

P r o f e s s i o n a l   E x p e r t i s e   D i s t i l l e d

# Oracle BPM Suite 12c Modeling Patterns

Design and implement highly accurate Business Process  
Management solutions with Oracle BPM Patterns

**Vivek Acharya**

**[PACKT]** enterprise   
PUBLISHING professional expertise distilled

# Oracle BPM Suite 12c Modeling Patterns

Design and implement highly accurate Business  
Process Management solutions with Oracle  
BPM Patterns

**Vivek Acharya**



BIRMINGHAM - MUMBAI

# Oracle BPM Suite 12c Modeling Patterns

Copyright © 2014 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: September 2014

Production reference: 1220914

Published by Packt Publishing Ltd.  
Livery Place  
35 Livery Street  
Birmingham B3 2PB, UK.

ISBN 978-1-84968-902-1

[www.packtpub.com](http://www.packtpub.com)

Cover image by Artie Ng ([artherng@yahoo.com.au](mailto:artherng@yahoo.com.au))

# Credits

**Author**

Vivek Acharya

**Project Coordinator**

Kinjal Bari

**Reviewers**

Cyril Brigant

Haitham A. El-Ghareeb

Jaideep Ganguli

Ramakrishna Kandula

Max Pellizzaro

Surendra Pepakayala

**Proofreaders**

Simran Bhogal

Mario Cecere

Maria Gould

Paul Hindle

Chris Smith

**Indexers**

Mariammal Chettiyar

Monica Ajmera Mehta

Rekha Nair

Tejal Soni

**Acquisition Editor**

Nikhil Karkal

**Content Development Editor**

Rikshith Shetty

**Technical Editors**

Menza Mathew

Akash Rajiv Sharma

**Graphics**

Ronak Dhruv

Valentina D'silva

Abhinash Sahu

**Copy Editors**

Roshni Banerjee

Dipti Kapadia

Karuna Narayanan

Stuti Srivastava

**Production Coordinators**

Melwyn D'sa

Manu Joseph

**Cover Work**

Melwyn D'sa

# Disclaimer

The views expressed in this book are my own and do not reflect the views of Oracle Corporation or the company (or companies) I work (or have worked) for.

The information in this book is written based on personal experiences. You are free to use the information in this book, but I am not responsible and will not compensate you if you ever happen to suffer a loss/inconvenience/damage because of/while making use of this information.

This book is designed to provide information on BPM Patterns only. This information is provided and sold with the knowledge that the publisher and author do not offer any legal or professional advice. In case of a need for any such expertise, consult with the appropriate professional.

This book does not contain all the information available on the subject. Every effort has been made to make this book as accurate as possible. However, there may be typographical and/or content errors. Therefore, this book should serve only as a general guide and not as the ultimate source of subject information.

Furthermore, this manual contains information on writing and publishing that is current only up to the printing date.

# About the Author

**Vivek Acharya** is an Oracle BPM and Fusion Middleware Applications professional and works for Oracle Corporation, USA. He has been in the world of design, development, consulting, and architecture for approximately 10 years. He is an Oracle Certified Expert, an Oracle Fusion SOA 11g Implementation Specialist, and Oracle BPM 11g Implementation Specialist. He has experience working with Oracle Fusion Middleware and Fusion Applications. He loves all the things associated with Oracle Fusion Applications, Oracle BPM/SOA, Cloud and SaaS, predictive analytics, social BPM, and adaptive case management. He has been the author for a couple of books, has an interest in playing the synthesizer, and loves travelling. You can add him on LinkedIn at <http://www.linkedin.com/pub/vivek-acharya/15/377/26a>, write to him on [vivek.oraclesoa@gmail.com](mailto:vivek.oraclesoa@gmail.com), or reach him at <http://acharyavivek.wordpress.com/>.

# Acknowledgments

First and foremost, I would like to thank God. I could never have done this without the faith that I have in Him, the Almighty.

No one walks alone, and when one is walking the journey of life, you begin to thank those who joined you, walked beside you, and helped you along the way.

Many thanks to mom, papa, and my brother, Alankar; you all have been supreme. You have nurtured my learning and have always stood by me when things were odd or even. Thanks to my in-laws for giving wings to Richa.

Huge thanks to my wife, Richa, for inspiring me at every step, supporting my efforts, and encouraging me through the long journey. Thanks for having patience with me when I was facing yet another challenge in my life that reduced the amount of time I spent with you, and your sacrifice of all those weekends and vacations.

I would like to express my gratitude to Bill Swenton for all his support. I would like to take this opportunity to thank all those with whom I have worked in the past and those who have inspired me in one way or the other. Many thanks to Dean Welch, Vijay Navaluri, Prakash Devarakonda, Sebastiaan Dammann, Monique Albrecht, Nader Svärd, and Jugni for inspiring me.

Thanks to the reviewers who worked on this book. I would like to thank Rikshith Shetty, Binny Babu, Navu Dhillon, Larissa Pinto, Anthony Albuquerque, Menza Mathew, Akash Rajiv Sharma, and all the members of the Packt Publishing team for editing and polishing the book.

Last but not least, I beg forgiveness from all those who have been with me over the course of all these years and whose names I have failed to mention.

# About the Reviewers

**Cyril Brigant** is an application architect specialized in BPM modeling and SOA Architecture. He has been involved in various workflow projects and SOA initiatives for the last 10 years in various Enterprise environments. At the European Commission, DG RTD, he was in charge of modeling the Enterprise architecture and SOA governance. He has recently joined the CMA CGM Group to bring his expertise for an ambitious project to rebuild the complete information system based on the top-down approach.

**Haitham A. El-Ghareeb** is an Associate Professor at the Information Systems Department, Faculty of Computers and Information Sciences, Mansoura University, Egypt. He is a member of many distinguished computer organizations, reviewer for different highly recognized academic journals, contributor to open source projects, and the author of different books.

Haitham is interested in e-learning, Enterprise architecture, information architecture, and software architecture, especially in Service-Oriented Architecture (SOA), Business Process Management (BPM), Business Process Management Systems (BPMS), Information Storage and Management, Virtualization, Cloud Computing, Big Data, and in collaboration with Information Systems and e-learning organizations and researchers.

Haitham holds a Master of Science degree (in 2008) from the same faculty that he is currently working for. His thesis was titled *Evaluation of Service Oriented Architecture in e-Learning*. This thesis was highly recognized and has been published as an international book under the same title (ISBN-13: 978-3-83835-538-2). He holds a PhD degree (in 2012) from the same faculty. His PhD dissertation was titled *Optimizing Service Oriented Architecture to Support e-Learning with Adaptive and Intelligent Features*, which was highly recognized and has been published as an international book under the title, *Optimizing Service Oriented Architecture to Support e-Learning*, LAP Lambert Academic Publishing, (ISBN-13: 978-3-84731-187-4).



Haitham is the author of the book, *Enterprise Integration Opportunities and Challenges*, LAP Lambert Academic Publishing, (ISBN-13: 978-3-65937-179-0). For an updated list of Haitham's activities and research articles, please consider the following websites:

- Haitham's personal website: <http://www.helghareeb.me>
- Haitham's blog: <http://blog.helghareeb.me>
- Haitham's channel on YouTube: <http://video.helghareeb.me>

**Jaideep Ganguli** has more than 20 years of experience in developing software solutions for several domains, including financial services, e-government, criminal justice, and wireless application services. Over the last 10 years, he has delivered several Enterprise-scale solutions based on WebLogic, JEE, Oracle BPM, SOA Suite, ADF, and WebCenter. He is a Certified Implementation Specialist with Oracle WebCenter Portal 11g and Oracle BPM 11g.

Currently, Jaideep is one of the cofounders and partners of Fusion Applied ([www.fusionapplied.com](http://www.fusionapplied.com)). Fusion Applied offers top-notch Oracle Fusion Middleware-focused consulting and training.

Jaideep holds an MBA degree from Johns Hopkins University and a BS in Electronics Engineering from Mumbai University, India.

He can be contacted at <http://www.linkedin.com/in/jaideepganguli/>, or you can e-mail him at [jaideep@fusionapplied.com](mailto:jaideep@fusionapplied.com).

---

I'd like to thank my wife, Rajeshwari, for her patience and support and my partners, Vivek Chaudhari, Vikram Bailur, and Sanjib Rajbhandari, for their encouragement and technical expertise.

---

**Ramakrishna Kandula** has an experience of 10 years in the IT sector and 7 years with Fusion Middleware technologies. He has worked on different projects in SOA and BPM Suites with various clients.

He completed his graduation with a Bachelor's of Technology degree in Electronics and Communication, in 2003.

He has received many accolades and awards in his career from client and internal organization recognition events for key implementations and innovative approach design.

He has designed and implemented many B2B and EAI architectures for different business implementations, which have also become role model architectures for many other implementations.

He has technically reviewed *Oracle BPM Suite 11g Developer's Cookbook*, Packt Publishing.

You can e-mail him at [ramakrishna.rpkandula@gmail.com](mailto:ramakrishna.rpkandula@gmail.com).

**Max Pellizzaro**, with over 15 years of working experience, has been working as a software/IT consultant in complex projects within different industries: automotive, telecommunication, and entertainment and media. Within the projects he has been involved with, he has developed experience throughout all the phases of a project's life cycle: collecting user requirements, designing software solutions, leading software development, and designing monitor tools for the on-going production environment. In his last organization, Max was the leading architect of Center of Excellence of Oracle Technology. His main activity was to understand clients' needs to help him drive the design and prototype of Oracle Solutions.

