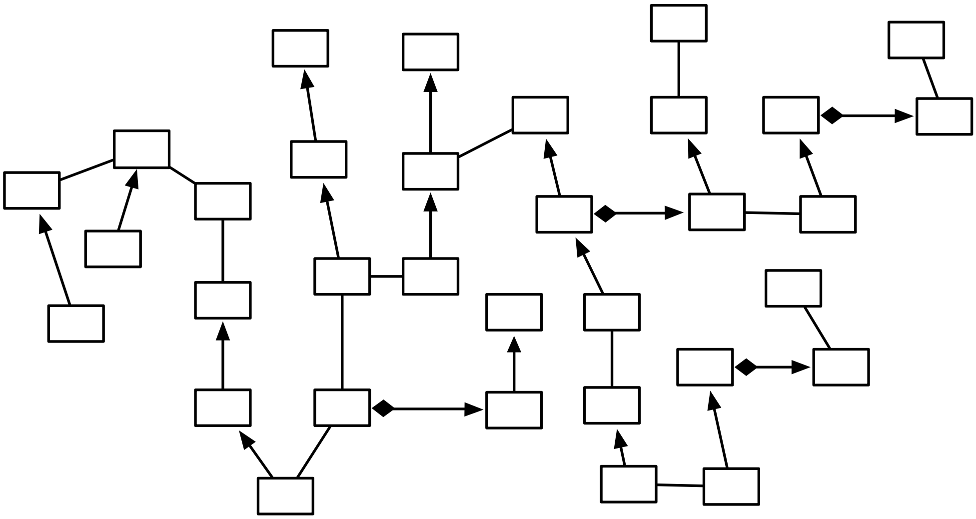


Figure 3.5 Structure of a continuous fabric common information model

The result is a model that describes a broad landscape of information for the organization. Subsets of a continuous fabric model are typically used in a project. The developer selects the concepts and relationships that are of interest and just uses that portion of the model.

Consequences

Benefits:

- The continuous fabric common information model creates consistency in how concepts are linked together. This creates a deeper level of consistency in the use of information. For many businesses, much of the value of information comes from these relationships. For example, knowing which customers use and buy particular products is more valuable than just having a list of customers and a list of products. The continuous fabric model defines where the valuable relationships are and how they should be represented.

Liabilities:

- This pattern takes a greater level of maturity in the organization's governance and willingness to share information across lines of business.

- The continuous fabric model is typically very rich in object relationships and will require significant rationalization and transformation before it can depict a physical artifact that can be incorporated in an application. The process will involve careful consideration of which aspects of the model are in scope for the project.
- Continuous fabric models can take a long time to develop. Skilled modelers who may not be members of a project team build them. As a result there can be knowledge disconnect between the team owning the model and the teams using it in projects.

Example Resolved

The analysts at GKDMR Travel create a model that describes all the concepts that could affect a customer's trip, including the causes and impacts and how they relate to each of the aspects of the customer's travel plans, insurance, and other related products they have purchased. These are linked together to create a single linked structure. This structure is used as the basis for the company's analytical repository supporting proactive customer service. Figure 3.6 shows a small part of this model.

Figure 3.6 is a conceptual model that uses the UML notation because this was easiest to use with the company's business users. After the model is accepted, it can be translated into an Entity-Relationship (E-R) logical model as part of the design process for the analytical repository.

Known Uses

Continuous fabric models are often used to describe repositories that consolidate information from many sources. Examples of this are data warehouses and master data management systems.

Continuous fabric models are also often used to describe the concepts in a specific industry. The SID model as published by the TeleManagement Forum (tmForum) is a good example.[3]

3. The Information Framework (SID) is a reference model and common vocabulary for the information required to implement the Business Process Framework (eTOM) processes. See http://www.tmforum.org/InformationFramework/1684/home.html for more information.

