

# PROGRESS<sup>®</sup> ARTIX<sup>™</sup>

## Getting Started with Artix

Version 5.6, December 2011

**© 2011 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.**

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Actional, Apama, Artix, Business Empowerment, DataDirect (and design), DataDirect Connect, DataDirect Connect64, DataDirect Technologies, DataDirect XML Converters, DataDirect XQuery, DataXtend, Dynamic Routing Architecture, EdgeXtend, Empowerment Center, Fathom, Fuse Mediation Router, Fuse Message Broker, Fuse Services Framework, IntelliStream, IONA, Making Software Work Together, Mindreef, ObjectStore, OpenEdge, Orbix, PeerDirect, POSSENET, Powered by Progress, PowerTier, Progress, Progress DataXtend, Progress Dynamics, Progress Business Empowerment, Progress Empowerment Center, Progress Empowerment Program, Progress OpenEdge, Progress Profiles, Progress Results, Progress Software Developers Network, Progress Sonic, ProVision, PS Select, Savvion, SequeLink, Shadow, SOAPscope, SOAPStation, Sonic, Sonic ESB, SonicMQ, Sonic Orchestration Server, SpeedScript, Stylus Studio, Technical Empowerment, Web-Speed, Xcalia (and design), and Your Software, Our Technology—Experience the Connection are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. AccelEvent, Apama Dashboard Studio, Apama Event Manager, Apama Event Modeler, Apama Event Store, Apama Risk Firewall, AppsAlive, AppServer, ASPen, ASP-in-a-Box, BusinessEdge, Business Making Progress, Cache-Forward, CloudEdge, DataDirect Spy, DataDirect SupportLink, Fuse, FuseSource, Future Proof, GVAC, High Performance Integration, ObjectStore Inspector, ObjectStore Performance Expert, OpenAccess, Orbacus, Pantero, POSSE, ProDataSet, Progress Arcade, Progress CloudEdge, Progress Control Tower, Progress ESP Event Manager, Progress ESP Event Modeler, Progress Event Engine, Progress RFID, Progress RPM, Progress Software Business Making Progress, PSE Pro, SectorAlliance, SeeThinkAct, Shadow z/Services, Shadow z/Direct, Shadow z/Events, Shadow z/Presentation, Shadow Studio, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Sonic Business Integration Suite, Sonic Process Manager, Sonic Collaboration Server, Sonic Continuous Availability Architecture, Sonic Database Service, Sonic Workbench, Sonic XML Server, The Brains Behind BAM, WebClient, and Who Makes Progress are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

### **Third Party Acknowledgments:**

Progress Artix ESB for C++ v5.6 incorporates Xalan v2.3.1 technologies from the Apache Software Foundation (<http://www.apache.org>). Such Apache technologies are subject to the following terms and conditions: The Apache Software License, Version 1.1. Copyright (C) 1999-2002 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org>). Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear. 4. The names "Ant", "Xerces," "Xalan," "Log 4J," and "Apache Software Foundation" must not be used to: endorse or promote products derived from this software without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org). 5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation. THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see <http://www.apache.org/>. Xalan was originally based on software copyright (c) 1999, Lotus Development Corporation., <http://www.lotus.com>. Xerces was originally based on software copyright (c) 1999, International Business Machines, Inc., <http://www.ibm.com>.

Progress Artix ESB for C++ v5.6 incorporates Xerces C++ v2.4 technology from the Apache Software Foundation (<http://www.apache.org>). Such Apache technology is subject to the following terms and conditions: The Apache Software License, Version 1.1 - Copyright (c) 1999-2001 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "Xerces" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org).

5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation. THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Artix ESB for C++ v5.6 incorporates Apache Xerces v2.5.0 technology from the Apache Software Foundation (<http://www.apache.org/>). Such Apache technology is subject to the following terms and conditions: The Apache Software License, Version 1.1 - Copyright (c) 1999-2002 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "Xerces" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org).