Max loves technologies; even in his day-to-day jobs, he mainly deals with Oracle technology. His passion has brought him to learn other technologies such as mobile development, game development, and 3D development.

Max has contributed to the review of other books on XML technology and open source frameworks.

**Surendra Pepakayala** is a seasoned technology professional and entrepreneur with over 16 years of experience in the US and in India. He has a broad experience in building enterprise software products for both startups and multinational companies.

After 11 years in the corporate industry, Surendra founded an enterprise software product company based in India. He subsequently sold the company and started Cloud computing, Big Data, and Data Science Consulting practice.

Surendra has reviewed drafts, recommended changes, and formulated questions for various IT certification literature/tests, such as CGEIT, CRISC, MSP, and TOGAF.

# www.PacktPub.com

## Support files, eBooks, discount offers, and more

You might want to visit [www.PacktPub.com](http://www.PacktPub.com) for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at [www.PacktPub.com](http://www.PacktPub.com) and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at [service@packtpub.com](mailto:service@packtpub.com) for more details.

At [www.PacktPub.com](http://www.PacktPub.com), you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<http://PacktLib.PacktPub.com>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

## Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

## Free access for Packt account holders

If you have an account with Packt at [www.PacktPub.com](http://www.PacktPub.com), you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

## Instant updates on new Packt books

Get notified! Find out when new books are published by following [@PacktEnterprise](#) on Twitter, or the *Packt Enterprise* Facebook page.



# Table of Contents

<b>Preface</b>	<b>1</b>
<b>Chapter 1: Flow Control Patterns</b>	<b>9</b>
<b>Sequence flow pattern</b>	<b>10</b>
Working with the sequence flow pattern	12
Elucidating the sequence flow pattern	14
<b>Getting ready for executing use cases</b>	<b>14</b>
<b>Exclusive choice and simple merge pattern</b>	<b>16</b>
Working with exclusive choice and simple merge pattern	18
Knowing about the exclusive choice pattern	21
Elucidating the simple merge pattern	22
<b>Multichoice and synchronizing merge pattern</b>	<b>22</b>
Demonstrating multichoice and synchronization with the OR gateway	23
The working of multichoice and synchronization pattern	26
Structured synchronizing merge pattern	26
Local synchronizing merge pattern	27
<b>The parallel split and synchronization pattern</b>	<b>28</b>
Parallel split pattern	28
Synchronization pattern	29
<b>Conditional parallel split and parallel merge pattern</b>	<b>32</b>
Working with conditional parallel split and merge	33
Antipattern – the conditional parallel split and merge	35
<b>Multimerge pattern</b>	<b>36</b>
Exploring multimerge	38
<b>Discriminator and partial join pattern</b>	<b>40</b>
Structured discriminator pattern	41
Structured partial join	42
Working with a complex gateway to implement the discriminator and partial join pattern	43

Testing a process by failing the complex gateway exit expression	44
Testing process as success by the complex gateway exit expression	44
<b>Complex synchronization pattern</b>	<b>45</b>
Canceling discriminator pattern	46
Canceling partial join pattern	47
<b>Summary</b>	<b>48</b>
<b>Chapter 2: Multi-instance and State-based Patterns</b>	<b>49</b>
<b>Multiple instances with prior design-time knowledge pattern</b>	<b>50</b>
Executing the multi-instance subprocess with prior design-time knowledge	51
<b>Multiple instances with prior runtime knowledge pattern</b>	<b>54</b>
Demonstrating MI with prior runtime knowledge	55
Understanding how MI with prior runtime knowledge work	57
<b>Multiple instances without prior runtime knowledge pattern</b>	<b>57</b>
Working on MI without prior runtime knowledge	58
Testing the use case	60
<b>Static partial join for multiple instances pattern</b>	<b>62</b>
Testing the use case	64
Understanding how static partial join for MI works	66
There's more	66
<b>Canceling partial join pattern</b>	<b>66</b>
<b>Dynamic partial join for multiple instances pattern</b>	<b>67</b>
Working with dynamic partial join	68
Understanding the functionality behind partial join for MI	69
<b>Structured loop pattern</b>	<b>69</b>
Working with structured loops	70
Demystifying do-while	70
Demystifying while-do	72
<b>Arbitrary cycle pattern</b>	<b>72</b>
Exploring arbitrary cycle	73
Understanding the functionality of the arbitrary cycle	76
<b>Trigger patterns</b>	<b>76</b>
Transient trigger pattern	76
Persistent trigger pattern	77
<b>Implicit termination pattern</b>	<b>78</b>
Amalgamating implicit termination in the process flow	78
<b>Explicit termination pattern</b>	<b>79</b>
Learning how explicit termination works	79
<b>Cancellation patterns</b>	<b>80</b>
Cancel multi-instance task pattern	80
<b>Summary</b>	<b>83</b>

<b>Chapter 3: Invocation Patterns</b>	<b>85</b>
<b>Web service pattern</b>	<b>86</b>
Asynchronous request-response (request-callback) pattern	87
Request-response pattern	90
One request, one of the two possible responses pattern	92
Two request a pattern	94
Exposing the BPM process using Receive and Send Tasks	97
Loan Origination over Send and Receive tasks	97
<b>One-way invocation pattern</b>	<b>99</b>
Implementing one-way invocation using a timer	100
Implementing one-way invocation using an e-mail	102
The Loan Origination process over e-mail	103
Testing the flow to instantiate a process over e-mail	105
<b>Publish-subscribe pattern – initiating the business process through an event</b>	<b>105</b>
Loan origination over an event	107
<b>Multievent instantiation pattern – process instantiation over multiple events</b>	<b>111</b>
Loan origination over multiple event occurrence	111
<b>Human task initiator pattern – initiating processes through human tasks</b>	<b>113</b>
Loan origination via the human task form	114
Testing the process	116
<b>Guaranteed delivery pattern – process instantiation over JMS – Queue/Topic</b>	<b>117</b>
Loan origination over JMS – Queue/Topic	119
Creating JMS resources	120
Creating the publisher process	124
Developing the consumer process	124
Testing the process	126
<b>Understanding multiple start events</b>	<b>128</b>
<b>Summary</b>	<b>129</b>
<b>Chapter 4: Human Task Patterns</b>	<b>131</b>
<b>Learning about human tasks</b>	<b>133</b>
<b>Milestone pattern</b>	<b>136</b>
Modeling in a human task versus a BPMN process	139
<b>Routing pattern</b>	<b>139</b>
<b>Task assignment pattern</b>	<b>140</b>
<b>List builder pattern</b>	<b>142</b>
Absolute or nonhierarchical list builders	143
Hierarchical list builders	144
Rule-based list builders	145



<b>Parallel routing pattern</b>	<b>147</b>
Getting ready to test sample use cases	147
Parallel routing pattern with name and expression list builders	148
Parallel routing pattern with approval group list builder	152
Parallel routing pattern with lane participant list builder	153
Parallel routing pattern with rule-based list builder	154
Parallel routing pattern with management chain	156
<b>Serial routing pattern</b>	<b>158</b>
Serial routing pattern with list builder – name and expression	158
Participant identification type – users	158
Participant identification type – groups	159
Participant identification type – application role	159
Serial routing pattern with list builder – approval group	159
Serial routing pattern with list builder – management chain	160
Serial routing pattern with list builder – job level	160
Modifying participant lists using list modification	162
Substituting participants using list substitution	162
Serial routing pattern with list builder – position	163
Serial routing pattern with list builder – supervisory	164
Serial routing pattern with list builder – rules	165
<b>Single routing pattern</b>	<b>165</b>
Single approver pattern with list builder – name and expression	166
Single approver pattern with list builder – approval group	166
Single approver pattern with list builder – management chain	166
<b>Notify/FYI pattern</b>	<b>166</b>
FYI approver pattern with list builder – job level	167
FYI approver pattern with list builder – name and expression	167
<b>Task aggregation pattern</b>	<b>167</b>
<b>Dispatching pattern</b>	<b>170</b>
<b>Escalation pattern</b>	<b>171</b>
<b>Rule-based reassignment and delegation pattern</b>	<b>172</b>
<b>Ad hoc routing pattern</b>	<b>173</b>
<b>Request information feature</b>	<b>175</b>
<b>Reassignment and delegation pattern</b>	<b>177</b>
<b>Force completion pattern</b>	<b>178</b>
Enabling early completion in parallel subtasks	180
<b>Routing rule pattern</b>	<b>180</b>
<b>Deadlines</b>	<b>182</b>
<b>Escalation, expiry, and renewal feature</b>	<b>186</b>
<b>Exclusion feature</b>	<b>190</b>
<b>Error assignee and reviewers</b>	<b>190</b>
<b>Notifications</b>	<b>192</b>

