



# Web Service Implementation Methodology – Rational Unified Process (Example)

Working Draft 02, 23 December 2004

**Document identifier:**

fwsim-1.0-RUPExample-doc-wd-01.doc

**Location:**

[http://www.oasis-open.org/committees/documents.php?wg\\_abbrev=fwsim](http://www.oasis-open.org/committees/documents.php?wg_abbrev=fwsim)

**Editors:**

Lai Peng CHAN, Singapore Institute of Manufacturing Technology  
<[lpchan@SIMTech.a-star.edu.sg](mailto:lpchan@SIMTech.a-star.edu.sg)>  
Chai Hong ANG, Singapore Institute of Manufacturing Technology  
<[chang@SIMTech.a-star.edu.sg](mailto:chang@SIMTech.a-star.edu.sg)>

**Contributors:**

Eng Wah LEE, Singapore Institute of Manufacturing Technology  
<[ewlee@SIMTech.a-star.edu.sg](mailto:ewlee@SIMTech.a-star.edu.sg)>  
Puay Siew TAN, Singapore Institute of Manufacturing Technology  
<[pstan@SIMTech.a-star.edu.sg](mailto:pstan@SIMTech.a-star.edu.sg)>  
Yushi CHENG, Singapore Institute of Manufacturing Technology  
<[ycheng@SIMTech.a-star.edu.sg](mailto:ycheng@SIMTech.a-star.edu.sg)>  
Xingjian XU, Singapore Institute of Manufacturing Technology  
<[xjxu@SIMTech.a-star.edu.sg](mailto:xjxu@SIMTech.a-star.edu.sg)>  
Zun Liang YIN, Singapore Institute of Manufacturing Technology  
<[zlyin@SIMTech.a-star.edu.sg](mailto:zlyin@SIMTech.a-star.edu.sg)>  
Andy TAN, individual, <[andytan@intrinix.net](mailto:andytan@intrinix.net)>  
Roberto PASCUAL, The Infocomm Development Authority of Singapore  
<[Roberto\\_B\\_Pascual@ida.gov.sg](mailto:Roberto_B_Pascual@ida.gov.sg)>  
JAGDIP Talla, CrimsonLogic Pte Ltd <[jagdip@crimsonlogic.com](mailto:jagdip@crimsonlogic.com)>  
RAVI SHANKAR Narayanan Nair, CrimsonLogic Pte Ltd  
<[ravishankar@crimsonlogic.com](mailto:ravishankar@crimsonlogic.com)>  
Marc HAINES, individual <[mhaines@uwm.edu](mailto:mhaines@uwm.edu)>

**Abstract:**

This document specifies Web service specific activities in a Web service implementation methodology and illustrates the approach to incorporate these activities into an existing agile software development methodology.

**Status:**

This document is updated periodically on no particular schedule. Send comments to the editor.  
Committee members should send comments on this specification to the [fwsim-ims@lists.oasis-open.org](mailto:fwsim-ims@lists.oasis-open.org) list. Others should subscribe to and send comments to the [fwsim-comment@lists.oasis-open.org](mailto:fwsim-comment@lists.oasis-open.org) list. To subscribe, send an email message to

45 [fws-comment-request@lists.oasis-open.org](mailto:fws-comment-request@lists.oasis-open.org) with the word "subscribe" as the body of  
46 the message.  
47 For information on whether any patents have been disclosed that may be essential to  
48 implementing this specification, and any offers of patent licensing terms, please refer  
49 to the Intellectual Property Rights section of the FWSI TC web page  
50 (<http://www.oasis-open.org/committees/fws/>).

## Table of Contents

52	1	Case Example(s) of Applying the Implementation Methodology .....	9
53	1.1	Rational Unified Process (Example) .....	9
54	1.1.1	Overview.....	9
55	1.1.2	Approach .....	11
56	1.1.2.1	Discipline: Requirements .....	12
57	1.1.2.1.1	Workflow Detail: Analyze the Problem .....	12
58	1.1.2.1.1.1	Activity: Capture a Common Vocabulary .....	12
59	1.1.2.1.1.1.1	Role .....	12
60	1.1.2.1.1.1.2	Artifact.....	13
61	1.1.2.1.1.2	Activity: Develop Vision (across all Web services) .....	13
62	1.1.2.1.1.2.1	Role .....	13
63	1.1.2.1.1.2.2	Artifact.....	13
64	1.1.2.1.2	Workflow Detail: Understand Stakeholder Needs .....	13
65	1.1.2.1.2.1	Activity: Elicit Needs .....	13
66	1.1.2.1.2.1.1	Role .....	13
67	1.1.2.1.2.1.2	Artifact.....	13
68	1.1.2.1.2.2	Activity: Develop Vision .....	13
69	1.1.2.1.2.2.1	Role .....	13
70	1.1.2.1.2.2.2	Artifact.....	13
71	1.1.2.1.2.3	Activity: Manage Dependencies .....	13
72	1.1.2.1.2.3.1	Role .....	14
73	1.1.2.1.2.3.2	Artifact.....	14
74	1.1.2.1.2.4	Activity: Find Actors and Use Cases (per Web service) .....	14
75	1.1.2.1.2.4.1	Role .....	14
76	1.1.2.1.2.4.2	Artifact.....	14
77	1.1.2.1.3	Workflow Detail: Define the System .....	14
78	1.1.2.1.3.1	Activity: Find Actors and Use Cases (per Web service) .....	14
79	1.1.2.1.3.1.1	Role .....	14
80	1.1.2.1.3.1.2	Artifact.....	14
81	1.1.2.1.4	Workflow Detail: Manage the Scope of the System .....	14
82	1.1.2.1.4.1	Activity: Prioritise the Use Cases (per Web service).....	14
83	1.1.2.1.4.1.1	Role .....	14
84	1.1.2.1.4.1.2	Artifact.....	14
85	1.1.2.1.5	Workflow Detail: Refine the System Definition .....	15
86	1.1.2.1.5.1	Activity: Detail a Use Case .....	15
87	1.1.2.1.5.1.1	Role .....	15
88	1.1.2.1.5.1.2	Artifact.....	15
89	1.1.2.1.5.2	Activity: Detail the Software Requirements.....	15
90	1.1.2.1.5.2.1	Role .....	15
91	1.1.2.1.5.2.2	Artifact.....	15
92	1.1.2.2	Discipline: Analysis and Design .....	16
93	1.1.2.2.1	Workflow Detail: Define a Candidate Architecture (for each Web service).....	16

94	1.1.2.2.1.1 Activity: Architectural Analysis.....	16
95	1.1.2.2.1.1.1 Role.....	16
96	1.1.2.2.1.1.2 Artifact.....	16
97	1.1.2.2.2 Workflow Detail: Analyse Behaviour (for each use case).....	16
98	1.1.2.2.2.1 Activity: Use Case Analysis.....	16
99	1.1.2.2.2.1.1 Role.....	17
100	1.1.2.2.2.1.2 Artifact.....	17
101	1.1.2.2.3 Workflow Detail: Refine the Architecture (for each Web service).....	17
102	1.1.2.2.3.1 Activity: Identify Design Elements.....	17
103	1.1.2.2.3.1.1 Role.....	17
104	1.1.2.2.3.1.2 Artifact.....	17
105	1.1.2.2.3.2 Activity: Identify Design Mechanisms.....	17
106	1.1.2.2.3.2.1 Role.....	17
107	1.1.2.2.3.2.2 Artifact.....	17
108	1.1.2.2.4 Workflow Detail: Design Components.....	17
109	1.1.2.2.4.1 Activity: Use Case Design.....	17
110	1.1.2.2.4.1.1 Role.....	18
111	1.1.2.2.4.1.2 Artifact.....	18
112	1.1.2.2.4.2 Activity: Subsystem Design.....	18
113	1.1.2.2.4.2.1 Role.....	18
114	1.1.2.2.4.2.2 Artifact.....	18
115	1.1.2.2.4.3 Activity: Class Design.....	18
116	1.1.2.2.4.3.1 Role.....	18
117	1.1.2.2.4.3.2 Artifact.....	18
118	1.1.2.3 Discipline: Implementation.....	19
119	1.1.2.3.1 Workflow Detail: Structure the Implementation Model.....	19
120	1.1.2.3.1.1 Activity: Structure the Implementation Model.....	19
121	1.1.2.3.1.1.1 Role.....	19
122	1.1.2.3.1.1.2 Artifact.....	19
123	1.1.2.3.2 Workflow Detail: Implement Components.....	19
124	1.1.2.3.2.1 Activity: Implement Design Elements.....	19
125	1.1.2.3.2.1.1 Role.....	20
126	1.1.2.3.2.1.2 Artifact.....	20
127	1.1.2.3.3 Workflow Detail: Integrate Each Web Service.....	20
128	1.1.2.3.3.1 Activity: Integrate Subsystem.....	20
129	1.1.2.3.3.1.1 Role.....	20
130	1.1.2.3.3.1.2 Artifact.....	20
131	1.1.2.3.4 Workflow Detail: Manage the Scope of the System.....	20
132	1.1.2.3.4.1 Activity: Prioritise the Use Cases (per Web service).....	20
133	1.1.2.3.4.1.1 Role.....	20
134	1.1.2.3.4.1.2 Artifact.....	20
135	1.1.2.3.5 Workflow Detail: Refine the System Definition.....	20
136	1.1.2.3.5.1 Activity: Detail a Use Case.....	20
137	1.1.2.3.5.1.1 Role.....	20
138	1.1.2.3.5.1.2 Artifact.....	20