5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation and was originally based on software copyright (c) 1999, International Business Machines, Inc., <http://www.ibm.com>. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

Progress Artix ESB for C++ v5.6 incorporates Xerces C++ v1.7 technology from the Apache Software Foundation (<http://www.apache.org>). Such Apache technology is subject to the following terms and conditions: The Apache Software License, Version 1.1. - Copyright (c) 1999-2004 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.
4. The names "Xalan" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org).
5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING

ING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation and was originally based on software copyright (c) 1999, Lotus Development Corporation., <http://www.lotus.com>. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

Progress Artix ESB for C++ v5.6 incorporates Apache Velocity v1.3 technology from the Apache Software Foundation (<http://www.apache.org>). Such Apache technology is subject to the following terms and conditions: The Apache Software License, Version 1.1 - Copyright (c) 2000-2003 The Apache Software Foundation. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgement: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)." Alternately, this acknowledgement may appear in the software itself, if and wherever such third-party acknowledgements normally appear.

4. The names "The Jakarta Project", "Velocity", and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org).

5. Products derived from this software may not be called "Apache", "Velocity" nor may "Apache" appear in their names without prior written permission of the Apache Group.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Artix ESB for C++ v5.6 incorporates Log4J v1.2.6 technology from the Apache Software Foundation (<http://www.apache.org>). Such Apache technology is subject to the following terms and conditions: The Apache Software License, Version 1.1 - Copyright (C) 1999 The Apache Software Foundation. All rights reserved. Redistribution and use in

source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).". Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "log4j" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [apache@apache.org](mailto:apache@apache.org).

5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

(a) Progress Artix ESB for C++ v5.6 incorporates JDOM Beta 9 technology from JDOM. Such technology is subject to the following terms and conditions: Copyright (C) 2000-2004 Jason Hunter & Brett McLaughlin. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the disclaimer that follows these conditions in the documentation and/or other materials provided with the distribution. 3. The name "JDOM" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [request\\_AT\\_jdom\\_DOT\\_org](mailto:request_AT_jdom_DOT_org). 4. Products derived from this software may not be called "JDOM", nor may "JDOM" appear in their name, without prior written permission from the JDOM Project Management [request\\_AT\\_jdom\\_DOT\\_org](mailto:request_AT_jdom_DOT_org). In addition, we request (but do not require) that you include in the end-user documentation provided with the redistribution and/or in the software itself an acknowledgement equivalent to the following: "This

product includes software developed by the JDOM Project (<http://www.jdom.org/>).<sup>1</sup> Alternatively, the acknowledgment may be graphical using the logos available at <http://www.jdom.org/images/logos>. THIS SOFTWARE IS PROVIDED AS IS AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. This software consists of voluntary contributions made by many individuals on behalf of the JDOM Project and was originally created by Jason Hunter <jhunter\_AT\_jdom\_DOT\_org> and Brett McLaughlin <brett\_AT\_jdom\_DOT\_org>. For more information on the JDOM Project, please see <<http://www.jdom.org/>>

Progress Artix ESB for C++ v5.6 incorporates IBM-ICU v2.6 and IBM-ICU v2.6.1 technologies from IBM. Such technologies are subject to the following terms and conditions: Copyright (c) 1995-2003 International Business Machines Corporation and others All rights reserved. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder. All trademarks and registered trademarks mentioned herein are the property of their respective owners.

Progress Artix ESB for C++ v5.6 incorporates John Wilson MinML v1.7 technology from John Wilson. Such technology is subject to the following terms and conditions: Copyright (c) 1999, John Wilson ([tug@wilson.co.uk](mailto:tug@wilson.co.uk)). All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright



notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by John Wilson. The name of John Wilson may not be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY JOHN WILSON ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JOHN WILSON BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Artix ESB for C++ v5.6 incorporates SourceForge - NET-SNMP v5.0.7 technology from SourceForge and Networks Associates Technology, Inc. Such technology is subject to the following terms and conditions: Various copyrights apply to this package, listed in various separate parts below. Please make sure that you read all the parts. Up until 2001, the project was based at UC Davis, and the first part covers all code written during this time. From 2001 onwards, the project has been based at SourceForge, and Networks Associates Technology, Inc hold the copyright on behalf of the wider Net-SNMP community, covering all derivative work done since then. An additional copyright section has been added as Part 3 below also under a BSD license for the work contributed by Cambridge Broadband Ltd. to the project since 2001. An additional copyright section has been added as Part 4 below also under a BSD license for the work contributed by Sun Microsystems, Inc. to the project since 2003. Code has been contributed to this project by many people over the years it has been in development, and a full list of contributors can be found in the README file under the THANKS section. ---- Part 1: CMU/UCD copyright notice: (BSD like) ---- Copyright 1989, 1991, 1992 by Carnegie Mellon University. Derivative Work - 1996, 1998-2000. Copyright 1996, 1998-2000 The Regents of the University of California. All Rights Reserved. Permission to use, copy, modify and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of CMU and The Regents of the University of California not be used in advertising or publicity pertaining to distribution of the software without specific written permission. CMU AND THE REGENTS OF THE UNIVERSITY OF CALIFORNIA DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL CMU OR THE REGENTS OF THE UNIVERSITY OF CALIFORNIA BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM THE LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR

IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. ----  
Part 2: Networks Associates Technology, Inc copyright notice (BSD) ----- Copyright (c)  
2001-2003, Networks Associates Technology, Inc. All rights reserved. Redistribution and  
use in source and binary forms, with or without modification, are permitted provided that  
the following conditions are met: \*Redistributions of source code must retain the above  
copyright notice, this list of conditions and the following disclaimer.\* Redistributions in  
binary form must reproduce the above copyright notice, this list of conditions and the fol-  
lowing disclaimer in the documentation and/or other materials provided with the distribu-  
tion.\* Neither the name of the Networks Associates Technology, Inc nor the names of its  
contributors may be used to endorse or promote products derived from this software without  
specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPY-  
RIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR  
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED  
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PUR-  
POSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR  
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPE-  
CIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT  
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF  
USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED  
AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIA-  
BILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY  
WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSI-  
BILITY OF SUCH DAMAGE. ---- Part 3: Cambridge Broadband Ltd. copyright notice  
(BSD) ----- Portions of this code are copyright (c) 2001-2003, Cambridge Broadband Ltd.  
All rights reserved. Redistribution and use in source and binary forms, with or without mod-  
ification, are permitted provided that the following conditions are met:\*Redistributions of  
source code must retain the above copyright notice, this list of conditions and the following  
disclaimer.\* Redistributions in binary form must reproduce the above copyright notice, this  
list of conditions and the following disclaimer in the documentation and/or other materials  
provided with the distribution.\* The name of Cambridge Broadband Ltd. may not be used to  
endorse or promote products derived from this software without specific prior written per-  
mission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER ``AS IS"  
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED  
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A  
PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPY-  
RIGHT HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPE-  
CIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT  
LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF  
USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED  
AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIA-  
BILITY, OR TORT (INCLUDING NEGLIGENCE

OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,  
EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. ---- Part 4: Sun  
Microsystems, Inc. copyright notice (BSD) -----Copyright © 2003 Sun Microsystems, Inc.,  
4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved. Use is  
subject to license terms below. This distribution may include materials developed by third  
parties. Sun, Sun Microsystems, the Sun logo and Solaris are trademarks or registered trade-  
marks of Sun Microsystems, Inc. in the U.S. and other countries. Redistribution and use in  
source and binary forms, with or without modification, are permitted provided that the fol-

lowing conditions are met:\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.\* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.\* Neither the name of the Sun Microsystems, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. ---- Part 5: Sparta, Inc copyright notice (BSD) -----Copyright (c) 2003-2005, Sparta, Inc. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.\* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.\* Neither the name of Sparta, Inc nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. ---- Part 6: Cisco/BUPTNIC copyright notice (BSD) ----- Copyright (c) 2004, Cisco, Inc and Information Network Center of Beijing University of Posts and Telecommunications. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. \* Neither the name of Cisco, Inc, Beijing University of Posts and Telecommunications, nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS

PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. ---- Part 7: Fabasoft R&D Software GmbH & Co KG copyright notice (BSD) ----- Copyright (c) Fabasoft R&D Software GmbH & Co KG, 2003 oss@fabasoft.com Author: Bernhard Penz. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.\* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. \* The name of Fabasoft R&D Software GmbH & Co KG or any of its subsidiaries, brand or product names may not be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Artix ESB for C++ v5.6 incorporates OpenSSL/SSLey v0.9.8i technology from OpenSSL.org. Such Technology is subject to the following terms and conditions: LICENSE ISSUES =====

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLey license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License -----

/\*

=====  
=====

Copyright (c) 1998-2008 The OpenSSL Project. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (<http://www.openssl.org/>)"
4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact [openssl-core@openssl.org](mailto:openssl-core@openssl.org).
5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.openssl.org/>)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

=====  
=====

This product includes cryptographic software written by Eric Young ([ey@cryptsoft.com](mailto:ey@cryptsoft.com)).  
This product includes software written by Tim Hudson ([tjh@cryptsoft.com](mailto:tjh@cryptsoft.com)).

Original SSLey License -----

Copyright (C) 1995-1998 Eric Young ([ey@cryptsoft.com](mailto:ey@cryptsoft.com)) All rights reserved.

This package is an SSL implementation written by Eric Young ([ey@cryptsoft.com](mailto:ey@cryptsoft.com)). The implementation was written so as to conform with Netscapes SSL. This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson ([tjh@cryptsoft.com](mailto:tjh@cryptsoft.com)). Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be

given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related :-).
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

Progress Artix ESB for C++ v5.6 incorporates Bouncycastle v1.3.3 cryptographic technology from the Legion Of The Bouncy Castle (<http://www.bouncycastle.org>). Such Bouncycastle 1.3.3 cryptographic technology is subject to the following terms and conditions: Copyright (c) 2000 - 2006 The Legion Of The Bouncy Castle (<http://www.bouncycastle.org>). Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING

FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Progress Artix ESB for C++ v5.6 incorporates PCRE 7.8 from PCRE for the purpose of providing a set of functions that implement regular expression pattern matching using the same syntax and semantics as Perl 5. Such technology is subject to the following terms and conditions: PCRE LICENCE. PCRE is a library of functions to support regular expressions whose syntax and semantics are as close as possible to those of the Perl 5 language. Release 7 of PCRE is distributed under the terms of the "BSD" licence, as specified below. The documentation for PCRE, supplied in the "doc" directory, is distributed under the same terms as the software itself. The basic library functions are written in C and are freestanding. Also included in the distribution is a set of C++ wrapper functions. THE BASIC LIBRARY FUNCTIONS. Written by: Philip Hazel. Email local part: ph10. Email domain: cam.ac.uk. University of Cambridge Computing Service, Cambridge, England. Copyright (c) 1997-2008 University of Cambridge All rights reserved. THE C++ WRAPPER FUNCTIONS. Contributed by: Google Inc. Copyright (c) 2007-2008, Google Inc. All rights reserved. THE "BSD" LICENCE. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. \* Neither the name of the University of Cambridge nor the name of Google Inc. nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Artix ESB for C++ v5.6 incorporates mcpp v2.6.4 from Kiyoshi Matsui. Such technology is subject to the following terms and conditions: Copyright (c) 1998, 2002-2007 Kiyoshi Matsui kmatsui@t3.rim.or.jp All rights reserved. This software including the files in this directory is provided under the following license. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Progress Artix ESB for C++ v5.6 contains IBM Licensed Materials Copyright IBM Corporation 2010 (IBM 32-bit Runtime Environment for AIX, Java Technology Edition v 1.6.0 SR9 FP2).