Configuring driver properties and attributes	193
Configuring the notification definition	194
<b>Content access policy and task actions</b>	<b>196</b>
<b>Enterprise content management for task documents</b>	<b>197</b>
<b>Summary</b>	<b>199</b>
<b>Chapter 5: Interaction Patterns</b>	<b>201</b>
<b>Defining use cases to demonstrate interaction patterns</b>	<b>202</b>
The BackOffice process	202
The Loan origination process	203
The CatchFraudDetails and Feedback processes	203
<b>Conversation pattern</b>	<b>207</b>
<b>Asynchronous interaction pattern</b>	<b>211</b>
Interacting with an asynchronous process using the Message	
Throw and Catch events	213
Interacting with an asynchronous service using the Message	
Throw and Catch Events	216
Enabling external services interaction	217
Interacting with an asynchronous process and service using Send and Receive Tasks	219
Attaching boundary events on Send and Receive Tasks	221
Interacting with a process defined with Receive Task as a start activity	222
<b>Synchronous request-response pattern</b>	<b>224</b>
The business catalog	226
<b>Subprocess interaction patterns</b>	<b>227</b>
Reusable process interaction pattern	229
Use case scenario for reusable process interaction pattern	231
<b>Embedded subprocess interaction pattern</b>	<b>232</b>
Interrupting a boundary event	234
Boundary event on an activity	234
<b>Event-driven interaction pattern</b>	<b>236</b>
Defining an event-based interaction pattern scenario	238
<b>Summary</b>	<b>240</b>
<b>Chapter 6: Correlation Patterns</b>	<b>241</b>
<b>Correlation mechanism</b>	<b>242</b>
Types of correlations	242
Components of correlation	243
Configuring the environment	244
Defining correlation properties	246
Defining correlation keys and configuring the correlation definition	247
Understanding the correlation behavior	249

<b>Message-based correlation pattern</b>	<b>250</b>
Testing the message-based correlation pattern	256
<b>Cancel instance pattern</b>	<b>258</b>
Understanding the components	259
Testing cancelation pattern	261
Restart instance pattern	262
Testing the Loan Origination process to restart a loan	263
Testing the restart scenario	264
<b>Update task pattern</b>	<b>266</b>
Demonstrating the update task functionality	268
<b>Query pattern</b>	<b>268</b>
Testing the query pattern	270
<b>Suspend process pattern</b>	<b>272</b>
<b>Suspend activity pattern</b>	<b>274</b>
<b>Cancel activity pattern</b>	<b>275</b>
How a boundary event based activity correlation works	276
Testing the cancelation pattern on an activity	277
<b>Summary</b>	<b>278</b>
<b>Chapter 7: Exception Handling Patterns</b>	<b>279</b>
<b>Classifying exceptions</b>	<b>280</b>
<b>Business process state</b>	<b>281</b>
<b>Reassigned Exception Handling Pattern</b>	<b>284</b>
<b>Allocated Exception Handling Pattern</b>	<b>285</b>
Changing the Boundary Catch Event from Interrupting to Noninterrupting	289
<b>Force-Terminate Exception Handling Pattern</b>	<b>292</b>
<b>Force-Error Exception Handling Pattern</b>	<b>293</b>
<b>Force-Complete Exception Handling Pattern</b>	<b>295</b>
<b>Invoked Exception Handling Pattern</b>	<b>296</b>
<b>Invoked State Exception Handling Pattern</b>	<b>297</b>
<b>Continue Execution Exception Handling Pattern</b>	<b>299</b>
<b>Force-Terminate Execution Exception Handling Pattern</b>	<b>302</b>
<b>Force-Error Execution Exception Handling Pattern</b>	<b>303</b>
Allocated state – External Exception Handling Pattern	304
Implementing Allocated state – External Exception Handling Pattern	306
Allocated state – Internal Exception Handling Pattern	309
Implementing Allocated state – Internal Exception Handling Pattern	309
Reallocated Exception Handling Pattern	313
<b>External Exception Handling Pattern</b>	<b>314</b>
<b>Process-Level Exception Handling Pattern</b>	<b>314</b>
Implementing Process-Level Exception Handling Pattern	315

Testing Process-Level Exception Handling Pattern	317
<b>System-Level exception handling pattern</b>	<b>318</b>
<b>External Triggers</b>	<b>318</b>
<b>Summary</b>	<b>319</b>
<b>Chapter 8: Adaptive Case Management</b>	<b>321</b>
<b>Defining adaptive case management</b>	<b>322</b>
Case	323
Case management	323
Dynamic case management	323
Mechanism of adaptive case management	324
Process versus case	325
Case management offerings	325
The building blocks of adaptive case management	327
<b>Exploring ACM use case scenarios</b>	<b>329</b>
The building blocks of the Insurance Claim use case	332
Testing the use case	333
<b>Case stage</b>	<b>336</b>
<b>Event pattern</b>	<b>338</b>
<b>Milestone pattern</b>	<b>341</b>
<b>Case interaction pattern</b>	<b>344</b>
<b>Localization feature</b>	<b>345</b>
<b>Holistic view pattern</b>	<b>346</b>
<b>Ad hoc feature</b>	<b>348</b>
Ad hoc inclusion of stakeholders	349
Ad hoc inclusion of activities	349
Ad hoc inclusion of documents	350
Association of a case with subcases	350
Ad hoc inclusion of rules and activities	351
<b>Summary</b>	<b>352</b>
<b>Chapter 9: Advanced Patterns</b>	<b>353</b>
<b>Strategic Alignment Pattern</b>	<b>354</b>
The Value Chain Model	357
The Strategy Model	361
Mapping goals to an organization	363
Defining KPIs in a BPMN project	363
Defining KPIs in a BA project	365
Defining KPIs in a child Value Chain Model	365
Defining KPIs in the master Value Chain Model	368
Publishing report data	370
<b>Capturing the business context</b>	<b>372</b>

*Table of Contents*

---

<b>Emulating Process Behavior</b>	<b>377</b>
<b>The Debugger feature</b>	<b>381</b>
<b>Round Trip and Business-IT Collaboration</b>	<b>383</b>
<b>Summary</b>	<b>392</b>
<b>Appendix: Installing Oracle BPM Suite12c</b>	<b>393</b>
<b>Installing JDK</b>	<b>393</b>
<b>Installing BPM suite</b>	<b>394</b>
<b>Configuring the default domain</b>	<b>397</b>
<b>Enabling the demo user community</b>	<b>399</b>
<b>Custom domain creation</b>	<b>402</b>
<b>The BPM/SOA configuration</b>	<b>407</b>
<b>Summary</b>	<b>412</b>
<b>Index</b>	<b>413</b>

---

# Preface

This book demonstrates the perceptible regularity in the world of BPMN design and implementation while diving into the comprehensive learning path of the much-awaited Oracle BPM modeling and implementation patterns, where, the readers will discover the doing rather than reading about the doing and this book, *Oracle BPM Suite 12c Modeling Patterns*, effectively demonstrates the doing. The scope of this book covers the patterns and scenarios from flow patterns to strategic alignment (goals and strategy model) – from conversation, collaboration, and correlation patterns to exception handling and management patterns; from human task patterns to asset management; from business-IT collaboration to adaptive case management; and much more.

This book will demystify various patterns that have to be followed while developing a professional BPM solution. The patterns such as split-join, multi-instance, loop, cycle, termination, and so on, allow you to drill into basic and advanced flow-based patterns. The integration, invocation, interaction, and correlation patterns demonstrate collaboration and correlation of BPM with other systems, processes, events and services. The human interaction pattern section leaves no stone unturned in covering task modeling, routing, dispatching, dynamic task assignment, rule-based assignments, list building, and other advanced topics. The chapter on Exception Handling Pattern is a comprehensive guide to model and implement exception handling in Oracle BPM implementation and design. The chapter on Adaptive Case Management offers detailed information about patterns handling unstructured data and unpredictable scenarios. The adaptive case management features and patterns will empower you to develop a milestone-oriented, state-based, rule-governed, content outbid, event-driven, and case management solution. Also, the witness patterns bring enhanced and dynamic business-IT collaboration. Experience the magic of strategic alignment features, which brings together the requirement and analysis gaps and makes the organizational activities very much in-line with the goals, strategies and objectives, KPIs, and reports.

This is an easy-to-follow yet comprehensive guide to demystify strategies and best practices to develop BPM solutions on the Oracle BPM 12c platform. All patterns are complemented with code examples to help you better discover how patterns work. The real-life scenarios and examples touch many facets of BPM, whereas solutions are a comprehensive guide to various BPM modeling and implementation challenges. Each pattern pairs the classic problem/solution format, which includes signature, intent, motivation, applicability, and implementation, where implementation is demonstrated via a use case scenario along with a BPMN application with each chapter.

## What this book covers

*Chapter 1, Flow Control Patterns*, covers the basic flow control patterns in BPMN. This chapter offers an exemplary and comprehensive exposure to flow control patterns that are helpful in modeling and implementing BPMN solutions. During the course of modeling from "As-Is" to "To-Be" process, a process analyst models, designs, drafts, and publishes a sequence of activities and their flow control. This chapter starts off by showcasing the essentials of flow control patterns. This chapter explains converging from conditional and unconditional sequence flow to simple and parallel split and merge; later, the flow in this chapter expands to multi merge and transitioning patterns. Then, there is a comprehensive guide to patterns such as the partial join and discriminator patterns.