139	1.1.2.3.5.2 Activity: Detail the Software Requirements.....	21
140	1.1.2.3.5.2.1 Role .....	21
141	1.1.2.3.5.2.2 Artifact.....	21
142	1.1.2.4 Discipline: Testing.....	22
143	1.1.2.4.1 Workflow Detail: Define Mission.....	22
144	1.1.2.4.1.1 Activity: Identify Test Motivators .....	22
145	1.1.2.4.1.1.1 Role .....	23
146	1.1.2.4.1.1.2 Artifact.....	23
147	1.1.2.4.1.2 Activity: Agree on the Mission .....	23
148	1.1.2.4.1.2.1 Role .....	24
149	1.1.2.4.1.2.2 Artifact.....	24
150	1.1.2.4.1.3 Activity: Define Assessment and Traceability Needs.....	24
151	1.1.2.4.1.3.1 Role .....	24
152	1.1.2.4.1.3.2 Artifact.....	24
153	1.1.2.4.1.4 Activity: Define Test Approach .....	25
154	1.1.2.4.1.4.1 Role .....	25
155	1.1.2.4.1.4.2 Artifact.....	25
156	1.1.2.4.1.5 Activity: Identify Test Ideas.....	25
157	1.1.2.4.1.5.1 Role .....	26
158	1.1.2.4.1.5.2 Artifact.....	26
159	1.1.2.4.2 Workflow Detail: Define Test Bed .....	26
160	1.1.2.4.2.1 Activity: Identify Test Environment.....	26
161	1.1.2.4.2.1.1 Role .....	26
162	1.1.2.4.2.1.2 Artifact.....	26
163	1.1.2.4.2.2 Activity: Prepare H/W & S/W Infrastructure .....	26
164	1.1.2.4.2.2.1 Role .....	27
165	1.1.2.4.2.2.2 Artifact.....	27
166	1.1.2.4.2.3 Activity: Prepare Test Data Sets.....	27
167	1.1.2.4.2.3.1 Role .....	27
168	1.1.2.4.2.3.2 Artifact.....	27
169	1.1.2.4.3 Workflow Detail: Develop, Test and Evaluate .....	27
170	1.1.2.4.3.1 Activity: Define Test Details.....	27
171	1.1.2.4.3.1.1 Role .....	27
172	1.1.2.4.3.1.2 Artifact.....	27
173	1.1.2.4.3.2 Activity: Implement Test .....	28
174	1.1.2.4.3.2.1 Role .....	28
175	1.1.2.4.3.2.2 Artifact.....	28
176	1.1.2.4.3.3 Activity: Implement Test Suite .....	28
177	1.1.2.4.3.3.1 Role .....	28
178	1.1.2.4.3.3.2 Artifact.....	28
179	1.1.2.4.3.4 Activity: Execute Test Suite.....	28
180	1.1.2.4.3.4.1 Role .....	29
181	1.1.2.4.3.4.2 Artifact.....	29
182	1.1.2.4.3.5 Activity: Analyse Test Failures.....	29
183	1.1.2.4.3.5.1 Role .....	29

184	1.1.2.4.3.5.2	Artifact.....	29
185	1.1.2.4.3.6	Activity: Determine Test Results.....	29
186	1.1.2.4.3.6.1	Role .....	29
187	1.1.2.4.3.6.2	Artifact.....	29
188	1.1.2.4.4	Workflow Detail: Improve Test Assets.....	29
189	1.1.2.4.4.1	Activity: Define Test Approach (Refinement).....	29
190	1.1.2.4.4.1.1	Role .....	30
191	1.1.2.4.4.1.2	Artifact.....	30
192	1.1.2.4.4.2	Activity: Identify Test Ideas (Refinement) .....	30
193	1.1.2.4.4.2.1	Role .....	30
194	1.1.2.4.4.2.2	Artifact.....	30
195	1.1.2.4.4.3	Activity: Prepare Guidelines for the Project .....	30
196	1.1.2.4.4.3.1	Role .....	30
197	1.1.2.4.4.3.2	Artifact.....	30
198	1.1.2.5	Discipline: Deployment .....	31
199	1.1.2.5.1	Workflow Detail: Plan Deployment.....	31
200	1.1.2.5.1.1	Activity: Develop Deployment Plan.....	31
201	1.1.2.5.1.1.1	Role .....	31
202	1.1.2.5.1.1.2	Artifact.....	31
203	1.1.2.5.2	Workflow Detail: Develop Support Material.....	32
204	1.1.2.5.2.1	Activity: Develop Training Material .....	32
205	1.1.2.5.2.1.1	Role .....	32
206	1.1.2.5.2.1.2	Artifact.....	32
207	1.1.2.5.2.2	Activity: Develop Support Material.....	32
208	1.1.2.5.2.2.1	Role .....	32
209	1.1.2.5.2.2.2	Artifact.....	32
210	1.1.2.5.3	Workflow Detail: Produce Deployment Unit (Web service) .....	32
211	1.1.2.5.3.1	Activity: Write Release Notes .....	32
212	1.1.2.5.3.1.1	Role .....	32
213	1.1.2.5.3.1.2	Artifact.....	32
214	1.1.2.5.3.2	Activity: Develop Installation Artifacts .....	32
215	1.1.2.5.3.2.1	Role .....	32
216	1.1.2.5.3.2.2	Artifact.....	32
217	1.1.2.5.3.3	Activity: Create Deployment Unit (Web service).....	33
218	1.1.2.5.3.3.1	Role .....	33
219	1.1.2.5.3.3.2	Artifact.....	33
220	1.1.2.5.3.4	Activity: Deploy Web Service to identified app servers.....	33
221	1.1.2.5.3.4.1	Role .....	33
222	1.1.2.5.3.4.2	Artifact.....	33
223	1.1.2.5.3.5	Activity: Publish Web service [optional] .....	33
224	1.1.2.5.3.5.1	Role .....	33
225	1.1.2.5.3.5.2	Artifact.....	33
226	2	References .....	34
227	2.1	Normative .....	34
228	2.2	Non-Normative .....	34

229	Appendix A. Acknowledgments .....	35
230	Appendix B. Revision History.....	36
231	Appendix C. Notices.....	37

---

232

233

---

## List of Figures

234	Figure 3: The Rational Unified Process.....	10
235	Figure 4: The RUP development disciplines .....	12
236	Figure 5: Requirements Workflow.....	12
237	Figure 6: Analysis and Design Workflow.....	16
238	Figure 7: Implementation Workflow.....	19
239	Figure 8: Testing Workflow .....	22
240	Figure 9: Deployment Workflow.....	31

---

241

---

# 242 1 Case Example(s) of Applying the 243 Implementation Methodology

244  
245 The case example(s) aims to illustrate how an agile software methodology could be adapted  
246 to incorporate the Web services-specific activities described in the previous chapters.  
247

248 It is not the intention of this Technical Committee to recommend any of the following case  
249 example(s) as the recommended agile software methodology to be used for Web services  
250 implementation. The Web service implementation methodology is intended to be generic and  
251 should be able to incorporate to any agile software methodology.  
252

