THE PRENTICE HALL SERVICE TECHNOLOGY SERIES FROM THOMAS ERL

"This book illuminates the connection of the two domains—SOA and REST—in a manner that is concrete and practical, providing concise application to everyday architectural challenges. Fantastic!"
—Ryan Frazier, Technology Strategist, Microsoft

"…a tour de force that elegantly applies REST principles to the industry-standard SOA framework described in prior titles in this series…. This book is a must-read for anyone developing REST services."
—Dave Slotnick, Enterprise Architect, Rackspace Hosting

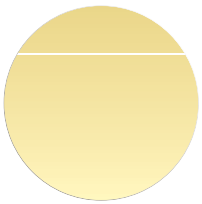
"This book undoubtedly will help SOA to reap the benefits from the main value propositions of Web architecture…."
—Dr. Erik Wilde, Architect, EMC Corporation

SOA with REST

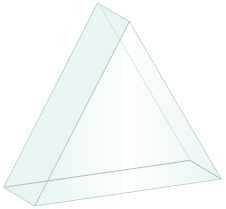
Principles, Patterns & Constraints for Building Enterprise Solutions with REST

Co-authored and Edited by Thomas Erl, World's Top-Selling SOA Author
Co-authored by Benjamin Carlyle, Cesare Pautasso, Raj Balasubramanian
Foreword by Stefan Tilkov

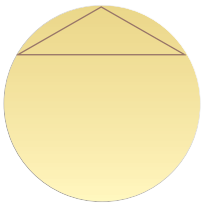
PRENTICE HALL
ServiceTech PRESS



service contract
(chorded circle notation)

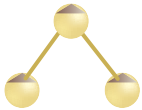


uniform
contract



service contract accessed
via a uniform contract
(chorded circle notation)

REST service
(labeled)



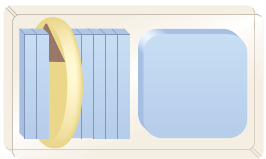
REST service
composition

component
or program

decoupled
service
contract

decoupled service
contract accessed
via a uniform
contract

service
agent



REST service using
uniform contract

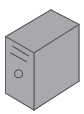
process logic

WSDL
definition

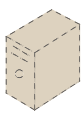
XML Schema
definition

general machine
processable
document

human-readable
document
or content



physical
server

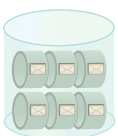


virtual
server

firewall

message

security element
or locked resource



message
queue

repository
or registry

actively
processing

process step or
project/lifecycle stage

Internet

cloud

zone or
region

conflict
symbol

transition
arrow

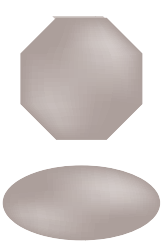
human
or role

client
workstation

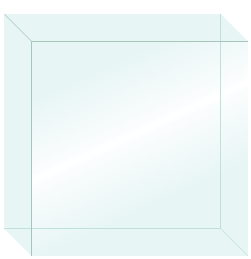
user
interface

mobile
device

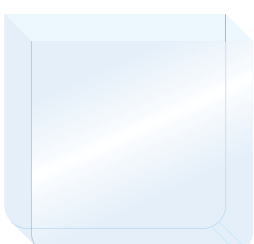
product
or system



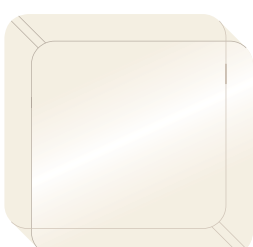
symbols used in conceptual
relationship diagrams



general physical
boundary



service inventory
boundary



service
boundary

SUMMARY OF KEY POINTS

- The modeling of the uniform contract is closely associated with the service inventory analysis project stage.
- The initial definition of the uniform contract should be carried out prior to REST service contract design because individual REST service contracts will be dependent on the uniform contract features and limitations.
- A straight-forward means of incorporating the task of modeling the uniform contract with the service inventory analysis lifecycle is to combine it with the existing *Define Technology Architecture* step or to establish it as an independent step.

9.2 REST Service Modeling

The need to build a particular service can originate from various sources and under different circumstances.

For example:

- A service might be identified as being necessary based on a service inventory blueprint that maps out required functions and allocates them to functional contexts that form the basis of existing and future services.
- A service might be identified through change requests when it is determined that new functionality is needed and that this new functionality does not belong in any existing functional service contexts.
- A service might be identified through the analysis of legacy systems not yet encapsulated within the scope of a service inventory.

In any case, the service candidate is typically defined with whatever service capability candidates result from the service-oriented analysis process, during which we carry out a service modeling process that decomposes business process logic into granular actions that form the basis of service capability candidates.

The incorporation of resources and uniform contract features add new dimensions to service modeling. When we are aware that a given service candidate is being modeled

specifically for a REST implementation, we can take these considerations into account by extending the service modeling process to include steps to better shape the service candidate as a basis for a REST service contract.