*Chapter 2, Multi-instance and State-based Patterns*, discusses a set of patterns that will demonstrate how processes can handle batch jobs and simultaneously spawn multiple work item instances in a process. This chapter simplifies the usage of loop characteristics while showcasing multi-instance perspectives. This chapter emphasizes on developing solutions for use cases with multi-instance requirements with design time and run time knowledge. This chapter further covers iteration patterns by demonstrating structured loop and unstructured looping mechanism. Then, implicit and explicit termination patterns will showcase the termination pattern.

*Chapter 3, Invocation Patterns*, gives an insight into the various discrete mechanisms to initiate processes and this chapter covers various patterns that illustrate these discrete invocation patterns. Process interfacing offers other processes, services, and external systems to communicate with BPM processes. This chapter uncovers process interfacing with queues, services, and processes by exposing different operations which external systems can interact with.

*Chapter 4, Human Task Patterns*, discusses the patterns and features that offer formalized best practices and solutions for the commonly occurring issues and challenges that allow process analysts, developers, and designers to build solutions to bring in human intuition in the process. This chapter discusses various task flow

patterns and also demonstrates working with complex task flow. This chapter also demonstrates the inclusion of business rules to build a dynamic participant list. This chapter covers patterns that allow you to explore the feasibility to build a participant list statically, dynamically, or based on rules. The task assignment patterns section demonstrates how tasks are assigned statically, dynamically, or based on rules to the participants. The ad hoc assignment patterns, delegation patterns, and escalation patterns give depth to the chapter. The various other advanced features such as exclusion, notification, ECM integration, access policy, and so on are covered in detail along with elaboration on routing patterns, delegation, and so on.

*Chapter 5, Interaction Patterns*, discusses how processes interact and integrate with other systems, processes, and services and how these interactions are facilitated by various interaction patterns. This chapter includes various patterns that help to communicate with other processes, systems, and services. This chapter focuses on patterns that facilitate collaborative interaction of process with other processes, service, events, and signals.

*Chapter 6, Correlation Patterns*, showcases patterns that offer solutions to scenarios where processes need to be interrupted on the fly and sometimes need to be cancelled. The solution to a scenario where a task needs to be changed and/or updated in an in-flight process or cases such as querying an in-flight process. This chapter also uncovers all those patterns that need to interact with an in-flight process and also will explain how we can relate processes and associate a message with the conversation that it belongs to. The much awaited 12c features include suspending process and activities. These are elaborated in the chapter along with various other patterns to cancel, update, and query a process or activity.

*Chapter 7, Exception Handling Patterns*, focuses on demystifying various Exception Handling Patterns. This chapter focuses on exception classification, exception propagation, exception handling mechanism, and fault management framework. This chapter explains the strategies of how exceptions are handled in Oracle BPMN with detailed coverage of the fault management framework. We will examine the handling of exceptions in tasks, subprocess, and processes while covering different categories of faults. We will also cover modeling for exception handling and various modeling best practice while taking care of exception handling. Though the chapter is focused on exception handling patterns, it covers various exception handling mechanisms, their implementation, and usage in Oracle BPM.



*Chapter 8, Adaptive Case Management*, focuses on the case management framework that enables building case management applications, which comprise business processes, human interaction, decision making, data, collaboration, events, documents, contents, rules, policies, reporting, and history. This chapter demonstrates the inclusion of human intuition, empowered case, knowledge workers, collaborative decision-making, enhanced content management, and social collaboration. This chapter elaborates on Oracle Adaptive Case Management solution and in the course of learning it, one can explore various patterns and features that enable designers, developers, and analysts to model case management solutions and bring in agility, true dynamism, collaborative decision making, and a 360-degree holistic view of the case. This chapter also covers milestone patterns, case framework, event patterns, localization, case states, case interaction patterns, holistic view, and ad hoc features.

*Chapter 9, Advanced Patterns*, covers patterns in analysis and discovery category, where alignment patterns demonstrates features such as analyze, refine, define, optimize and report, and business processes in the enterprise. Alignment patterns highlight how IT development and process models can be aligned with organization goals while performing alignment, learning enterprise maps, strategy models, value chain models, KPIs, and reports. This chapter will also show how to create different reports based on the information documented in the process such as RACI reports, and so on. This chapter heavily focuses on demonstrating round trips and business IT collaboration, which facilitates storing, sharing, and collaborating on process assets and business architecture assets. This chapter also focuses on creating a collaborative ecosystem for business and IT and a detailed analysis of PAM methods to emulate the process behavior.

*Appendix, Installing Oracle BPM Suite12c*, gives us a brief introduction to the technology used in the book and also lists the steps to install Oracle BPM. Perform the steps given in this appendix to install Oracle BPM 12c to implement the use cases demonstrated for each pattern in this book.

## What you need for this book

To explore modeling and implementation patterns and various features of BPM 12c through recipes in this book, you need the following software installed in your development environment:

- JDK 1.7.0\_15 or higher
- Oracle BPM Suite Downloads 12c (12.1.3)
- Oracle Database XE (11g)

The detailed steps to set up the environment are included in *Appendix, Installing Oracle BPM Suite12c*.

The important considerations that should be taken care of are as follows:

- Tool/IDE JDeveloper 12c to develop solutions should be a part of the 12c BPM installation
- The installation document (*Appendix, Installing Oracle BPM Suite12c*) contains two methods to install the database; follow the one that suits your development requirements the most. It's a quick installation guide.

## Who this book is for

This book is an invaluable resource for enterprise architects, solution architects, developers, process analysts, application functional and technical analysts, consultants, and all those who use business process and BPMN to model and implement enterprise IT applications, SaaS, and cloud applications. The primary focus is to showcase BPM patterns which are generic and can be read by anyone allied with any BPM offering. Hence, if you are associated with BPMN, you can relate to this title.

## Conventions


In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.


Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "This is a static approval group defined in the BPM workspace with users (Christine, salesrep, Jim, and Kim)."

A block of code is set as follows:

```
If Discount < 10% then
Process performs other activity and process ends.
Else-if Discount > 50%
Accept Quote task is revisited by salesrep user.
Else-if Discount > 10% and Discount < 50%
Sales Manager Approval task is initiated.
```

**New terms** and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Now, click on the sequence flow with the **Deal** or **Terms Reject** tag and check its properties."

[  Warnings or important notes appear in a box like this. ]

[  Tips and tricks appear like this. ]

## Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book – what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to [feedback@packtpub.com](mailto:feedback@packtpub.com), and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on [www.packtpub.com/authors](http://www.packtpub.com/authors).

## Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

## Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

## Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

## Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at [copyright@packtpub.com](mailto:copyright@packtpub.com) with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

## Questions

You can contact us at [questions@packtpub.com](mailto:questions@packtpub.com) if you are having a problem with any aspect of the book, and we will do our best to address it.



# 1

## Flow Control Patterns

A pattern is a generic solution to a recurring problem. Patterns describe a problem and its solution, which can be adopted in discrete situations. Patterns are adorned best practices that deliver a reusable architecture outline. **Business Process Management (BPM)** is widely adopted for process transparency, process intelligence, business empowerment, and business alignment. While designing business processes, we are not just automating and managing processes; it's more about how an enterprise adapts to a comprehensive view of business processes.

This chapter offers an exemplary and comprehensive exposure to flow control patterns, which are helpful in the modeling and implementation of Oracle BPM 12c solutions. During the journey, it will walk you through various BPM patterns based on real-life examples. The book offers projects to download with each chapter; these projects allow you to design, model, and analyze the patterns discussed in each chapter. Hence, it offers an interactive way to learn and implement BPM patterns. It allows you to fill the gaps and offers content that allows you to use BPMN to its full potential.

Process analysts, architects, and process developers deal with process modeling, define and design process models, and implement them. While performing process modeling and implementing them, they constantly deal with varied common challenges. Process modeling and BPM patterns offer techniques to solve repeatable issues, enhance the process-modeling approach, improve process modeling and implementation quality, and offer great productivity.

This chapter covers the basic and advanced flow control patterns in Oracle BPM. Perceptible regularity in the world of process control flow is demonstrated here. During the course of modeling from the "As-Is" to "To-Be" process, a process analyst models, designs, drafts, and publishes a sequence of activities and their flow control. This chapter starts off the book by showcasing the essentials of flow control patterns. Flow control patterns capture the various ways in which activities are represented and controlled in workflows. Implementing these patterns gives Oracle BPM the capability to handle the widest range of possible scenarios to model and execute processes.

This chapter will focus on the flow control patterns in the following points:

- Sequence flow pattern
- Exclusive choice and simple merge pattern
- Multichoice and synchronizing merge pattern
- Structured synchronizing merge pattern
  - Local synchronizing merge pattern
- Parallel split and synchronization pattern
- Conditional parallel split and parallel merge pattern
- Multimerge pattern
- Discriminator and partial join pattern
  - Structured discriminator pattern
  - Structured partial join pattern
- Complex synchronization pattern
  - Canceling discriminator pattern
  - Canceling partial join pattern

## **Sequence flow pattern**

One of the fundamental steps in the BPM process modeling is to build a process model (diagram) which enables a shared understanding between participants on a process flow pattern. The process participants are not going to discuss each and every page of the document, neither will a collaborative, iterative process improvement or approach succeed with a group of people sitting and walking through documents. However, this group will be interested in a process model (diagram) and discuss the flow, sequence, and process patterns visible through the process model. This makes sequence flow patterns of paramount importance, as each

and every activity is related to the other. In a process diagram, this relationship is created and managed through sequence flows. The following table summarizes the details of the sequence flow pattern:

<b>Signature</b>	Sequence Flow Pattern
<b>Classification</b>	Basic Flow Control Pattern
<b>Intent</b>	Offers sequence routing.
<b>Motivation</b>	The fundamental constituent to weave process components and demonstrate dependency and state transition between tasks/activities.
<b>Applicability</b>	The sequence pattern enforces a transitive temporal ordering to process activities. In business terms, sequences denote a strong dependency between activities and cater to strictly separating process involvement at organizational boundaries. They define the behavior of a business process.
<b>Implementation</b>	Widely adopted in most of the modeling languages including Oracle BPMN.
<b>Known issues</b>	Difference in acceptance.
<b>Known solution</b>	Usage of tokens in process instances.