## 253 1.1 Rational Unified Process (Example)

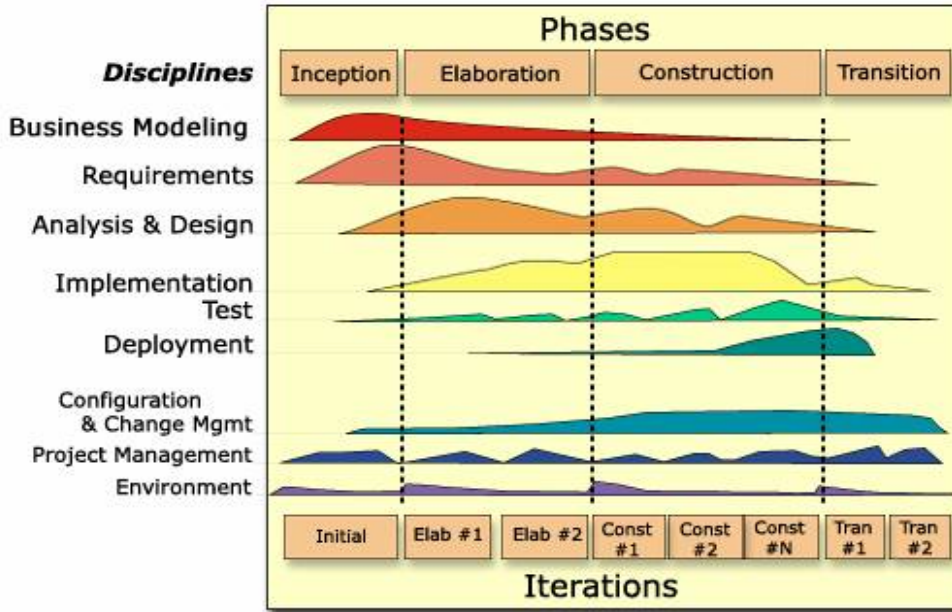
254 In this section, the Rational Unified Process (RUP) is illustrated as an example of how the  
255 Web service implementation methodology can be incorporated into RUP. Although the  
256 example tries to be as complete as possible, it is foreseeable that different projects set-up  
257 may be different in nature and would need to further customise this case example according  
258 to their needs.  
259

### 260 1.1.1 Overview

261  
262 The Rational Unified Process® is a software engineering process. It provides a disciplined  
263 approach to assigning tasks and responsibilities within a development organization. Its goal  
264 is to ensure the production of high quality software that meets the needs of its end users  
265 within a predictable schedule and budget.  
266

267 As illustrated in Source: IBM-Rational Software  
268 Figure 1, the overall architecture of RUP has two dimensions: the horizontal axis represents  
269 time and shows the lifecycle aspects of the process as it unfolds. This dimension is  
270 expressed in terms of phases, iterations and milestones. The vertical axis represents  
271 disciplines that logically group activities by nature. This second dimension portrays the static  
272 aspect of the process and is described in terms of process components, disciplines, activities,  
273 workflows, artifacts, and roles.  
274

275 The “humps” in the graph illustrate the relative emphases of the disciplines change over the  
276 life of the project. For example, in early iterations more time is spent on Requirements, and in  
277 later iterations more time is spent on implementation.  
278  
279



Source: IBM-Rational Software

Figure 1: The Rational Unified Process

280  
281  
282  
283  
284  
285  
286  
287  
288

As the terms used in RUP are different from the terms used in Web Service Implementation Methodology, Table 1 maps the terms used between the two software methodologies.

RUP		Web Service Implementation Methodology	
Phases		Lifecycle	
Disciplines		Phases	
Roles		Roles	
Analyst	System Analysts Requirements Specifier	Analyst	Architect Requirements Analyst
Developer	Software Architect Designer Implementer Integrator	Developer	Designer Developer Deployer
Tester	Test Manager Test Analyst Test Designer Tester Test System Administrator	Tester	Test Manager Test Designer Tester Test System Administrator
Production & Support	Deployment Manager DBA Process Engineer	Others	User System Engineer Project Manager Stakeholder
Artifacts		Artifacts	
Glossary Vision		Business Requirement Specifications	

Stakeholder Requests Requirements Attributes Requirements Management Plan Software Requirement Software Requirements Specifications Use Case Model Supplementary Specifications	Requirement Specifications
Software Architecture Document	Software Architecture Specifications
Design Model Analysis Model Use Case Realization	Web Service Signature Specifications XML Schema Design Specifications
Test Plan Test Environment Configuration Test Strategy	Test Plan – UAT and System Test Test Plan – Performance Test Test Plan – Integration / Interoperability Test Test Plan – Functional Test
Test-Ideas List Test Case Test Data Test Suite	Test Plan – Testbed
Test Script	Test Scripts Unit Test Scripts Client Test Code
Test Log Test Results	Test Results
Implementation Model Implementation Element Build	Implementation Codes Web Service Client Codes
Deployment Unit	Release Notes Deployment Scripts WSDL File
End-User Support Material	Interoperability Guide User Guide On-line Help Tutorials
Training Materials	Training Materials
Change Request	Not Applicable
Deployment Plan	Not Applicable
Installation Artifacts	Not Applicable
Project Specific Guidelines	Not Applicable
Release Notes	Not Applicable
Test Environment Summary	Not Applicable

289 *Table 1: Mapping of the terms used by both the RUP and the Web Service Implementation Methodology*

290

## 291 **1.1.2 Approach**

292

293 In RUP, there are 9 disciplines altogether namely, Business Modeling, Requirements,  
 294 Analysis & Design, Implementation, Test, Deployment, Configuration & Change Management,  
 295 Project Management and Environment. However, as the focus of this Web service  
 296 implementation methodology is on the implementation aspects only, the disciplines Business  
 297 Modeling, Configuration & Change Management, Project Management and Environment are  
 298 beyond the scope of this document and will not be discussed here. The following disciplines  
 299 as shown in Figure 2, are of interest and will be illustrated in detail in the subsequent  
 300 sections.

301

302



303

304

Figure 2: The RUP development disciplines

305

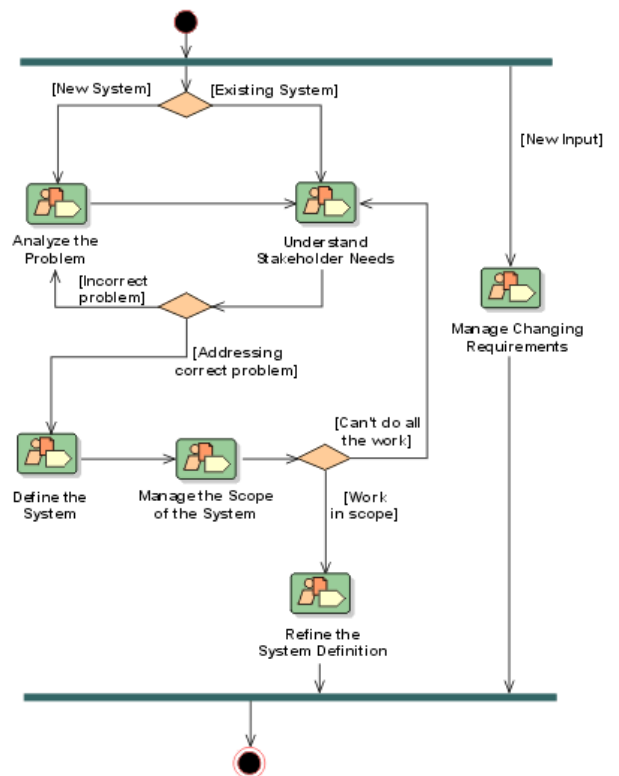
306 For each of these disciplines, the relevant workflow details will be described and are extracted  
307 directly from RUP. Most of the activities in these workflow do not need special customisation.  
308 However, in areas where the activities are specific to Web services, these will be highlighted  
309 in bold<sup>1</sup>.

310

### 311 1.1.2.1 Discipline: Requirements



- **Analyze Problem**
  - **Capture a Common Vocabulary**
  - **Develop Vision (across all WS)**
- **Understand Stakeholder Needs**
  - **Elicit Needs**
    - **Categorise needs into respective WSs**
    - **Identify potential WS**
  - **Develop Vision**
    - **Refine the Categorisation Based on Features**
- **Manage Dependencies**
  - **Prioritise the WS**
- **Find Actors & UCs (per WS)**
- **Define System**
  - **Find Actors & UCs (per WS)**
- **Manage Scope Of System**
  - **Prioritise the UCs (per WS)**
- **Refine System Definition**
  - **Detail a UC**
  - **Detail the Software Requirements**



Source: IBM-Rational Software

312

313

314

315

316

Figure 3: Requirements Workflow

#### 317 1.1.2.1.1 Workflow Detail: Analyze the Problem

##### 318 1.1.2.1.1.1 Activity: Capture a Common Vocabulary

- Find Common Terms

##### 320 1.1.2.1.1.1.1 Role

321 System Analyst

<sup>1</sup> The activities in italics represent refinement of an activity previously defined.