Updated: December 5, 2011



# Contents

	17
<b>List of Figures</b>	<b>19</b>
<b>List of Tables</b>	<b>21</b>
<b>Preface</b>	<b>23</b>
<b>What is Covered in This Book</b>	<b>23</b>
<b>Who Should Read This Book</b>	<b>23</b>
<b>Organization of This Book</b>	<b>23</b>
<b>The Artix Documentation Library</b>	<b>23</b>
<b>Chapter 1 About Artix ESB</b>	<b>25</b>
<b>What is Artix ESB?</b>	<b>26</b>
<b>Runtime Comparison</b>	<b>29</b>
<b>Key Concepts in Depth</b>	<b>32</b>
Artix ESB Runtime Components	33
Artix Bus	34
Artix Endpoints	35
Artix Contracts	36
Artix Services	38
<b>Solving Problems with Artix ESB</b>	<b>40</b>
<b>Chapter 2 Understanding WSDL</b>	<b>43</b>
<b>WSDL Basics</b>	<b>44</b>
<b>Abstract Data Type Definitions</b>	<b>46</b>
<b>Abstract Message Definitions</b>	<b>49</b>
<b>Abstract Interface Definitions</b>	<b>52</b>
<b>Mapping to the Concrete Details</b>	<b>56</b>
<b>Index</b>	<b>57</b>

## CONTENTS

# List of Figures

Figure 1: Artix ESB Runtime Components

33

## LIST OF FIGURES

# List of Tables

Table 1: Artix ESB Feature Comparison	29
Table 2: Part Data Type Attributes	51
Table 3: Operation Message Elements	53
Table 4: Attributes of the Input and Output Elements	53

## LIST OF TABLES

# Preface

## **What is Covered in This Book**

*Getting Started with Artix* introduces IONA's Artix ESB technology, the Artix Designer GUI tool, and Web Services Description Language (WSDL).

## **Who Should Read This Book**

*Getting Started with Artix* is for anyone who needs to understand the concepts and terms used in IONA's Artix product.

## **Organization of This Book**

This book contains conceptual information about Artix and WSDL:

- [Chapter 1, "About Artix ESB,"](#) introduces the Artix ESB product, discussing key concepts in depth and describing the types of problems it is designed to solve.
- [Chapter 2, "Understanding WSDL,"](#) explains the basics of WSDL.

## **The Artix Documentation Library**

For information on the organization of the Artix library, the document conventions used, and finding additional resources, see [Using the Artix Library](#).

## PREFACE



# About Artix ESB

*This chapter introduces the main features of Artix ESB.*

---

**In this chapter**

This chapter discusses the following topics:

<a href="#">What is Artix ESB?</a>	<a href="#">page 26</a>
<a href="#">Runtime Comparison</a>	<a href="#">page 29</a>
<a href="#">Key Concepts in Depth</a>	<a href="#">page 32</a>
<a href="#">Solving Problems with Artix ESB</a>	<a href="#">page 40</a>

---

# What is Artix ESB?

---

## Overview

Artix ESB is an extensible enterprise service bus. It provides the tools for rapid application integration that exploits the middleware technologies and products already present within your organization.

The approach taken by Artix ESB relies on existing Web service standards and extends these standards to provide rapid integration solutions that increase operational efficiencies, capitalize on existing infrastructure, and enable the adoption or extension of a service-oriented architecture (SOA).

---

## Web services and SOAs

The information services community generally regards Web services as application-to-application interactions that use SOAP over HTTP.

Web services have the following advantages:

- The data encoding scheme and transport semantics are based on standardized specifications.
- The XML message content is human readable.
- The contract defining the service is XML-based and can be edited by any text editor.
- They promote loosely coupled architectures.

SOAs take the Web services concept and extend it to the entire enterprise. Using a SOA, your infrastructure becomes a collection of loosely coupled services.

Each service becomes an endpoint defined by a contract written in Web Services Description Language (WSDL). Clients, or service consumers, can then access the services by reading a service's contract.

---

## Artix and services

Artix extends the Web service standards to include more than just SOAP over HTTP. Thus, Artix allows organizations to define their existing applications as services without worrying about the underlying middleware. It also provides the ability to expose those applications across a number of middleware technologies without writing any new code.