The sequence is the simplest pattern and is implemented through a graphical sequence of actions, as graphical form is used for the sequencing of patterns. In BPMN, the model elements that are to be executed in sequence are connected with sequence flow connectors. When activities are connected with sequence flow connectors, processing of the second activity will not commence before the first activity is completed. This pattern defines the dependency of one task on the other and governs the fact that execution of one task is dependent on the other and cannot be completed until that task gets completed. Ordering of tasks in a business process is determined by sequence flow, and it governs how the process token will flow through the process. With sequence pattern, you can create a series of consecutive tasks, which are executed one after another based on the sequence connector's connections.

**Categories:** The sequence flow can be categorized as follows:

- **Incoming sequence flow:** This refers to flow that leads into a flow object
- **Outgoing sequence flow:** This refers to flow that leads out of a flow object

Some activities/flow objects can have both the sequence flows, and most of the activities/objects in a process have them. However, the `start` object can only contain an outgoing sequence flow and the `end` object can only contain an incoming sequence flow.



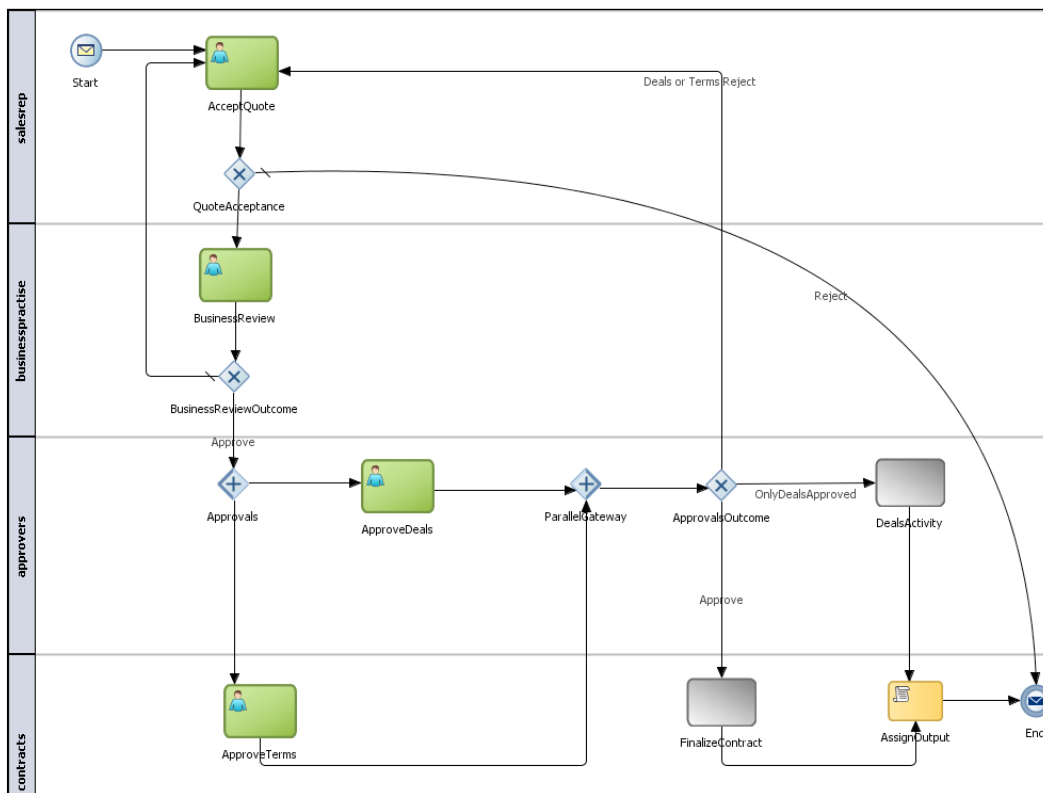
There are different types of sequence flows which are as follows:

- Default sequence flow / unconditional sequence flow
- Conditional sequence flow

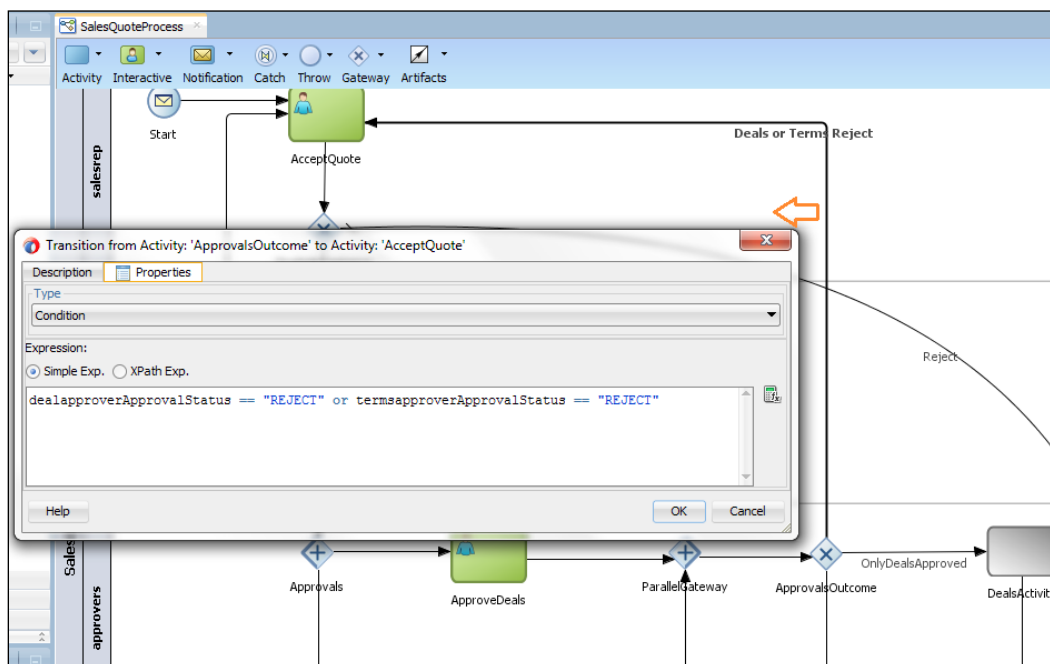
## Working with the sequence flow pattern

Perform the following steps to check the sequence flow usage in action:

1. Download the application (**SalesQuoteDemo**) contained in the download link of this chapter.
2. Open **SalesQuoteProject** in JDeveloper 12c.
3. Open **SalesQuoteProcess**; this will open the process flow in the design area.
4. Go to **Approvers Swim lane** and click on **Exclusive Gateway (ApprovalsOutcome)** that works on the **ApproveDeals** and **ApproveTerms** outcomes. The process is shown in the following screenshot:



5. Click on the outgoing sequence flow with the **Approve** tag. In the properties, you will find that the type of sequence flow is **Unconditional**. This is the default sequence flow from the **Exclusive Gateway**.
6. Now, click on the sequence flow with the **Deal or Terms Reject** tag and check its properties.
7. The sequence flow type is **Condition**, and it has a conditional expression build. When this conditional expression returns `true`, the process token will take this sequence flow path. This is shown in the following screenshot:



8. Click on the sequence flow with the **OnlyDealsApproved** tag and check its properties. This sequence flow is also a conditional flow with the following expression:

```
DealapproverAppr ovalStatus == "APPROVE" and
termsapproverApprovalStatus == "REJECT"
```



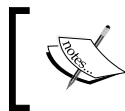
#### Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

## Elucidating the sequence flow pattern

The conditional sequence flow governs the token movement based on conditions associated with the sequence flows, where conditions are expressed using the x-path expressions. A path that is taken out of the gateways when none of the conditions specified on the conditional flow is evaluated. This is termed as default sequence flow, and it's drawn as an arrow line with a tick mark at one end.

Upon the arrival of token at the gateways, conditions associated with the drawn sequence flows are evaluated, and that sequence route is picked whose conditional evaluation returns `true`. Then, the token starts trailing this path. However, if none of the evaluations of the conditional flow returns `true`, then the default route is picked.



Conditional sequence flows can be associated with exclusive and inclusive gateways for split.

## Getting ready for executing use cases

This section talks about the steps that we will perform to get ready to execute the use cases demonstrated in this chapter. As we check **SalesQuoteProcess**, there are various human tasks. The following is the list of roles associated with the human task and users associated with the role:

Task	Role	User
Accept Quote	Salesrep	salesrep
Business Review	Business practice	fkafka
Approvers	Approvers	jcooper
Contracts	Contracts	jstein

We have to perform the following steps to execute the processes that have human task:

1. Log in to the WebLogic console and navigate to **myrealm** (embedded LDAP).
2. Click on the **User and Group** tab.
3. Verify that we have the aforementioned listed users in **myrealm**. If not, we can create users (salesrep, fkafka, jcooper, and jstein) in **myrealm**.