322 **1.1.2.1.1.1.2 Artifact**

323 The results should be recorded in Glossary.  
324

325 **1.1.2.1.1.2 Activity: Develop Vision (across all Web services)**

- 326 - Gain agreement on the some problems faced
- 327 - Identify stakeholders of the Web services
- 328 - Define boundaries of Web services
- 329 - Identify (initial) constraints to be imposed on the Web services
- 330 - Formulate Problem Statement (positioning why the need to develop Web services)

331 **1.1.2.1.1.2.1 Role**

332 System Analyst

333 **1.1.2.1.1.2.2 Artifact**

334 The results should be recorded in Requirements Attributes and Vision.  
335

336 **1.1.2.1.2 Workflow Detail: Understand Stakeholder Needs**

337 **1.1.2.1.2.1 Activity: Elicit Needs**

- 338 - Determine sources for requirements (who and where can you gather the
- 339 requirements of Web services)
- 340 - Gather information (based on stakeholders identified)
- 341 - Conduct brainstorming session
- 342 - **Categorise the needs into respective Web services**
- 343 - **Identify the Web services**

344 **1.1.2.1.2.1.1 Role**

345 System Analyst

346 **1.1.2.1.2.1.2 Artifact**

347 The results should be recorded in Stakeholder Requests.  
348

349 **1.1.2.1.2.2 Activity: Develop Vision**

- 350 - *Gain agreement on the some problems faced*
- 351 - *Identify stakeholders of the Web services*
- 352 - Refine boundaries of Web services
- 353 - Identify the new constraints (or refine on existing constraints)
- 354 - *Formulate Problem Statement (positioning why the need to develop Web services)*
- 355 - Define features of the Web services based on the needs list
- 356 - **Refine the categorisation of the Web services based on the features**

357 **1.1.2.1.2.2.1 Role**

358 System Analyst

359 **1.1.2.1.2.2.2 Artifact**

360 The results should be recorded in Requirements Attributes and Vision.  
361

362 **1.1.2.1.2.3 Activity: Manage Dependencies**

- 363 - Assign attributes to the features of the Web services

- 364 - **Prioritise the Web services**
- 365 - Establish and verify traceability (what requirement types to be traced)
- 366 - Manage changing requirements

#### 367 **1.1.2.1.2.3.1 Role**

368 System Analyst

#### 369 **1.1.2.1.2.3.2 Artifact**

370 The results should be recorded in Requirements Attributes, Requirements Management Plan  
371 and Vision.  
372

#### 373 **1.1.2.1.2.4 Activity: Find Actors and Use Cases (per Web service)**

- 374 - Find Actors
- 375 - Find Use Cases
- 376 - Create use case model

#### 377 **1.1.2.1.2.4.1 Role**

378 System Analyst

#### 379 **1.1.2.1.2.4.2 Artifact**

380 The results should be recorded in Use Case Model and Supplementary Specifications.  
381

### 382 **1.1.2.1.3 Workflow Detail: Define the System**

#### 383 **1.1.2.1.3.1 Activity: Find Actors and Use Cases (per Web service)**

- 384 - *Find Actors*
- 385 - *Find Use Cases*
- 386 - Refine use case model to include outlined use cases

#### 387 **1.1.2.1.3.1.1 Role**

388 System Analyst

#### 389 **1.1.2.1.3.1.2 Artifact**

390 The results should be recorded in Use Case Model and Supplementary Specifications.  
391

### 392 **1.1.2.1.4 Workflow Detail: Manage the Scope of the System**

#### 393 **1.1.2.1.4.1 Activity: Prioritise the Use Cases (per Web service)**

- 394 - Prioritise Use Cases and Scenarios

#### 395 **1.1.2.1.4.1.1 Role**

396 Software Architect

#### 397 **1.1.2.1.4.1.2 Artifact**

398 The results should be recorded in Software Architecture Document and Software  
399 Requirement.  
400

401 **1.1.2.1.5 Workflow Detail: Refine the System Definition**

402 **1.1.2.1.5.1 Activity: Detail a Use Case**

- 403 - Review and Refine the Scenarios
- 404 - Detail the Flow of Events
- 405 - Structure the Flow of Events
- 406 - Describe any Special Requirements

407 **1.1.2.1.5.1.1 Role**

408 Requirements Specifier

409 **1.1.2.1.5.1.2 Artifact**

410 The results should be recorded in Use Case Model, Supplementary Specifications.

411

412 **1.1.2.1.5.2 Activity: Detail the Software Requirements**

- 413 - Detail the Software Requirements
- 414 - Generate Supporting Reports (Use Case Model and Supplementary Specs)

415 **1.1.2.1.5.2.1 Role**

416 Requirements Specifier

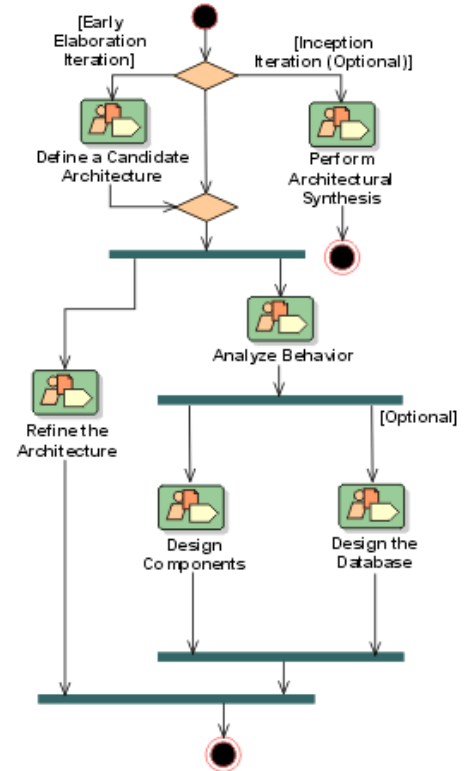
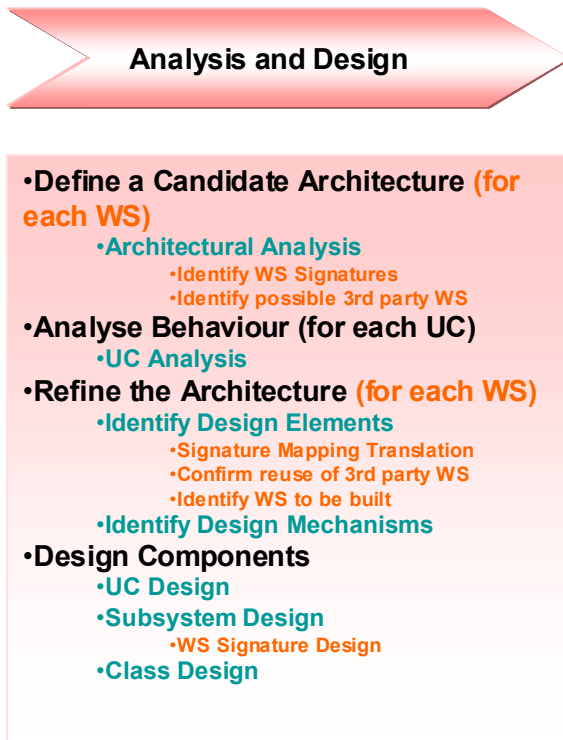
417 **1.1.2.1.5.2.2 Artifact**

418 The results should be recorded in Software Requirement and Software Requirements  
419 Specifications.

420

421 1.1.2.2 Discipline: Analysis and Design

422



Source: IBM-Rational Software

423  
424

Figure 4: Analysis and Design Workflow

425

426

427 1.1.2.2.1 Workflow Detail: Define a Candidate Architecture (for each Web service)

428

429 1.1.2.2.1.1 Activity: Architectural Analysis

- 430 - Define the high-level organization
- 431 - Identify key abstractions
- 432 - Identify analysis mechanisms
- 433 - Identify the use case realization for the current iteration
- 434 - **Identify the Web Service signatures**
- 435 - **Identify the possible 3<sup>rd</sup> party Web Service reuse**

436 1.1.2.2.1.1.1 Role

437 Software Architect

438 1.1.2.2.1.1.2 Artifact

439 The results should be recorded in Software Architecture Document and Design Model.

440

441 1.1.2.2.2 Workflow Detail: Analyse Behaviour (for each use case)

442 1.1.2.2.2.1 Activity: Use Case Analysis

- 443 - Find analysis classes from Use Case Behaviour

- 444 - Distribute behaviour to analysis classes
- 445 - Describe responsibilities
- 446 - Reconcile the use case realization
- 447 - Qualify analysis mechanisms

#### 448 **1.1.2.2.1.1 Role**

449 Designer

#### 450 **1.1.2.2.1.2 Artifact**

451 The results should be recorded in Analysis Model and Use Case Realization.  
452

### 453 **1.1.2.2.3 Workflow Detail: Refine the Architecture (for each Web service)**

#### 454 **1.1.2.2.3.1 Activity: Identify Design Elements**

- 455 - **Signature mapping translation**
- 456 - **Confirm reuse of 3<sup>rd</sup> party Web Services**
- 457 - **Identify Web Services to be built**
- 458 - Identify classes and subsystems based on the activity Use Case Analysis
- 459 - Identify subsystem interfaces
  - 460 o Identify candidate interfaces (from internal or external source)
  - 461 o Look for similarities between interfaces
- 462 - Identify reuse opportunities
  - 463 o Look for existing subsystems with similar interfaces
  - 464 o Modify the newly identified interfaces to improve the fit
- 465 - Update the organization of the design model