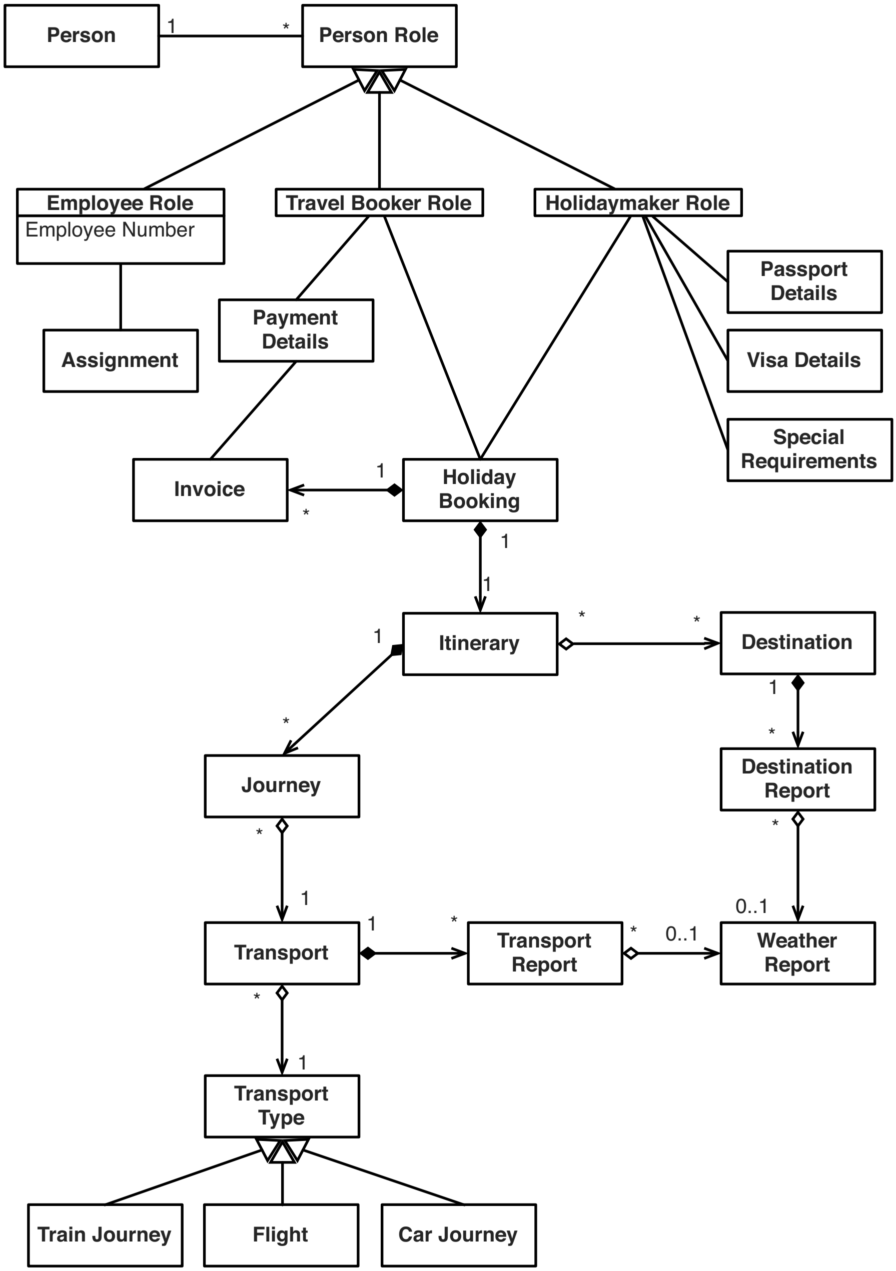


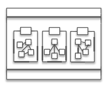
Figure 3.6 Fragment of a continuous fabric conceptual model

Related Patterns

The concept beads pattern provides a starting point for creating a continuous fabric model because it contains the core entities that will make up the model.

A continuous fabric model can become large and requires considerable effort to learn. The encapsulated view pattern that follows is an approach that breaks up the model into smaller pieces primarily for use as parameters (message structures) in information service definitions. However, it can also be used to break up a large continuous fabric model into subject areas.

Encapsulated Views



Context

An organization is looking to improve the methods it uses to develop and integrate IT systems into its operations.

Problem

An organization needs common definitions to exchange information for multiple purposes.