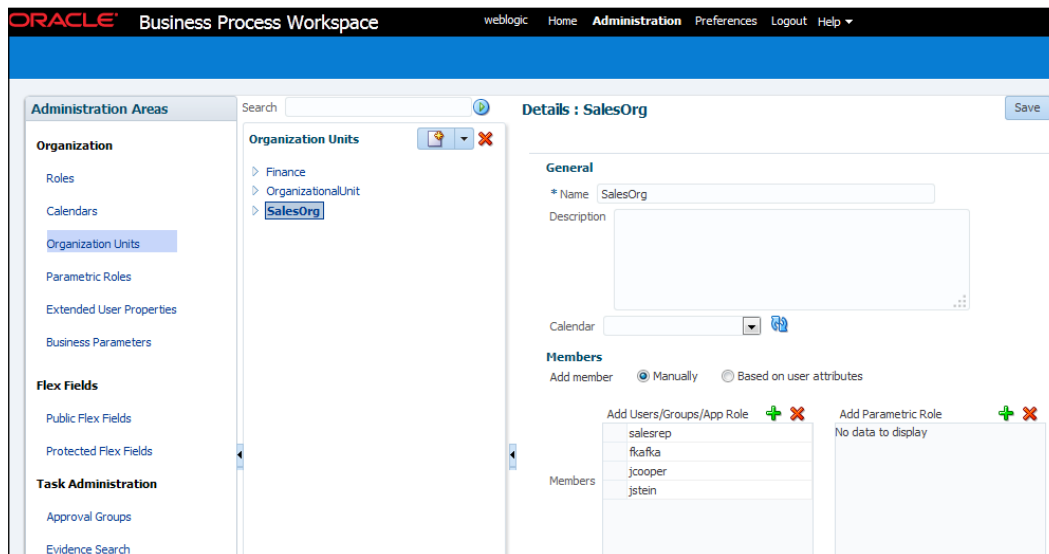


If we execute demo community that is installed while configuring Oracle BPM 12c, we will get users (fkafka, jcooper, and jstein). However, we can follow the preceding steps and create a user (salesrep).

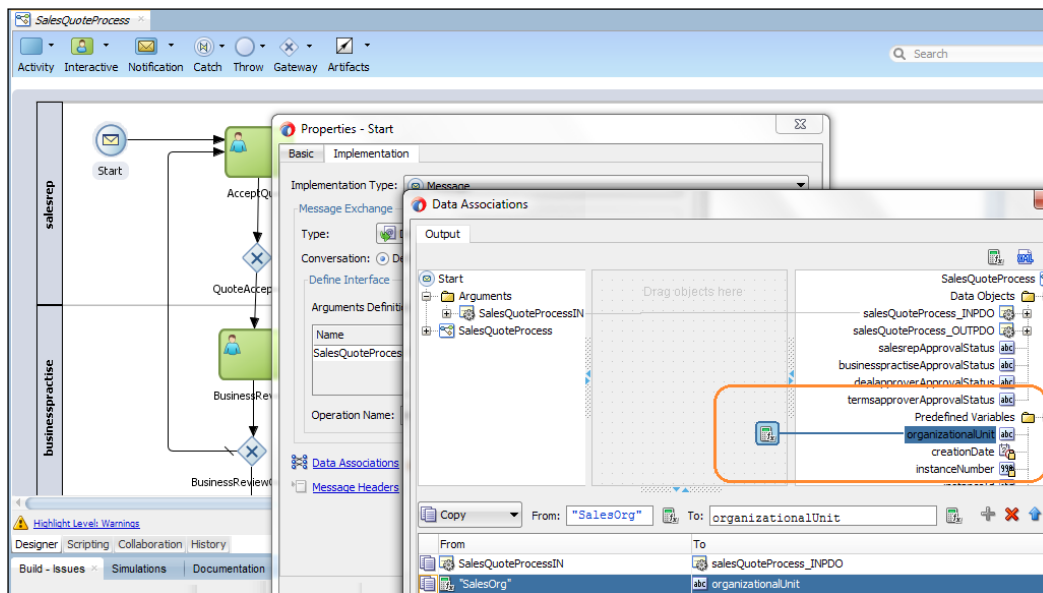
4. Open JDeveloper and navigate to **Organization** in **SalesQuoteProject**.
5. Click on **Roles** and associate users to roles as listed in the preceding table. Save the changes.

Human tasks are executed with respect to organization units. Hence, we will create an organization unit and associate the users to it. We will also make sure that the organization unit is passed to the process when the process executes. Execute the following steps:

1. Log in to the Oracle BPM workspace as an admin user (weblogic).
2. Navigate to **Administration** | **Organization** | **Organization Units**.
3. Click on the **+** icon to create a root organization.
4. Enter the name of the organization as `SalesOrg`.
5. In the **Members** section, add the users we listed in the preceding table. To add users, we can browse the **myrealm** LDAP.
6. When users are added, we can save the changes. This process is shown in the following screenshot:



7. Go back to JDeveloper and open **SalesQuoteProcess**.
8. Click on the Message Start Event (**Start**) and open its properties.
9. Go to the **Implementation** tab and open data association.
10. On the right-hand side of data association, scroll to the predefined variable (**Organization Unit**).
11. Assign the newly created organization units, **SalesOrg**, to the predefined variable (**Organization Units**) and save the project. This is demonstrated in the following screenshot:



## Exclusive choice and simple merge pattern

In this section, we will uncover the exclusive choice and simple merge pattern. It's also known as the exclusive choice pattern.

The control points in the process flow, where the sequence flows converge or diverge are known as gateways. There are different types of gateways, each supporting specific control logics. The gateway types are indicated with a marker in the center of the gateway symbol. Gateways can split and/or join (merge) sequence flows. You need gateways to control the process flow. A gateway is used to model decisions, merges, forks, and joins on a BPMN business process diagram. An exclusive gateway in Oracle BPMN offers simple split and merge patterns. An exclusive gateway

(represented by XOR) evaluates the state of the business process and based on the condition, breaks the flow into one of the two or more mutually exclusive paths. This is how the name "mutually exclusive" got derived. The exclusive gateway splits the process into multiple paths, but the token follows only one path. The following table illustrates the details of the exclusive choice pattern:

<b>Signature</b>	Exclusive Choice Pattern
<b>Classification</b>	Basic Flow Control Pattern
<b>Intent</b>	Breaks the flow into one of the two or more mutually exclusive paths.
<b>Motivation</b>	Fundamental constituent to enable dynamic routing decision.
<b>Applicability</b>	Decision point in the business process where the sequence flow will take only one of the possible outgoing paths when the process is performed.
<b>Implementation</b>	Widely adopted in most of the modeling languages, including Oracle BPMN, as the XOR gateway.
<b>Known issues</b>	Enforcing accuracy in triggering an outgoing path.
<b>Known solution</b>	Based on the evaluation of the conditions associated with the outgoing sequence flows from the gateway, routes are determinate. In case of multiple outgoing sequence flow, it is always a best practice to associate an order of their evaluation, as this will enable the fact that in case of multiple conditions getting evaluated as <code>true</code> , the process token will route to the first sequence flow for which the evaluation is <code>true</code> .

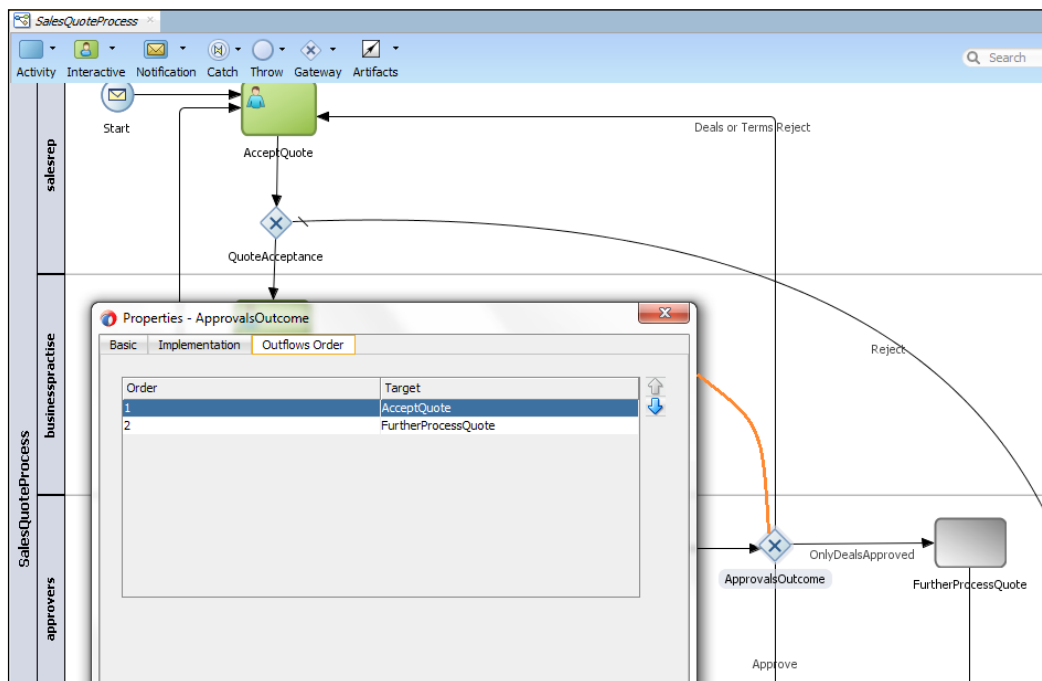
The decision mechanisms are categorized as follows:

- **Data:** An example of data is conditional expression. The conditional expressions are evaluated at the gateway when the process token reaches the gateway. That path whose evaluation result is `true` is followed, and it can route to only one flow
- **Events** (for example, the receipt of alternative messages): An event-based XOR gateway represents a divergence point where the alternatives paths are picked based on the event that occurs at that instance in the process flow. The event could be a receipt of message or a timer event. In an event-based gateway, it's the events that determine the path to be taken and not the conditional evaluations. The process becomes dynamic as process divergence is based on the external system's interaction with the process.