#### 466 **1.1.2.2.3.1.1 Role**

467 Software Architect

#### 468 **1.1.2.2.3.1.2 Artifact**

469 The results should be recorded in Design Model.  
470

#### 471 **1.1.2.2.3.2 Activity: Identify Design Mechanisms**

- 472 - Inventory the implementation mechanisms
- 473 - Map design mechanisms to implementation mechanisms
- 474 - Document architectural mechanisms (in terms of patterns)

#### 475 **1.1.2.2.3.2.1 Role**

476 Software Architect

#### 477 **1.1.2.2.3.2.2 Artifact**

478 The results should be recorded in Software Architecture Document and Design Model.  
479

### 480 **1.1.2.2.4 Workflow Detail: Design Components**

#### 481 **1.1.2.2.4.1 Activity: Use Case Design**

- 482 - Describe Interactions between design objects (refine interaction diagrams)
- 483 - Simplify sequence diagrams with interfaces of subsystems (if any subsystems found)
- 484 - Unify design classes and subsystems

485 **1.1.2.2.4.1.1 Role**

486 Designer

487 **1.1.2.2.4.1.2 Artifact**

488 The results should be recorded in Design Model and Use Case Realization.  
489

490 **1.1.2.2.4.2 Activity: Subsystem Design**

- 491 - Interface realization (**Web service signature design**)
- 492 - Distribute subsystem behaviour to subsystem elements (design the Web service)
- 493 - Document subsystem elements (internal structure of the Web service)
- 494 - Describe subsystem dependencies

495 **1.1.2.2.4.2.1 Role**

496 Designer

497 **1.1.2.2.4.2.2 Artifact**

498 The results should be recorded in Design Model.  
499

500 **1.1.2.2.4.3 Activity: Class Design**

- 501 - Create initial design classes
- 502 - Define class visibility
- 503 - Define operations
- 504 - Define attributes
- 505 - Define dependencies
- 506 - Define associations
- 507 - Define generalizations
- 508 - Handle non-function requirements

509 **1.1.2.2.4.3.1 Role**

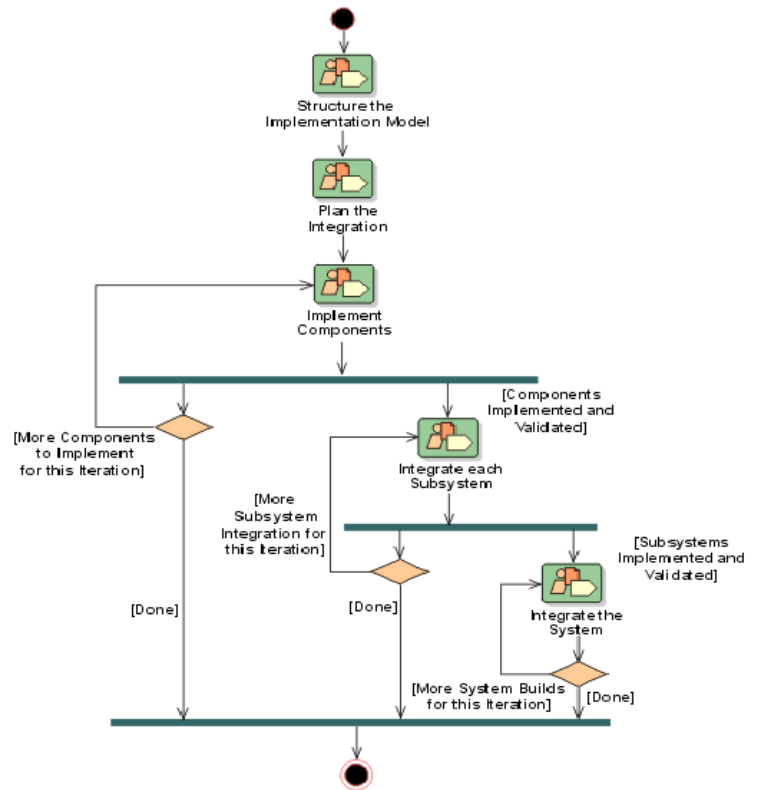
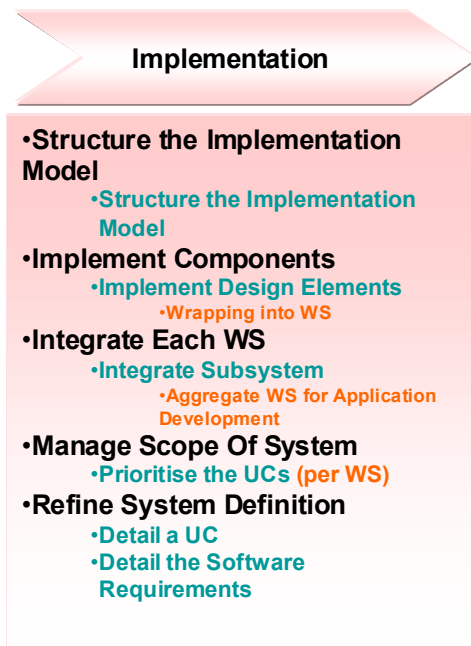
510 Designer

511 **1.1.2.2.4.3.2 Artifact**

512 The results should be recorded in Design Model.  
513

514 **1.1.2.3 Discipline: Implementation**

515



Source: IBM-Rational Software

516

517

518

Figure 5: Implementation Workflow

519

520 **1.1.2.3.1 Workflow Detail: Structure the Implementation Model**

521 **1.1.2.3.1.1 Activity: Structure the Implementation Model**

- 522 - Establish the implementation model structure
- 523 - Define imports for each implementation components
- 524 - Decide how to treat executables (and other derived objects)

525 **1.1.2.3.1.1.1 Role**

526 Software Architect

527 **1.1.2.3.1.1.2 Artifact**

528 The results should be recorded in Software Architecture Document and Implementation Model.

529

531 **1.1.2.3.2 Workflow Detail: Implement Components**

532 **1.1.2.3.2.1 Activity: Implement Design Elements**

- 533 - Implement operations
- 534 - Implement associations
- 535 - Implement attributes
- 536 - Provide feedback to design

537 - **Wrapping into Web Service**

538 **1.1.2.3.2.1.1 Role**

539 Implementer

540 **1.1.2.3.2.1.2 Artifact**

541 The results should be recorded in Implementation Element.  
542

543 **1.1.2.3.3 Workflow Detail: Integrate Each Web Service**

544 **1.1.2.3.3.1 Activity: Integrate Subsystem**

- 545 - Integrate implementation elements
- 546 - Aggregate Web Service for application development

547 **1.1.2.3.3.1.1 Role**

548 Integrator

549 **1.1.2.3.3.1.2 Artifact**

550 The results should be recorded in Build and Implementation Model.  
551

552 **1.1.2.3.4 Workflow Detail: Manage the Scope of the System**

553 **1.1.2.3.4.1 Activity: Prioritise the Use Cases (per Web service)**

- 554 - *Prioritise Use Cases and Scenarios*

555 **1.1.2.3.4.1.1 Role**

556 Software Architect

557 **1.1.2.3.4.1.2 Artifact**

558 The results should be recorded in Software Architecture Document and Software  
559 Requirement.  
560

561 **1.1.2.3.5 Workflow Detail: Refine the System Definition**

562 **1.1.2.3.5.1 Activity: Detail a Use Case**

- 563 - *Review and Refine the Scenarios*
- 564 - *Detail the Flow of Events*
- 565 - *Structure the Flow of Events*
- 566 - *Describe any Special Requirements*

567 **1.1.2.3.5.1.1 Role**

568 Requirement Specifier

569 **1.1.2.3.5.1.2 Artifact**

570 The results should be recorded in Use Case Model and Supplementary Specifications.  
571

572 **1.1.2.3.5.2 Activity: Detail the Software Requirements**

573 - *Detail the Software Requirements*

574 - *Generate Supporting Reports (Use Case Model and Supplementary Specs)*

575 **1.1.2.3.5.2.1 Role**

576 Requirement Specifier

577 **1.1.2.3.5.2.2 Artifact**

578 The results should be recorded in Software Requirement and Software Requirements  
579 Specifications.

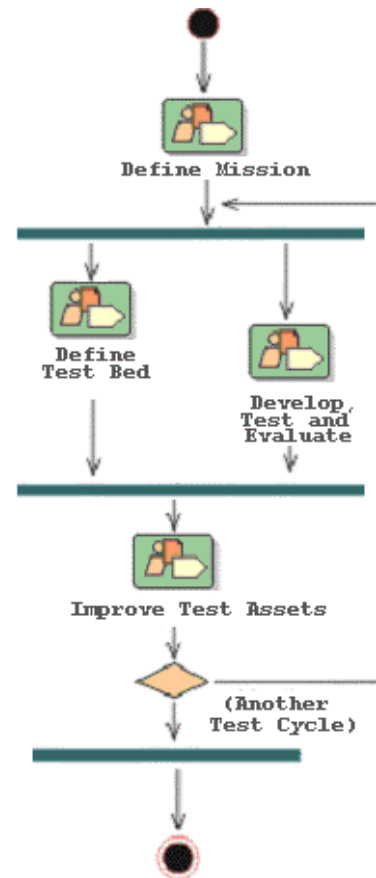
580

581 **1.1.2.4 Discipline: Testing**

582



- **Define Mission**
  - **Identify Test Motivators**
    - **Functionality**
    - **Reliability**
    - **Performance**
    - **interoperability**
  - **Agree on Mission**
  - **Define Assessment & Traceability Needs**
  - **Define Test Approach**
    - **API Level**
    - **SOAP Level**
  - **Identify Test ideas**
- **Define Test Bed**
  - **Identify Test Environment**
  - **Prepare H/W & S/W Infrastructure**
  - **Prepare Test Data Sets**
- **Develop, Test & Evaluate**
  - **Define Test Details**
  - **Implement Test**
    - **Generate WS Client**
  - **Implement Test Suite**
  - **Execute Test Suite**
  - **Analyse Test Failures**
  - **Determine Test Results**
- **Improve Test Assets**
  - **Define Test Approach**
  - **Identify Test Ideas**
  - **Prepare Guidelines for Project**