Mature organizations typically have many existing applications, each supporting a particular business function. Within an application, valuable data and capability is often locked within an individual application's unique data structures and interface styles. When the business wants to integrate these applications together so the information and functions they contain can be shared, it must reconcile the different data structures and naming conventions.

This information sharing is typically achieved through integration interfaces called information services or application programming interfaces (APIs). Each interface describes a number of operations that enable the caller to perform functions and create, update, retrieve, and delete information. Each operation typically has a set of parameters or structures to pass information to the operation and a structure to return the results.

What is the best way to structure a common information model to help developers create consistent interfaces?

Example

GKDMR Travel wants to provide a consistent set of business interfaces that enable mobile and web applications to call their business applications to request details of a customer and to exchange information about holiday packages, bookings, and payments. Specifically, it wants the representation of common entities such as holidaymaker and booking to be consistent wherever they appear in the business interfaces. The business interface implementations will then map these common structures to the actual structure used by the application being called.

Forces

- Many concepts are related and decisions need to be made on how these relationships will be represented.
- In creating these common definitions, choices need to be made on terminology used in the business interface as well as data structure and the valid values for each field.
- Documenting how each of the actual application interfaces maps to the common model is necessary.
- The information needed to support a particular business interface operation may need to come from multiple applications.

Solution

Create a common information model that defines small clusters of related concepts that can be used as structures for exchanging information.

Each of the clusters of concepts is independent of one another with relationships to concepts in different clusters being defined as holding the identifier of the referenced concept rather than a link to the type. This is illustrated in Figure 3.7.

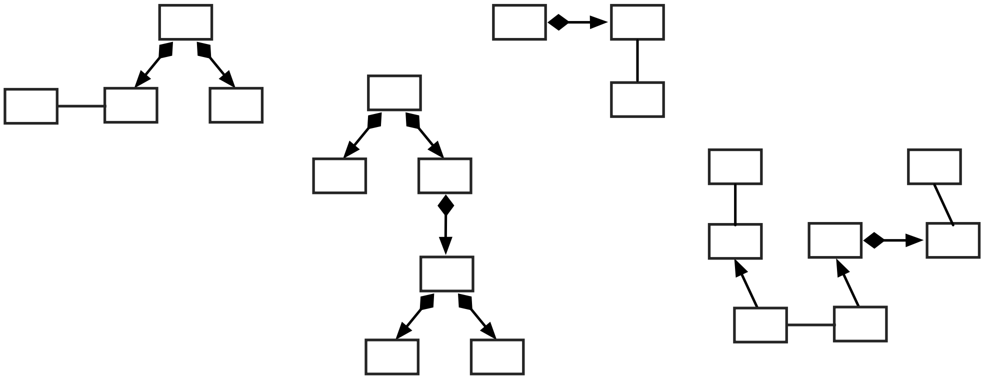


Figure 3.7 Structure of an encapsulated views common information model

A key characteristic of this pattern is that the encapsulation provides a means of restricting access by an interface to a partition of the model rather than the entire model, which is a valuable step toward defining service or message payload structures in a consistent manner.

Consequences

Benefits:

- This pattern provides a common information model that supports a service-oriented approach to integration. The data is modeled in discrete units that can be used directly in interface definitions as request or response parameters.
- This pattern is particularly important when integrating existing components because it seeks to reconcile the inevitable difference between these components.

Liabilities:

- The same concept may need multiple definitions so it can appear in messages as follows:
  - A small number of attributes to identify an instance of a concept. This is used on find queries, or to represent a link to the instance.
  - A summary of the concept, for reports or to populate a menu or table.
  - A complete definition of the concept when all details are required.

Figure 3.8 shows examples of variations of models of the person profile concept. **Person Profile Id** just includes the identifier and is used for passing a reference to an object.

Person Profile Locator defines the attributes used to query for a particular Person Profile. **Person Profile Summary** includes the core attributes of a Person Profile, and the full details are described in the **Person Profile** structure.