Artix also provides developers with the tools to write new applications in C++ or Java that can be exposed as middleware-neutral services. These tools aid in the definition of the new service in WSDL and in the generation of stub and skeleton code.

Just like the WSDL contracts used to define a service, the code that Artix generates adheres to industry standards.

---

### Benefits of Artix

Artix ESB's extensible nature provides a number of benefits over other ESBs and older enterprise application integration (EAI) products. Chief among these is its speed and flexibility. In addition, Artix ESB provides enterprise levels of service such as session management, service discovery, security, and cross-middleware transaction propagation.

EAI products typically use a proprietary, canonical message format in a centralized EAI hub. When the hub receives a message, it transforms the message to this canonical format and then transforms the message to the format of the target application before sending it to its destination. Each application requires two adapters that are typically proprietary and that translate to and from the canonical format.

By contrast, Artix ESB does not require a hub architecture, nor does it use any intermediate message format. When a message is received by the bus, it is transformed directly into the target application's message format.

Artix ESB is highly configurable and easily extendable. You can configure it to load only the pieces you need for the functionality you require. If Artix ESB does not provide a transport or message format you need, you can easily develop your own service, extend the contract definitions, and configure Artix to load it.

---

### Artix ESB features

Artix ESB includes the following features:

- Support for multiple transports and message data formats
- C++ and Java development
- Message routing
- Cross-middleware transaction support
- Asynchronous Web services
- Deployment of services as plug-ins via a number of different containers
- Role-based security, single sign-on, and security integration
- Session management and stateful Web services
- Look-up services
- Load-balancing
- High-availability service clustering
- Integration with EJBs
- Easy-to-use development tools

- Integration with enterprise management tools such as IBM Tivoli and BMC Patrol
  - Support for XSLT-based message transformation
  - No need to hard-code WSDL references into applications
- 

### **Runtimes and programming models**

Artix ESB ships with two separate runtimes:

- The Artix ESB C++ Runtime supports development using the Artix C++ API.
  - The Artix ESB for Java Runtime is based on the Apache CXF, which provides a JAX-WS API and a JavaScript API.
- 

### **Using Artix ESB**

There are two ways to use Artix ESB in your enterprise:

- You can use Artix ESB to develop new applications using one of the supported APIs. In this situation, developers generate stub and skeleton code from WSDL, and Artix becomes a part of your development environment.
  - You can use the Artix bus to integrate two existing applications, built on different middleware technologies, into a single application. In this situation, developers simply create an Artix contract defining the integration of the systems. In most cases, no new code is needed.
- 

### **Becoming proficient with Artix ESB**

To become an effective Artix ESB developer you need an understanding of the following:

1. The difference between the two runtimes and the programming models available in Artix ESB.
2. The syntax for WSDL and the Artix ESB extensions to the WSDL specification.
3. The configuration mechanisms available in the two Artix runtimes.
4. The Artix APIs that you can use in your application.

This book introduces these concepts. The other books in the Artix documentation library covers the same technologies in greater detail.

---

# Runtime Comparison

---

## C++ Runtime

Artix ESB C++ Runtime provides developers with a C++ API to implement services. It is built on top of Progress Software's patented Adaptive Runtime Technology (ART).

Artix ESB C++ Runtime has a C++ core that provides a fast and stable platform for building applications.

---

## Java Runtime

Artix ESB Java Runtime provides the developer with both a JAX-WS 2.0 API and a JavaScript API with which to implement services.

It is based on the Apache CXF services framework and provides a fast, modular, and extensible platform for implementing services that is built purely in Java.

---

## Feature comparison

The table below gives a comparison of the bindings, transports, and quality of service features supported by each runtime.