583

584

Source: IBM-Rational Software

585

Figure 6: Testing Workflow

586

587 **1.1.2.4.1 Workflow Detail: Define Mission**

588 **1.1.2.4.1.1 Activity: Identify Test Motivators**

- 589 - Identify iteration target items
  - 590 ○ Web services
  - 591 ○ configuration (deployment platform)
- 592 - Gather and examine related information
  - 593 ○ dependencies of services to be tested
    - 594 ▪ other services
    - 595 ▪ configuration (deployment platform)
- 596 - Identify candidate motivators
  - 597 ○ Web services
    - 598 ▪ **functionality**
    - 599 ▪ usability

- 600                           ▪ **reliability**
- 601                           ▪ **performance**
- 602                           ▪ supportability
- 603 - Determine quality risks
  - 604           ○ prioritise the tests requirements
    - 605                   ▪ architecture significant
    - 606                   ▪ value of service
    - 607                   ▪ stability
  - 608           ○ define the acceptable quality level
- 609 - Define motivator list
  - 610           ○ define the specific Web services ready to be tested
- 611 - Maintain traceability relationships
  - 612           ○ manage the test requirements to the other requirements
    - 613                   ▪ impact analysis
- 614 - Evaluate and verify your results
  - 615           ○ verify with team members on the objectives of this activity

#### 616 **1.1.2.4.1.1.1 Role**

617 Test Manager

#### 618 **1.1.2.4.1.1.2 Artifact**

619 The results should be recorded in Test Plan.

620

#### 621 **1.1.2.4.1.2 Activity: Agree on the Mission**

- 622 - Investigate options for the scope of the assessment effort
  - 623           ○ determine the resources needed to achieve the testing
  - 624           ○ scope the test based on existing resources
- 625 - Formulate mission statement
  - 626           ○ determine a balance between the need for:
    - 627                   ▪ Test for correctness
    - 628                   ▪ Test for completeness
  - 629           ○ determine Test Coverage
    - 630                   ▪ Code
    - 631                   ▪ Test requirements
    - 632                   ▪ Defect
    - 633                   ▪ Test Result
  - 634           ○ determine the necessary stages of testing
    - 635                   ▪ Unit                (Formal/Informal)
    - 636                   ▪ Integration       (Formal/Informal)
    - 637                   ▪ System            (Formal)
    - 638                   ▪ UAT                (Formal)
- 639 - Identify test deliverables
  - 640           ○ **Test Plan (based on type of tests)**
    - 641                   ▪ **Test Cases (high-level)**
      - 642                           • **Test Scenarios (test case)**
        - 643                                   ○ Explanation
        - 644                                   ○ Deriving Test Cases from Use Cases
        - 645                                   ○ Deriving Test Cases from Supplementary Specifications
          - 646   ▪ Deriving Test Cases for Performance Tests
          - 647   ▪ Deriving Test Cases for Security / Access Tests
          - 648   ▪ Deriving Test Cases for Configuration Tests
          - 649   ▪ Deriving Test Cases for Installation Tests
          - 650   ▪ Deriving Test Cases for other Non Functional Tests

- 654 ○ Deriving Test Cases for Product Acceptance Tests
- 655 ○ Build Verification Test Cases for Regression Tests
- 656 ○ Defining Test Data for Test Cases
- 657 ○ use the existing test ideas as a mean to identify
- 658 possible scenarios
- 659 ○ verify all extension points where not explicitly defined
- 660     ▪ **Test Procedures/Steps**
- 661         • **Test Scripts**
- 662     ▪ **Test Result**
- 663 - Evaluate and verify your results
- 664     ○ verify with team members and stakeholder on the objectives of this activity

#### 665 **1.1.2.4.1.2.1 Role**

666 Test Manager

#### 667 **1.1.2.4.1.2.2 Artifact**

668 The results should be recorded in Test Plan.

669

#### 670 **1.1.2.4.1.3 Activity: Define Assessment and Traceability Needs**

- 671 - Identify assessment and traceability requirements
- 672     ○ how to assess formal and informal testing
- 673 - Consider constraints
- 674     ○ limitation that prohibits the testing effort
- 675         ▪ existing skills set
- 676         ▪ availability of resources
- 677             • tools
- 678             • information
- 679             • process
- 680             • skills set
- 681 - Consider possible strategies
- 682     ○ acquire required skills
- 683         ▪ vendor
- 684         ▪ training
- 685     ○ outsource
- 686 - Define and agree on the assessment strategy
- 687     ○ define checkpoint to assess the strategy (verify test approach)
- 688 - Define tool requirements
- 689     ○ Use existing tool
- 690         ▪ Good match
- 691         ▪ Workaround solution
- 692         ▪ Acquire new technology
- 693     ○ Increase productivity
- 694         ▪ Regression testing
- 695 - Evaluate and verify your results
- 696     ○ verify with team members and stakeholder on the objectives of this activity

#### 697 **1.1.2.4.1.3.1 Role**

698 Test Analyst

#### 699 **1.1.2.4.1.3.2 Artifact**

700 The results should be recorded in Test Plan.

701

#### 702 1.1.2.4.1.4 Activity: Define Test Approach

- 703 - Examine test motivators and target test items
- 704     o test requirements - FURPS
- 705 - Examine the software architecture
- 706     o understand how the target is deployed
- 707         ▪ select the appropriate test point
- 708 - Consider the appropriate breadth and depth of the test approach
- 709     o determine strategy for the test points
- 710         ▪ **API level**
- 711         ▪ **SOAP Client level**
- 712             • Atomic / Collaboration
- 713 - Identify existing test techniques for reuse
- 714     o gather experience from team members
- 715         ▪ identify the possible technique for use
- 716             • select appropriate technique
- 717             • the rest as contingency
- 718 - Define techniques
- 719     o Outline each technique for each type of test
- 720         ▪ Automated/Manual/semi-automated
- 721             • Require tool
- 722         ▪ Framework
- 723             • Test Design
- 724                 o maintenance
- 725                 o effectiveness
- 726                 o reusable
- 727                 o longevity
- 728             • Implementation tips and tricks
- 729         ▪ Test bed pre-condition/post-condition
- 730             • Test data management
- 731         ▪ Test Result logging/reporting
- 732 - Define the test asset configuration management strategy
- 733     o Identify possible test assets management
- 734         ▪ version control
- 735         ▪ backup
- 736 - Survey availability of reusable assets
- 737     o Identify existing test asset which could be reused
- 738         ▪ Based on test design
- 739 - Evaluate and verify your results
- 740     o verify with team members and stakeholder on the objectives of this activity

#### 741 1.1.2.4.1.4.1 Role

742 Test Designer

#### 743 1.1.2.4.1.4.2 Artifact

744 The results should be recorded in Test Environment Configuration, Test Plan and Test  
745 Strategy.

746

#### 747 1.1.2.4.1.5 Activity: Identify Test Ideas

- 748 - Identify relevant Test Motivators and Target Test Items
- 749     o Determine level of testing required for specific/partial targets
- 750         ▪ Correctness (must)
- 751         ▪ Completeness (vary)
- 752             • The test ideas will help you to identify the possible test
- 753             scenarios you would need to address based on the signature
- 754             of each method call found in the specific Web service