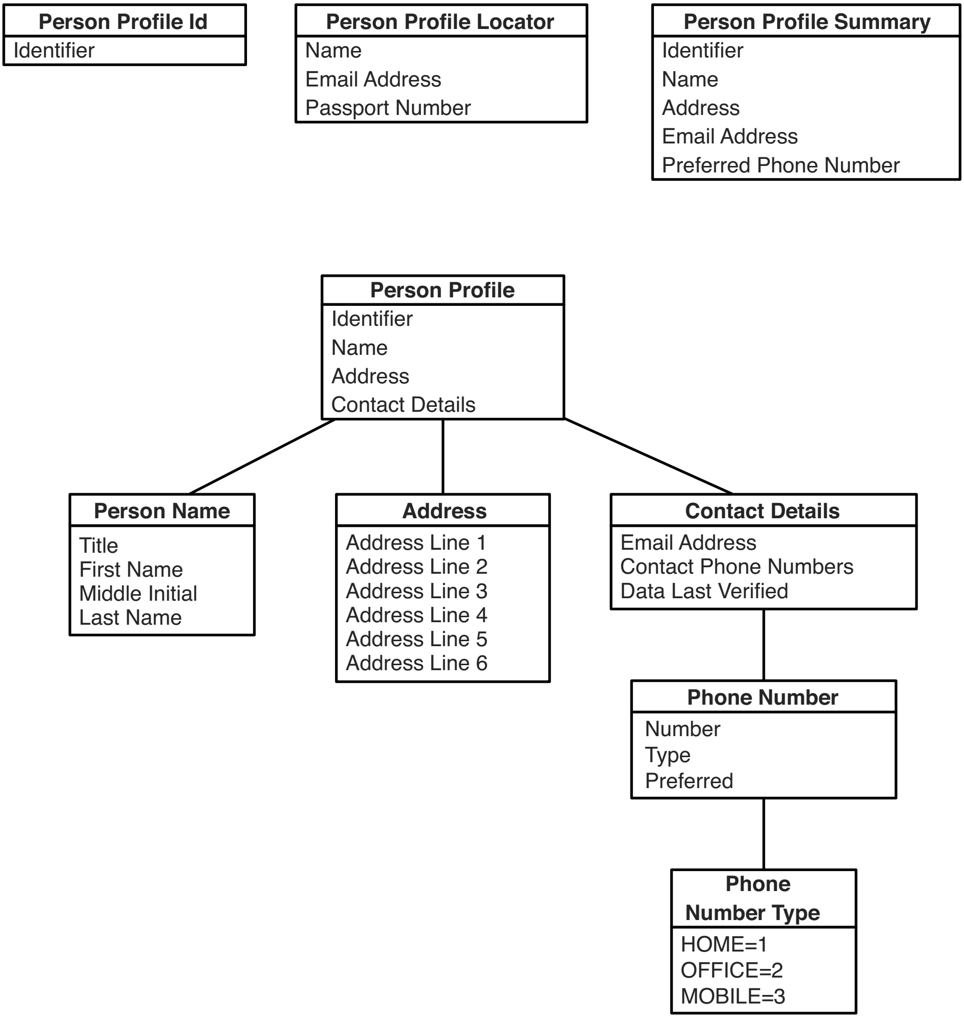


Figure 3.8 Variations on the model of Person Profile

Defining theses variations simplifies the use of the common definitions, and consequently increases the consistency in which they are used.

Example Resolved

GKDMR Travel creates encapsulated view models for the following concepts:

- Holidaymaker
- Customer
- Holiday Package
- Booking
- Itinerary
- Invoice
- Payment

Each of these views includes multiple variations of the model for each concept to cover references, summaries, and queries, as well as the complete definition of the concept.

Known Uses

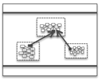
OMG's Common Object Request Broker (CORBA) standards defined a set of encapsulated views that heterogeneous systems can use to integrate their operations. See http://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture and http://www.corba.org.

For the travel industry, the OpenTravel Alliance provides standardized XML messages for exchanging information between organizations working in the travel industry, so this seemed a reasonable starting point. See http://www.opentravel.org.

Related Patterns

The concept beads pattern provides an excellent starting point for encapsulated views.