The upcoming *REST Service Modeling Process* section documents a generic service modeling process optimized for REST services. In preparation for the step-by-step coverage of this process, the next two sections highlight some key characteristics of REST service candidate definition.

REST Service Capability Granularity

When actions are defined at this stage, they are considered fine-grained in that each action is clearly distinguished with a specific purpose. However, within the scope of that purpose they can often still be somewhat vague and can easily encompass a range of possible variations.

Defining conceptual service candidates using this level of action granularity is common with mainstream service modeling approaches. It has proven sufficient for SOAP-based Web services because service capabilities that need to support variations of functionality can still be effectively mapped to WSDL-based operations capable of handling a range of input and output parameters.

With REST service contracts, service capabilities are required to incorporate methods (and media types) defined by an overarching uniform contract. As already discussed in the preceding section, the uniform contract for a given service inventory can be modeled alongside and in collaboration with the modeling of service candidates, as long as we know in advance that REST will act as the primary service implementation medium.

Whereas a WSDL-based service contract can incorporate custom parameter lists and other service-specific features, REST puts an upper bound on the granularity of message exchanges at the level of the most complex or most general purpose method and media type. This may, in some cases, lead to the need to define finer-grained service capabilities.

Figure 9.3 highlights the difference between a service candidate modeled in an implementation-neutral manner, versus one modeled specifically for the REST service implementation medium.

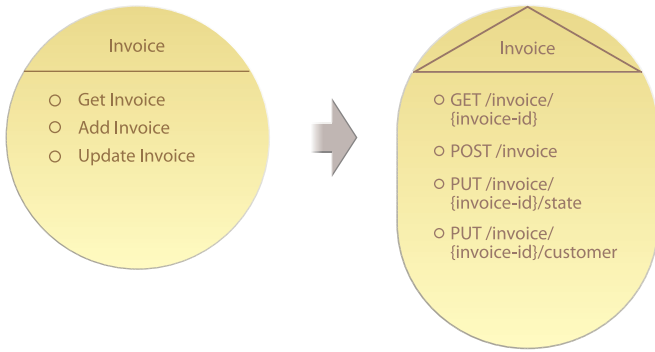


Figure 9.3
A REST service candidate can be modeled specifically to incorporate uniform contract characteristics. The Update Invoice service capability candidate is split into two variations—one that updates the invoice state value and another that updates the invoice customer value.

Resources vs. Entities

Part of the upcoming REST service modeling process explores the identification of resource candidates. It is through the definition of these resource candidates that we begin to introduce a Web-centric view of a service inventory. As discussed previously in Chapter 6, resources represent the "things" that need to be accessed and processed by service consumers.

What we are also interested in establishing during the service-oriented analysis stage is the encapsulation of entity logic, as explained in the coverage of the entity service model provided in Chapter 4. As with resources, entities also often represent "things" that need to be accessed and processed by service consumers.

What then is the difference between a resource and an entity? To understand REST service modeling, we need to clearly understand this distinction:

- Entities are business-centric and are derived from enterprise business models, such as entity relationship diagrams, logical data models, and ontologies.
- Resources can be business-centric or non-business-centric. A resource is any given "thing" associated with the business automation logic enabled by the service inventory.
- Entities are commonly limited to business artifacts and documents, such as invoices, claims, customers, etc.

- Some entities are more coarse-grained than others. Some entities can encapsulate others. For example, an invoice entity may encapsulate an invoice detail entity.
- Resources can also vary in granularity, but are often fine-grained. It is less common to have formally defined coarse-grained resources that encapsulate fine-grained resources.
- All entities can relate to or be based on resources. Not all resources can be associated with entities because some resources are non-business-centric.

The extent to which we need to formalize the mapping between business-centric resources and entities is up to us. In the upcoming REST service modeling process there are steps that encourage us to define and standardize resources as part of the service inventory blueprint so that we gain a better understanding of how and where resources need to be consumed.

From a pure modeling perspective we are further encouraged to relate business-centric resources to business entities so that we maintain a constant alignment with how business-centric artifacts and documents exist within our business. This perspective is especially valuable as the business and its automation requirements continue to evolve over time.

REST Service Modeling Process

We now introduce a formal process for modeling REST services. This section is divided into a series of parts that describe each process step. About half of the step descriptions are further supplemented with a string of case study examples that demonstrate how a set of service candidates are defined and refined to form a simple service composition candidate.

Figure 9.4 summarizes the process by providing a variation of the original mainstream service modeling process (first introduced in Chapter 8) containing modifications and additional steps incorporated in support of REST service modeling.

NOTE

As with any MSOAM process or model, the REST service modeling process should be considered a generic approach that can be further customized to incorporate preferences and requirements specific to the organization and its IT enterprise. The purpose of this process is to provide a starting point that highlights key considerations and steps commonly required to effectively model service candidates in support of REST implementation.