- 755 - Examine relevant available Test-Idea Catalogs
- 756     o Check existing best practices to test specific
- 757 - Brainstorm additional Test Ideas
- 758     o Enhance Test Idea catalog
- 759     o Examine Web service and analyse every method signature
- 760         ▪ Valid value
- 761             • Boundary value
- 762                 o Identify possible values for each parameter. Valid
- 763                     and invalid
- 764             • Special value
- 765         ▪ Invalid value
- 766             • Must conform to the syntax of the parameter data structure
- 767         ▪ Guidelines: Test Ideas for Booleans and Boundaries
- 768         ▪ Guidelines: Test Ideas for Method Calls
- 769 - Refine the Test-Ideas List
- 770     o update existing list
- 771 - Evaluate and verify your results
- 772     o verify with team members and stakeholder on the objectives of this activity

#### 773 **1.1.2.4.1.5.1 Role**

774 Test Analyst

#### 775 **1.1.2.4.1.5.2 Artifact**

776 The results should be recorded in Test-Ideas List.

777

### 778 **1.1.2.4.2 Workflow Detail: Define Test Bed**

#### 779 **1.1.2.4.2.1 Activity: Identify Test Environment**

- 780 - Identify the application architecture and deployment
- 781     o Hardware and Software (20% effort)
- 782         ▪ Configurations
- 783     o Test Data (80% effort)
- 784         ▪ Explanation
- 785         ▪ Depth
- 786         ▪ Breadth
- 787         ▪ Scope
- 788         ▪ Architecture
- 789 - Identify the test facilities
- 790     o Tools
- 791         ▪ Hardware and Software requirements
- 792     o Roles
- 793         ▪ Sys Admin, DBA, etc
- 794     o Security

#### 795 **1.1.2.4.2.1.1 Role**

796 Test Designer

#### 797 **1.1.2.4.2.1.2 Artifact**

798 The results should be recorded in Test Environment Configuration.

799

#### 800 **1.1.2.4.2.2 Activity: Prepare H/W & S/W Infrastructure**

- 801 - Select the appropriate technique to control environment

- 802 - Set up test environment
- 803     o settings and configuration
- 804 - Restore test environment
- 805     o Logs
- 806     o Temp files
- 807     o Software and Hardware settings

#### 808 **1.1.2.4.2.2.1 Role**

809 Test System Administrator, DBA

#### 810 **1.1.2.4.2.2.2 Artifact**

811 The results should be recorded in Test Environment Configuration.  
812

#### 813 **1.1.2.4.2.3 Activity: Prepare Test Data Sets**

- 814 - Identify the test data to be used
- 815     o Depth
- 816     o Breadth
- 817     o Scope
- 818 - Identify the strategy to restore test data
- 819     o data refresh
- 820     o data re-initialize
- 821     o data reset
- 822     o data roll forward

#### 823 **1.1.2.4.2.3.1 Role**

824 Test Analyst

#### 825 **1.1.2.4.2.3.2 Artifact**

826 The results should be recorded in Test Data.  
827

### 828 **1.1.2.4.3 Workflow Detail: Develop, Test and Evaluate**

#### 829 **1.1.2.4.3.1 Activity: Define Test Details**

- 830 - Examine the Target Test Item and related Test-Ideas List
- 831 - Select a subset of the test ideas to detail
- 832 - For each test idea, design the Test;
- 833     o identify input, output and execution conditions
- 834     o identify candidate points of observation
- 835     o identify candidate points of control
- 836     o Identify appropriate oracles
- 837 - Define required data sources, values and ranges
- 838 - Source sufficient consumable Test Data
- 839 - Maintain traceability relationships
- 840 - Evaluate and verify your results

#### 841 **1.1.2.4.3.1.1 Role**

842 Test Analyst

#### 843 **1.1.2.4.3.1.2 Artifact**

844 The results should be recorded in Test Case, Test Data and Test Script.  
845

846 **1.1.2.4.3.2 Activity: Implement Test**

- 847 - Test Design
- 848     o Macro-Level
- 849         ▪ Collaboration of high-level test case
- 850     o Micro-Level
- 851         ▪ Test script implementation level
- 852             • Affected by the test data management
- 853                 o Data partitioning strategy
- 854                 o Data “life-cycle” strategy
- 855 - Select appropriate implementation technique
- 856 - Set up test environment preconditions
- 857 - **Generate Web service client**
- 858 - Implement the test
- 859 - Establish external data sets
- 860 - Verify the test implementation
- 861 - Restore test environment to known state
- 862 - Maintain traceability relationships
- 863 - Evaluate and verify your results

864 **1.1.2.4.3.2.1 Role**

865 Tester

866 **1.1.2.4.3.2.2 Artifact**

867 The results should be recorded in Test Script.  
868

869 **1.1.2.4.3.3 Activity: Implement Test Suite**

- 870 - Examine candidate Test Suites
- 871 - Examine related Tests and Target Test Items
- 872 - Identify Test dependencies
- 873 - Identify opportunities for reuse
- 874 - Apply necessary infrastructure utilities
- 875 - Determine recovery requirements
- 876 - Implement recovery requirements
- 877 - Stabilize the Test Suite
- 878 - Maintain traceability relationships
- 879 - Evaluate and verify your results

880 **1.1.2.4.3.3.1 Role**

881 Tester

882 **1.1.2.4.3.3.2 Artifact**

883 The results should be recorded in Test Suite.  
884

885 **1.1.2.4.3.4 Activity: Execute Test Suite**

- 886 - Setup test environment to known state
- 887 - Set execution tool options
- 888 - Schedule Test Suite execution
- 889 - Execute Test Suite
- 890 - Evaluate execution of Test Suite
- 891 - Recover from halted tests
- 892 - Inspect the Test Logs for completeness and accuracy

- 893 - Restore test environment to known state
- 894 - Maintain traceability relationships
- 895 - Evaluate and verify your results

#### 896 **1.1.2.4.3.4.1 Role**

897 Tester

#### 898 **1.1.2.4.3.4.2 Artifact**

899 The results should be recorded in Test Log.  
900

#### 901 **1.1.2.4.3.5 Activity: Analyze Test Failures**

- 902 - Examine the Test Logs
- 903 - Capture nontrivial incident data
- 904 - Identify procedural errors in the test
- 905 - Locate and isolate failures
- 906 - Diagnose failure symptoms and characteristics
- 907 - Identify candidate solutions
- 908 - Document your findings appropriately
- 909 - Evaluate and verify your results

#### 910 **1.1.2.4.3.5.1 Role**

911 Tester

#### 912 **1.1.2.4.3.5.2 Artifact**

913 The results should be recorded in Change Request.  
914

#### 915 **1.1.2.4.3.6 Activity: Determine Test Results**

- 916 - Examine all test incidents and failures
- 917 - Create and maintain Change Requests
- 918 - Analyze and evaluate status
- 919 - Make an assessment of the current quality experience
- 920 - Make an assessment of outstanding quality risks
- 921 - Make an assessment of test coverage
- 922 - Draft the Test Evaluation Summary
- 923 - Advise stakeholders of key findings
- 924 - Evaluate and verify your results

#### 925 **1.1.2.4.3.6.1 Role**

926 Test Analyst

#### 927 **1.1.2.4.3.6.2 Artifact**

928 The results should be recorded in Test Evaluation Summary and Test Results.  
929

### 930 **1.1.2.4.4 Workflow Detail: Improve Test Assets**

#### 931 **1.1.2.4.4.1 Activity: Define Test Approach (Refinement)**

- 932 - As in "Define Test Approach" activity found in "Define Mission" Workflow

933 **1.1.2.4.4.1.1 Role**

934 Test Designer

935 **1.1.2.4.4.1.2 Artifact**

936 The results should be recorded in Test Environment Configuration, Test Plan and Test  
937 Strategy.  
938

939 **1.1.2.4.4.2 Activity: Identify Test Ideas (Refinement)**

940 - As in “Identify Test Ideas” activity found in “Define Mission” Workflow

941 **1.1.2.4.4.2.1 Role**

942 Test Analyst

943 **1.1.2.4.4.2.2 Artifact**

944 The results should be recorded in Test-Ideas List.  
945

946 **1.1.2.4.4.3 Activity: Prepare Guidelines for the Project**

- 947 - Identify the Project's Needs for Guidelines  
948 - Prepare Guidelines for Project Use  
949 - Maintain Guidelines