Unifying Context



Context

An organization is looking to improve the methods it uses to develop and integrate IT systems into its operations.

Problem

An organization needs to unify operations that have been independent in the past.

When two organizations integrate either by merger or partnership, they often discover there are at least two ways of doing anything and a confusing inconsistency in the terminology used by each of the formerly independent units. If the integration is to reap its intended benefits, the teams need to quickly reconcile their terminology and integrate/rationalize their systems.

These common definitions include terminology, concepts, policies, and processes.

Example

GKDMR Travel recently created a partnership with another holiday company that wanted to advertise GKDMR Travel's holidays in its brochure. This broadened the potential audience for GKDMR Travel's holidays but meant it needed to use the interfaces provided by its business partner. Although both organizations are in the travel industry, there are considerable differences in the terminology that each used to describe its customers and services.

Forces

- The most confusion caused when two different sources of models are brought together is when the same term is used in each, but it has a different meaning.
- Often the level of granularity and the style of modeling can be different when two models from different sources are compared. This means mapping directly between the models is hard.

Solution

Create a common information model that maps existing terminology and definitions to a set of definitions that represents a canonical view of the subject area.

Models created by different groups of people tend to use different terminology, levels of granularity, and patterns within the structures. As such, mapping between them directly is often very difficult. The unifying context common information model uses a semantic model to act as a lingua franca to show how the concepts in two models relate to one another. Figure 3.9 shows an example of this structure. The concepts from the semantic model are shown in gray (labeled "Linking Model Concept"). Each of these concepts has a dotted line relationship to each of the elements in the models being mapped that has the same meaning. Some of the concepts in the semantic model may only be present in one model, or appear in multiple places in the models. Not only does this modeling effort create understanding of how two models relate, but it also shows where there are gaps, duplication, and discrepancies in either model.

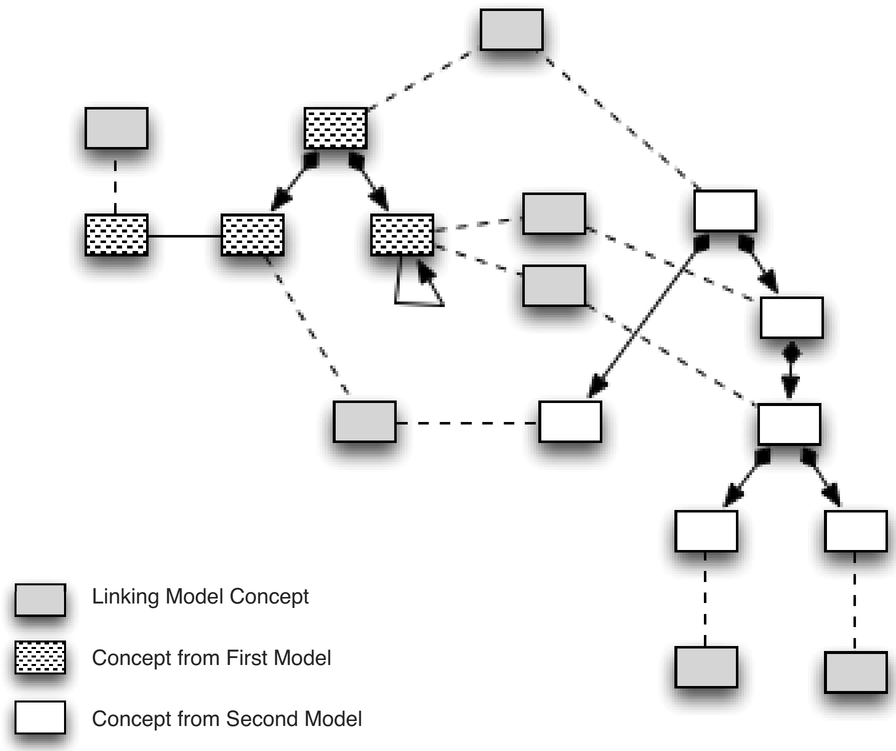


Figure 3.9 Existing models linked by the unifying concepts

Consequences

Benefits:

- The unifying context common information model provides an important communication vehicle when two teams are trying to work together for the first time.
- It highlights areas were mapping and normalization would be required and where mediations and transformations will need to be implemented.