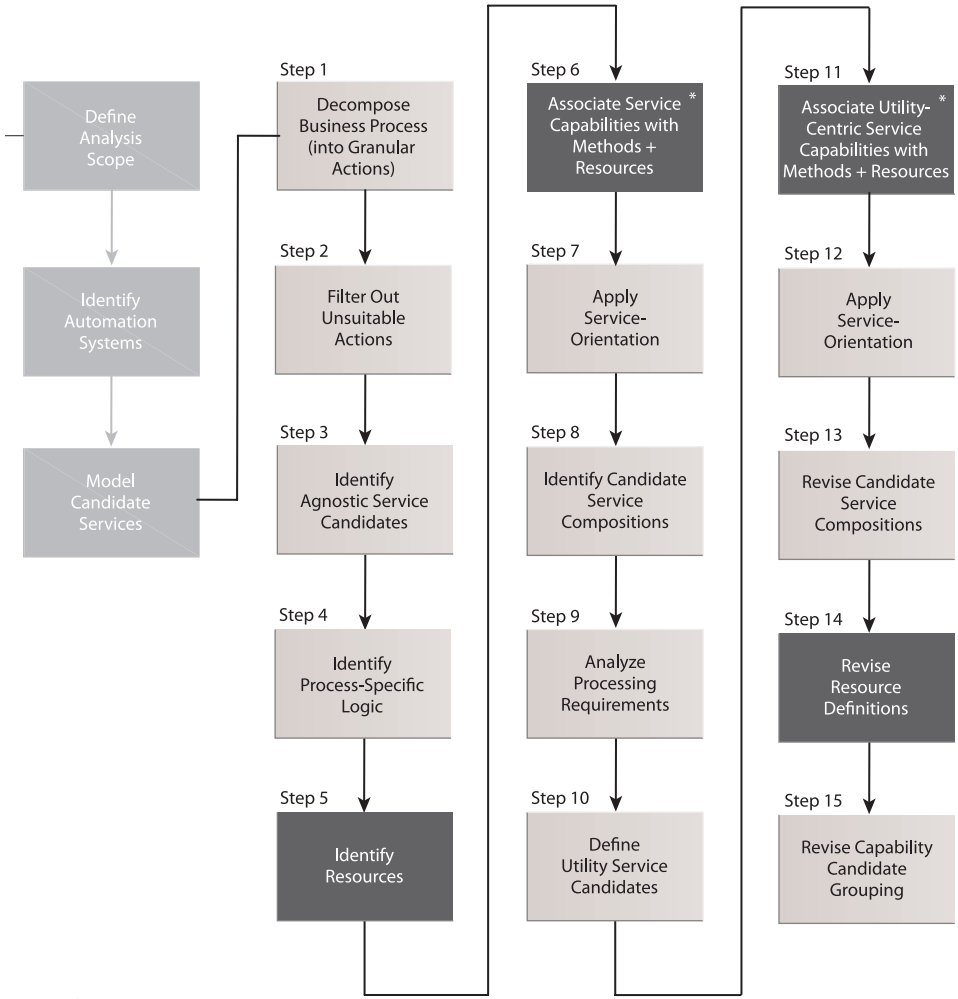


Figure 9.4

A RESTful version of the MSOAM service modeling process extended to include new steps (highlighted with dark grey) in support of modeling services specifically for REST implementation. The asterisk on the top-right of Steps 6 and 11 indicates a connection with the Model Uniform Contract task from the service inventory analysis cycle (as previously explained in the *Uniform Contract Modeling and REST Service Inventory Modeling* section).

Let's take a closer look at the individual steps of the REST service modeling process. We'll begin with a case study example that establishes the business process logic that will act as our primary input into the process steps.

CASE STUDY EXAMPLE

Existing MUA charter agreements with partner schools explicitly refer to the need to acknowledge individual academic achievements. This makes the correct conferral of awards important to the reputation of MUA and its elite students.

MUA assembles a service modeling team comprised of SOA architects, SOA analysts, and business analysts. The team begins with a REST service modeling process for the Student Achievement Award Conferral business process. As detailed in Figure 9.5, this business process logic represents the procedures followed for the assessment, conference, and rejection of individual achievement award applications submitted by students. An application that is approved results in the conferral of the achievement award and a notification of the conferral to the student. An application that is rejected results in a notification of the rejection to the student.

Step 1: Decompose Business Process (into Granular Actions)

We begin by taking the documented business process and breaking it down into a series of granular process steps. This requires further analysis of the process logic, during which we attempt to decompose the business process into a set of individual granular actions.

CASE STUDY EXAMPLE

The original Student Award Conferral business process is broken down into the following granular actions:

- Initiate Conferral Application
- Get Event Details
- Verify Event Details
- If Event is Invalid or Ineligible for Award, Cancel Process
- Get Award Details
- Get Student Transcript
- Verify Student Transcript Qualifies for Award Based on Award Conferral Rules
- If Student Transcript does not Qualify, Initiate Rejection