## Working with exclusive choice and simple merge pattern

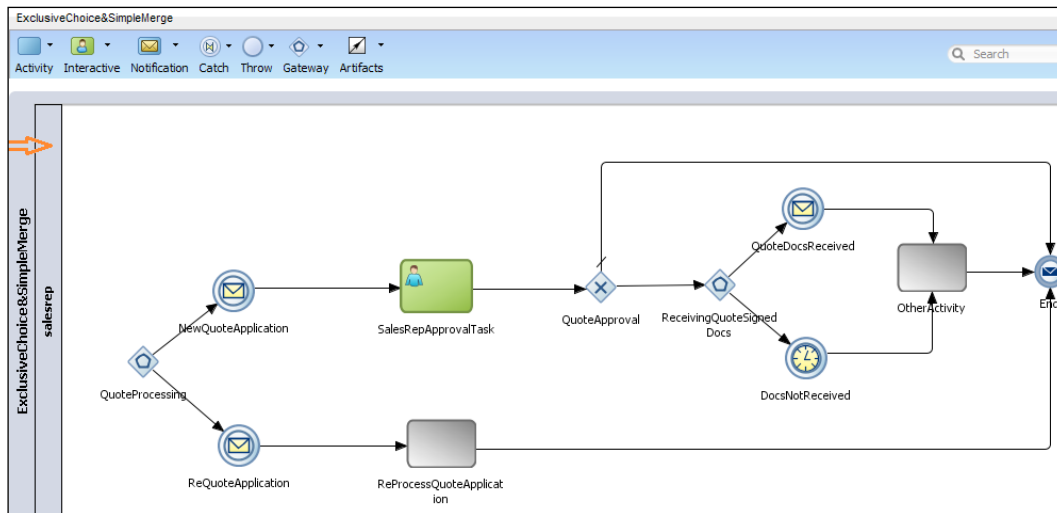
In order to evaluate the data-decision mechanism, refer to **SalesQuoteProcess** associated with the project (you have referred to it in the *Working with sequence flow pattern* section). Check the **Approvals Outcome** exclusive gateway, as shown in the following screenshot.

There are three outgoing sequence flows from the **Approvals Outcome** exclusive gateway. Two are conditional and one is default, as we discussed in *The Sequence flow pattern* section. Hence, these sequence flow conditions are based on the values of process data, the value of the data token itself, to determine which path should be taken. An order of evaluation is associated with the **Approvals Outcome** exclusive gateway, as this will enable the fact that in case of multiple conditions getting evaluated as `true`, the process token will route to the first sequence flow for which the evaluation is `true`. The following screenshot demonstrates this process:



Open the **ExclusiveChoice&SimpleMerge** process in JDeveloper 12c to evaluate the event-based gateway.

The use case illustrated in the preceding screenshot elucidates that quote processing can happen for both, **New Quote Application** and **Existing Quote Application**. In this case, use an event-based gateway, as there are multiple types of messages or events that can start a business process. The **SalesReqApprovalTask** human task is associated with the **salesrep** role, and we already assigned a user (salesrep) to this role. Hence, when the process executes the task, it will get assigned to the salesrep user, as shown in the following screenshot:

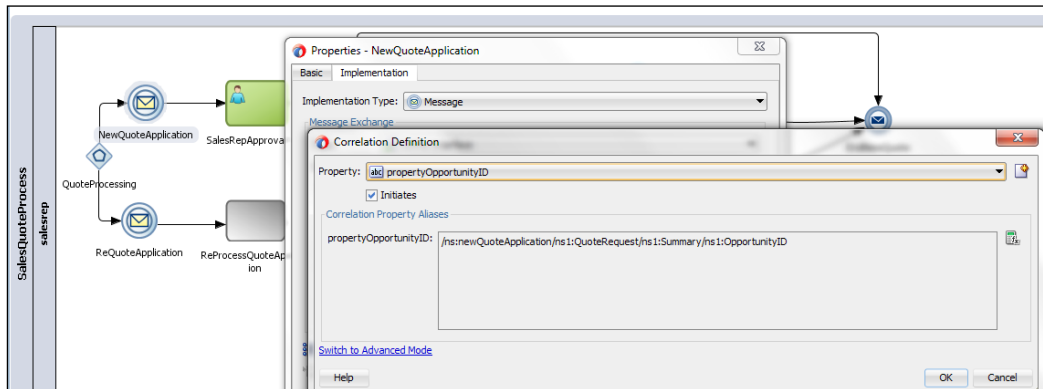


The following are the facts about the use case:

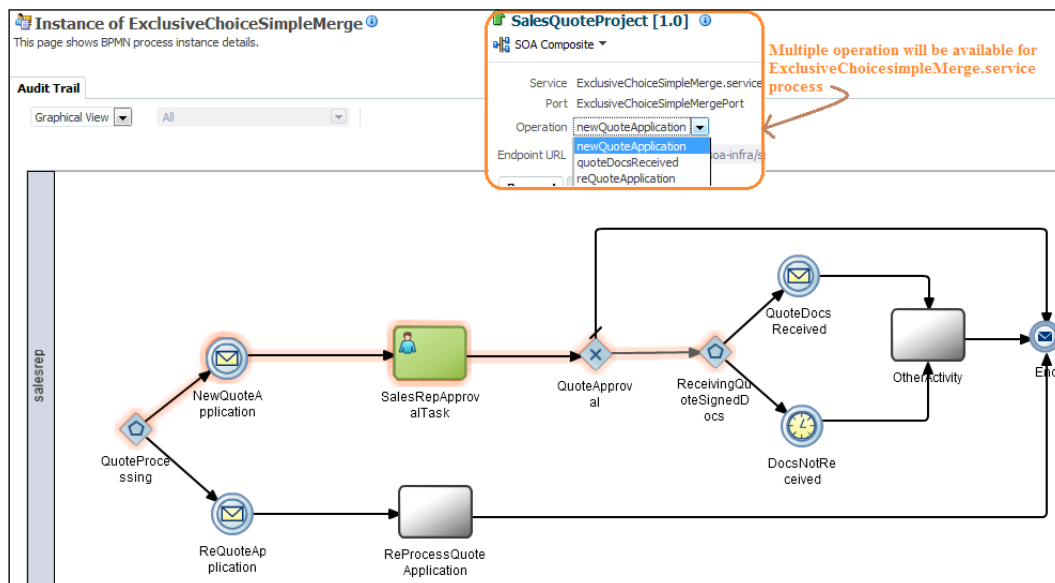
- **Quote Processing** is an initiating type of event-based gateway. **NewQuote Application** and **ReQuoteApplication** will catch the event messages. **SalesReqApprovalTask** is a task to be performed by the sales representative.
- **QuoteApproval** is the decision point based on process data which is the outcome of the user task (**SalesReqApprovalTask**) performed by the sales representative.
- **ReceivingQuoteSignedDocs** is a non-initiating event-based gateway.
- **QuoteDocsReceived** is a Message Catch Event, while the **DocsNotReceived** timer will move the token flow if documents are not received in 3 days.



- **OtherActivity** is a drafted process that performs further quote processing. The correlation key is designed and associated with all the event messages (**NewQuoteApplication**, **ReQuoteApplication**, and **QuoteDocsReceived**). This is demonstrated in the following screenshot:



When the process initiates, it would either initiate for a new quote or an existing quote. If initiated for a new quote, it would be caught by the **NewQuoteApplication** event message. If initiated for an existing quote, it would be caught by the **ReQuoteApplication** event message, as shown in the following screenshot:



Test the process for the **NewQuoteApplication** event message by performing the following steps:

1. Open EM Console and click on the **SalesQuoteProject** project.
2. Execute **ExclusiveChoiceSimpleMerge.service** to execute the **ExclusiveChoice&SimpleMerge** process.
3. Select the **NewQuoteApplication** operation. As we can see in the preceding screenshot, **ExclusiveChoiceSimpleMerge.service** exposes multiple operations, which are essentially the event gateway's Message Catch Events.
4. Browse through the **ExclusiveChoiceSimpleMerge.xml** test data file in the project by navigating to **SalesQuoteProject | SOA | testsuites**.
5. Execute the process instance.
6. Log in to the BPMN workspace as a salesrep user and **APPROVE** the **SalesReqApprovalTask** task.