Liabilities:

- If either model is changing, then the mapping needs constant maintenance.
- The mapping model may need to include semantic relationships, such as when generalizations and specializations of a particular concept of the models being mapped are at different levels of granularity.
- Although this pattern helps in understanding, it does not in itself address the structural and value-based issues that may need to be resolved at the logical and physical models.

Example Resolved

GKDMR Travel and its business partner agreed to base the canonical definition on the terminology used by the OpenTravel Alliance.[4] This organization provides standardized XML messages for exchanging information between organizations working in the travel industry, so this seemed a reasonable starting point. The team extracted all the key concepts from the OpenTravel's standard message sets and modeled its key attributes and relationships. These models were verified with the business partner. Then each organization mapped its messages to the OpenTravel messages. The result was a clear definition of how the terminology and APIs defined by the business partner maps to GKDMR Travel's definitions. Figure 3.10 shows an example mapping.

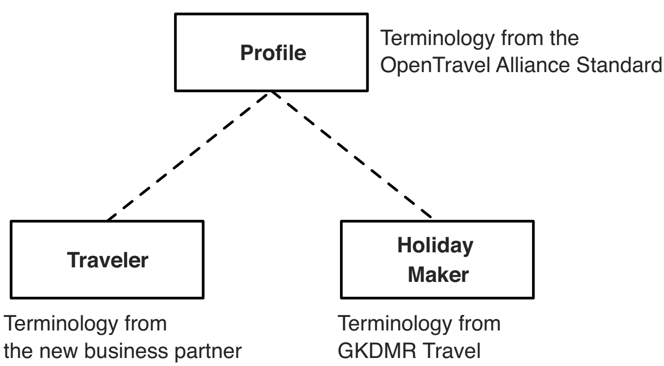


Figure 3.10 Example of a mapping between two model concepts using a canonical definition

4. OpenTravel Alliance, http://www.opentravel.org

With this initial mapping in place, it is then possible to verify that the valid values for the equivalent terms are compatible, and to reveal where there are missing concepts in GKDMR Travel's common information model.

Known Uses

- In a service-oriented architecture implementation, this style of common information model can be used to show the relationship between a canonical message format and the application-specific message formats used by each of the service providers and service requestors.
- For organizations involved in mergers and acquisitions, this style of common information model can be used to show the relationships between terminology, processes, and systems from each of the original organizations.

Related Patterns

The concept beads pattern or continuous fabric pattern provides a good basis for the canonical form of the model.

Combining the Patterns

The structural patterns for a common information model are designed to be complementary. A common information model for an organization that is performing a significant transformation is eventually going to include all the structural patterns in its contents, although these are likely to be added over time.

The concept beads are always a good starting point. Then, depending on your business priorities, you may consider adding

- Encapsulated views for a systems integration or mobile project
- Continuous fabric for a data warehouse or big data project
- Unifying context to link together different divisions within an organization—or to link internal definitions with industry standards

Whichever pattern you use, the common information model should express just the aspects of information meaning, structure, and use that the stakeholders want to govern. It should make it clear where the consumers need to follow the guidelines and where they are free to innovate.

Summary

In this chapter we have covered the common structural patterns for a common information model. These patterns are

- **Common information model:** Generic pattern for a common information model.
- **Concept beads:** Simple modeled elements for teams with low experience in using a common information model; it's a good starting point for any common information model initiative.
- **Continuous fabric:** Includes models that show how the information managed by the organization is linked together to form a single linked structure of concepts.
- **Encapsulated views:** Creates multiple small collections of linked and related concepts that can be used for request and response parameters on interfaces.
- **Unified context:** Used to link information models from different sources or organizations. It focuses on showing where concepts from one set of models map to synonym concepts in another set of models.

Chapter 4, "Modeling Best Practices," takes all this information to the next level and describes the process of creating, managing, and consuming artifacts for the common information model.

This page intentionally left blank

This page intentionally left blank

Index

H

I

N

O

P

Q - R

S

W - Z