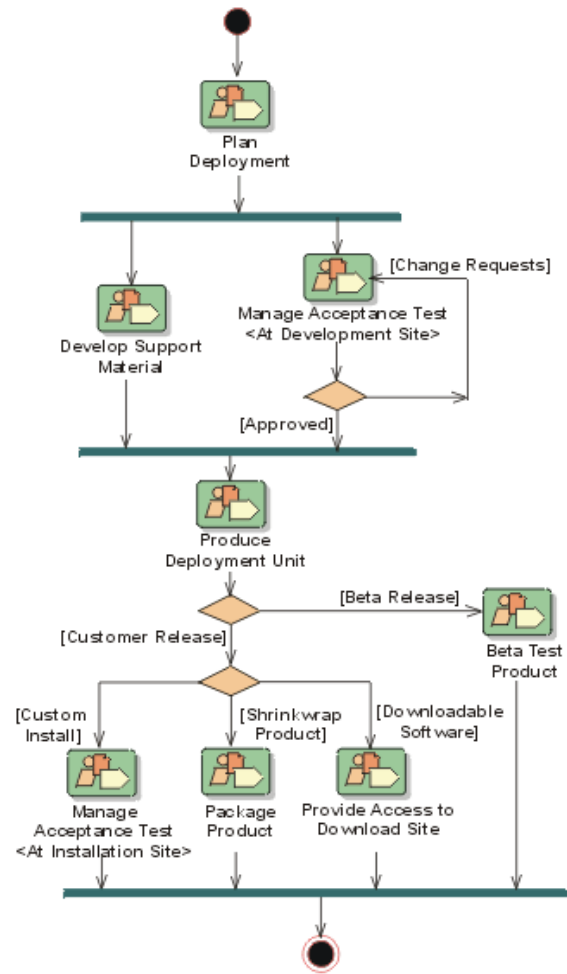
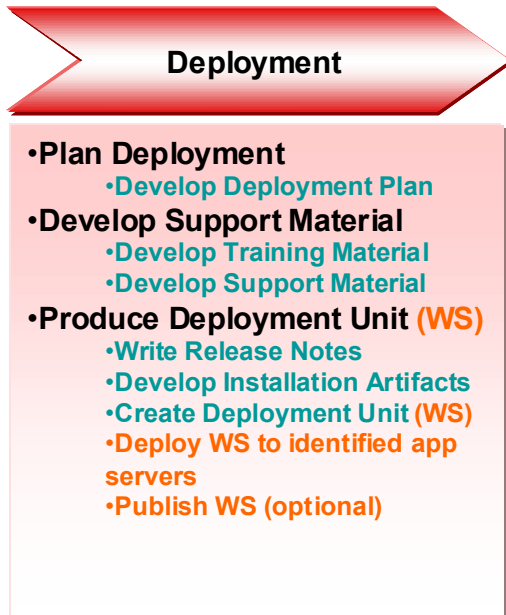
950 **1.1.2.4.4.3.1 Role**

951 Process Engineer

952 **1.1.2.4.4.3.2 Artifact**

953 The results should be recorded in Project Specific Guidelines.  
954

955 **1.1.2.5 Discipline: Deployment**



Source: IBM-Rational Software

956  
957

958 *Figure 7: Deployment Workflow*

959 **1.1.2.5.1 Workflow Detail: Plan Deployment**

960 **1.1.2.5.1.1 Activity: Develop Deployment Plan**

- 961 - Plan how to produce the software
- 962 - Plan how to package the software
- 963 - Plan how to distribute the software
- 964 - Plan how to install the software
- 965 - Migration
- 966 - Providing help and assistance to the users

967 **1.1.2.5.1.1.1 Role**

968 Deployment Manager

969 **1.1.2.5.1.1.2 Artifact**

970 The results should be recorded in Deployment Plan.  
971

- 972 **1.1.2.5.2 Workflow Detail: Develop Support Material**
- 973 **1.1.2.5.2.1 Activity: Develop Training Material**
- 974 - Develop an outline for the training materials  
975 - Write the training materials
- 976 **1.1.2.5.2.1.1 Role**
- 977 Course Developer
- 978 **1.1.2.5.2.1.2 Artifact**
- 979 The results should be recorded in Training Materials.  
980
- 981 **1.1.2.5.2.2 Activity: Develop Support Material**
- 982 - Develop end-user support material
- 983 **1.1.2.5.2.2.1 Role**
- 984 Technical Writer
- 985 **1.1.2.5.2.2.2 Artifact**
- 986 The results should be recorded in End-User Support Material.  
987
- 988 **1.1.2.5.3 Workflow Detail: Produce Deployment Unit (Web service)**
- 989 **1.1.2.5.3.1 Activity: Write Release Notes**
- 990 - Describe the major new features and changes in the release  
991 - Describe any known bugs and limitations or workarounds to using the product
- 992 **1.1.2.5.3.1.1 Role**
- 993 Deployment Manager
- 994 **1.1.2.5.3.1.2 Artifact**
- 995 The results should be recorded in Release Notes.  
996
- 997 **1.1.2.5.3.2 Activity: Develop Installation Artifacts**
- 998 - Produce all the software required to install and uninstall the product quickly, easily  
999 and safely without affecting other applications or system characteristics
- 1000 **1.1.2.5.3.2.1 Role**
- 1001 Implementer
- 1002 **1.1.2.5.3.2.2 Artifact**
- 1003 The results should be recorded in Installation Artifacts.  
1004

1005 **1.1.2.5.3.3 Activity: Create Deployment Unit (Web service)**

- 1006 - Create a deployment unit that is sufficiently complete to be downloadable, installable
- 1007 and run on a node as a group
- 1008 - A deployment unit consists of a build (an executable collection of components),
- 1009 documents (end-user support material and release notes) and installation artifacts

1010 **1.1.2.5.3.3.1 Role**

1011 Configuration Manager

1012 **1.1.2.5.3.3.2 Artifact**

1013 The results should be recorded in Deployment Unit.

1014

1015 **1.1.2.5.3.4 Activity: Deploy Web Service to identified app servers**

- 1016 - Identify the service end-point
- 1017 - Create the deployment script and configure the deployment parameters
- 1018 - Deploy the Web service
- 1019 - Generate the WSDL

1020 **1.1.2.5.3.4.1 Role**

1021 Implementer

1022 **1.1.2.5.3.4.2 Artifact**

1023 The results should be recorded in Installation Artifacts.

1024

1025 **1.1.2.5.3.5 Activity: Publish Web service [optional]**

- 1026 - Identify the UDDI registry
- 1027 - Prepare the information needed for publishing
- 1028 - Publish the Web service in the UDDI registry
- 1029 - Perform a find and bind to test

1030 **1.1.2.5.3.5.1 Role**

1031 Implementer

1032 **1.1.2.5.3.5.2 Artifact**

1033 The results should be recorded in Installation Artifacts.

1034

---

1035 **2 References**

1036 **2.1 Normative**

1037 1.

1038 **2.2 Non-Normative**

1039 1. "Rational Unified Process", Version 2003.06.00.65, IBM-Rational Software.

1040

1041 2. "Rational Unified Process for Developing Web Services", Version 1.0, Java Smart  
1042 Services Laboratory and Rational Software Pte. Ltd., Aug 2003.

1043

---

## Appendix A. Acknowledgments

1044 The following individuals were members of the committee during the development of this  
1045 documentation:

- 1046 • Ravi Shankar, CrimsonLogic Pte. Ltd.
- 1047 • Jagdip Talla, CrimsonLogic Pte. Ltd.
- 1048 • Andy Tan, Individual
- 1049 • Roberto Pascual, IDA
- 1050

1051

---

## Appendix B. Revision History

Rev	Date	By Whom	What
wd-01	2004-09-30	Lai Peng CHAN Chai Hong ANG	Initial version
	2004-12-23	Chai Hong ANG	Split the original document into two

1052

---

## Appendix C. Notices

1053

1054 OASIS takes no position regarding the validity or scope of any intellectual property or other  
1055 rights that might be claimed to pertain to the implementation or use of the technology  
1056 described in this document or the extent to which any license under such rights might or might  
1057 not be available; neither does it represent that it has made any effort to identify any such  
1058 rights. Information on OASIS's procedures with respect to rights in OASIS specifications can  
1059 be found at the OASIS website. Copies of claims of rights made available for publication and  
1060 any assurances of licenses to be made available, or the result of an attempt made to obtain a  
1061 general license or permission for the use of such proprietary rights by implementors or users  
1062 of this specification, can be obtained from the OASIS Executive Director.

1063 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
1064 applications, or other proprietary rights which may cover technology that may be required to  
1065 implement this specification. Please address the information to the OASIS Executive Director.

1066 **Copyright © OASIS Open 2004. All Rights Reserved.**

1067 This document and translations of it may be copied and furnished to others, and derivative  
1068 works that comment on or otherwise explain it or assist in its implementation may be  
1069 prepared, copied, published and distributed, in whole or in part, without restriction of any kind,  
1070 provided that the above copyright notice and this paragraph are included on all such copies  
1071 and derivative works. However, this document itself does not be modified in any way, such as  
1072 by removing the copyright notice or references to OASIS, except as needed for the purpose  
1073 of developing OASIS specifications, in which case the procedures for copyrights defined in  
1074 the OASIS Intellectual Property Rights document must be followed, or as required to translate  
1075 it into languages other than English.

1076 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
1077 successors or assigns.

1078 This document and the information contained herein is provided on an "AS IS" basis and  
1079 OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT  
1080 LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL  
1081 NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY  
1082 OR FITNESS FOR A PARTICULAR PURPOSE.

1083