The **Quote Processing** event gateway initiates the sequence that has the **NewQuoteApplication** message event, and the instance reaches the **SalesReqApprovalTask** user task. Once the task is approved, we will find that the process halts at the **ReceivingQuoteSignedDocs** event gateway. The instance status will be running, and the token will stay there until a token arrives from any of the branches. Either the supporting document message will be received, or the waiting time will exceed three days.

## Knowing about the exclusive choice pattern

Events receive communication, and hence, correlation needs to be defined to correlate them with the main process instance. A quote's opportunity ID is used as a correlation key. This correlation key is used in the intermediate events to correlate them with the existing process instance. With the correlation defined for the intermediate event gateway, the message will be correlated back to the original instance when it arrives at the **QuoteDocsReceived** event.

The message flow waits at the **ReceivingQuoteSignedDocs** event-based gateway, waiting for a token to arrive from any of its branches. In this case, the token can be a receipt of an event message or time. The first event triggers one of the alternatives that is an exclusion of any other path from the gateway. The event will basically pull the token from the gateway and continue to sequence flow that event.

## Elucidating the simple merge pattern

We can use exclusive gateway to merge incoming sequence flows; however, there is no synchronization with other tokens that might be coming from other paths within the process flow. Simple merge combines several transitions back into a single activity. Tokens that merge at an exclusive gateway will be passed through as they are, and they would not be evaluated. Token merging at the exclusive gateway will not be synchronized. At the converging point, you would never have more than one token.

The following table illustrates the details of a simple merge pattern:

<b>Signature</b>	Simple Merge Pattern
<b>Classification</b>	Basic Flow Control Pattern
<b>Intent</b>	Merging two or more paths.
<b>Motivation</b>	Fundamental constituent to enable simple merge.
<b>Applicability</b>	Combining several transitions back into a single activity. At converging point, you would never have more than one token.
<b>Implementation</b>	Widely adopted in most of the modeling languages using XOR-Join.
<b>Known issues</b>	Token merging at the exclusive gateway will not be synchronized.
<b>Known solution</b>	Multimerge.

For example, we have an invoice payment, and there are different ways to pay the invoice, which include paying through credit card, bank transfer, or check. However, to make the payment, only one method will be used for an invoice, and once paid, the data need to be infused into Oracle E-Business Suite ERP. We would always use only one payment method. This is an ideal candidate for a simple merge using an exclusive gateway.

## Multichoice and synchronizing merge pattern

We can perform simple split and merge with the gateway (inclusive gateway) offered by Oracle BPMS. It can perform token evaluation and also synchronize the token merging at the convergence. An inclusive gateway (OR) specifies that one or more of the available paths will be taken. They could all be taken, or only one of them will be taken. This capability is also termed **Multichoice**. Sometimes, you need to select a subset of alternatives from a set of possible alternatives. This is what the multiple choice (inclusive) patterns are for. The multiple choice pattern is a point in the workflow where, based on a decision or control data, one or more branches are chosen, triggering one or more paths of the process.

An inclusive OR merge is simply an OR gateway that is used to merge multiple sequence flows into one outgoing sequence flow. Each outgoing sequence flow from the gateway will have a Boolean expression that will be evaluated to determine which sequence flow should be used to continue the process. The downstream inclusive gateway is used to merge the paths created by the upstream inclusive gateway. The downstream inclusive gateway synchronizes all the alternative paths created by the multiple choice gateway. The following table shows details of the multichoice pattern:

<b>Signature</b>	Multichoice Pattern
<b>Classification</b>	Advance Flow Control Pattern
<b>Intent</b>	Breaks the flow into one of the two or more mutually exclusive paths.
<b>Motivation</b>	Fundamental constituent to enable selection of a subset of alternative paths from a set of possible alternatives.
<b>Applicability</b>	Decision point in the business process where the sequence flow will take one or more of the possible outgoing paths.
<b>Implementation</b>	Widely adopted in most of the modeling languages using the OR split.
<b>Known issues</b>	Ensure at least one path selection.
<b>Known solution</b>	Inclusive gateway splits the process at the divergence; however, process tokens can advance to multiple outgoing flows/paths. Sequence flow is picked based on the conditional evaluation where a token is generated for each flow for which the condition is evaluated as <code>true</code> , otherwise, a default sequence flow is picked. The solution is the default path.

## Demonstrating multichoice and synchronization with the OR gateway

Download **SalesQuoteProject** from the download link of this chapter. Open the project in JDeveloper. Open the **SalesQuoteSimpleMerge** process. The process accepts **QuoteRequestData** and waits for the sales representative's approval, which will be performed by the `salesrep` user (we already created a `salesrep` user in WebLogic **myrealm** in the previous section). Deploy the process to a WebLogic server.

Let's consider an example scenario. In this business process (**SalesQuoteProcess**), after **SalesQuoteApprovalTask**, the approval request also needs to be sent to **Legal** and **Terms** for approval. Once **Legal** and **Terms** approve, other activities are performed over **Quote**.

When **Legal** and **Terms** act on the task, the gateway will merge them, and the process will move ahead. Perform the following steps to test the **SalesQuoteSimpleMerge** process:

1. Test the process from EM or use SOAPUI.
2. Enter the **QuoteRequest** elements and submit **QuoteRequest**. We can use the test data (**SalesQuoteSimpleMerge.xml**) available in the `testsuites` folder in the project.
3. We will notice that the process token is waiting at **SalesQuoteApprovalTask** to be acted upon by the `salesrep` user.
4. Log in to the BPM workspace at `http://<server>:<port>/bpm/workspace` as a `salesrep` user and approve the **QuoteRequest**.

We will find that the process token will reach both the user tasks, **Legal** and **Terms**, for approval. There will be two threads created to process the **LegalApproval** and **TermsApproval** tasks and both will be in the processing mode.

As per the process design, both these tasks will again be assigned to the `salesrep` user. You can customize the sample and associate different users for **Terms** and **Legal** approval. For the moment, log in to the BPM workspace again as the `salesrep` user and approve the legal task. You will find that in the process, the thread processing the **LegalApproval** task is completed, while the thread processing the **TermsApproval** task is still processing.

As we can check in the following screenshot, the process flow shows the point where the process token is awaiting. The audit trail on the left-hand side showcases the snapshot when the **Legal** task is approved; however, the **Terms** task is not being acted upon by the `salesrep` user. We will notice that for both the tasks (**Legal** and **Terms**), there are two separate threads for processing. Even though the **Legal** task is approved, the process token waits at the merge inclusive gateway (**MergeQuoteApproval**). Log in back to the BPM workspace as the `salesrep` user and approve the **Terms** tasks. In the right-hand side of preceding screenshot, we can witness that once both tasks are acted upon by the user, the process token converges at the inclusive gateway (**MergeQuoteApproval**), and the process moves ahead to subsequent activities. This is shown in the following screenshot:

**Instance of SalesQuoteSimpleMerge**  
This page shows BPMN process instance details.

**Audit Trail**

Tree View | All

**SalesQuoteSimpleMerge** Thread 0

- Start Thread 0 Activity completed
- SalesRepApprovalTask Thread 0 Activity completed Task Number 200409
- SplitQuoteApproval Thread 0 Activity completed
- Threads Thread 0 Thread Grouped
- LegalApprovalTask Thread 1 Thread completed Task Number 200413
- LegalApprovalTask Thread 1 Activity completed Task Number 200413
- LegalApprovalTask Thread 1 Instance entered the Activity Responsible SalesQuoteProj Task Number 200413
- LegalApprovalTask Thread 1 Instance left the activity Responsible salesrep,user Task Number 200413
- TermsApprovalTask Thread 2 Thread processing Task Number 200411
- TermsApprovalTask Thread 2 Activity processing Task Number 200411
- TermsApprovalTask Thread 2 Instance entered the Activity Responsible SalesQuoteProj Task Number 200411

**Process Flow**

**Task history**

1 Stage1

1.1 - default.DefaultPerformer Task Completed - Approved salesrep

**Task history**

1 Stage1

1.1 SalesQuoteProject.salesrep Assigned

*Audit Trail After - Legal Task Approval, however no action performed on Terms Task..*

Tree View | All

**SalesQuoteSimpleMerge** Thread 0 Instance created

- Start Thread 0 Activity completed
- SalesRepApprovalTask Thread 0 Activity completed Task Number 200409
- SplitQuoteApproval Thread 0 Activity completed
- Threads Thread 0 Thread Grouped
- LegalApprovalTask Thread 1 Thread completed Task Number 200413
- LegalApprovalTask Thread 1 Activity completed Task Number 200413
- LegalApprovalTask Thread 1 Instance entered the Activity Responsible SalesQuoteProj Task Number 200413
- LegalApprovalTask Thread 1 Instance left the Activity Responsible salesrep Task Number 200413
- TermsApprovalTask Thread 2 Thread completed Task Number 200411
- TermsApprovalTask Thread 2 Task history
- TermsApprovalTask Thread 2 1 Stage1
- TermsApprovalTask Thread 2 1.1 Task Completed salesrep
- TermsApprovalTask Thread 2 Instance left the Activity Responsible salesrep Task Number 200411
- MergeQuoteApproval Thread 0 Activity completed
- End Thread 0 Activity completed

*Audit Trail After - Legal Task and Terms Tasks are Approval..*