**Table 1:** *Artix ESB Feature Comparison*

Feature	Java Runtime	C++ Runtime
<b>Supported APIs</b>		
JAXB 2.0	Yes	No
JAX-WS 2.0	Yes	No
WSDL 1.1	Yes	Yes
JCA Connector	Yes	No
<b>Development Language</b>		
Java	Yes	No
C++	No	Yes
JavaScript	Yes	No
<b>Bindings</b>		

**Table 1:** *Artix ESB Feature Comparison (Continued)*

<b>Feature</b>	<b>Java Runtime</b>	<b>C++ Runtime</b>
SOAP (1.1 and 1.2)	Yes	Yes
MTOM/XOP	Yes	Yes
RESTful	Yes	No
CORBA	Yes	Yes
Pure XML	Yes	Yes
Fixed length records	Via Artix Data Services	Yes
Tagged data	Via Artix Data Services	Yes
Tibco Rendezvous messages	No	Yes
FML	No	Yes
<b>Transports</b>		
HTTP	Yes	Yes
JMS	Yes	Yes
FTP	Yes	Yes
WebSphere MQ	Yes	Yes
IIOP	No	Yes
Tuxedo	No	Yes
<b>Quality of Service</b>		
Message routing	Yes	Yes
Security	Yes	Yes
Reliable messaging	Yes	Yes
High availability	Yes	Yes

**Table 1:** *Artix ESB Feature Comparison (Continued)*

<b>Feature</b>	<b>Java Runtime</b>	<b>C++ Runtime</b>
Load balancing	Yes	Yes
Location resolution	Yes	Yes
Statefulness	No	Yes

---

# Key Concepts in Depth

*This section discusses key Artix ESB concepts in depth.*

---

**In this section**

This section discusses the following topics:

<a href="#">Artix ESB Runtime Components</a>	page 33
<a href="#">Artix Bus</a>	page 34
<a href="#">Artix Endpoints</a>	page 35
<a href="#">Artix Contracts</a>	page 36
<a href="#">Artix Services</a>	page 38



---

## Artix ESB Runtime Components

---

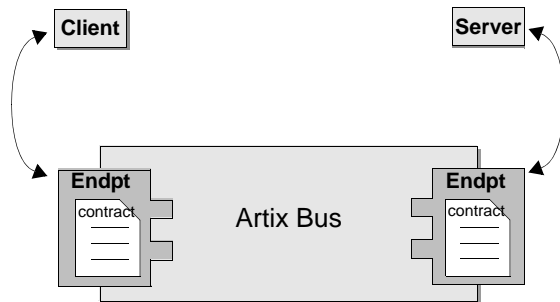
### How it fits together

The Artix ESB runtime consists of the following components :

- [Artix Bus](#) is at the core of Artix, and provides the support for various transports and payload formats.
- [Artix Contracts](#) describe your applications in such a way that they become services that can be deployed as [Artix Endpoints](#).
- [Artix Services](#) include a number of advanced services, such as the locator and session manager. Each Artix service is defined with an Artix contract and can be deployed as an Artix endpoint.

[Figure 1](#) illustrates how the Artix ESB elements fit together.

**Figure 1:** *Artix ESB Runtime Components*



## Artix Bus

---

### Overview

The Artix bus is at the heart of the Artix ESB architecture. It is the component that hosts the services that you create and connects your applications to those services. The bus is also responsible for translating data from one format into another.

In this way, Artix ESB enables all of the services in your company to communicate, without needing to communicate in the same way. It also means that clients can contact services without understanding the native language of the server handling requests.

---

### Benefits

While other products provide some ability to expose applications as services, they frequently require a good deal of coding. The Artix bus eliminates the need to modify your applications or write code by directly translating the application's native communication protocol into any of the other supported protocols.

For example, by deploying an Artix instance with a SOAP-over-WebSphere MQ endpoint and a SOAP-over-HTTP endpoint, you can expose a WebSphere MQ application directly as a Web service. The WebSphere MQ application does not need to be altered or made aware that it is being exposed using SOAP over HTTP.

The Artix bus translation facility also makes it a powerful integration tool. Unlike traditional EAI products, Artix translates directly between different middlewares without first translating into a canonical format. This saves processing overhead and increases the speed at which messages are transmitted.

---

# Artix Endpoints

---

## Overview

An Artix endpoint is the connection point at which a service or a service consumer connects to the Artix bus. Endpoints are described by a contract describing the services offered and the physical representation of the data on the network.

---

## Reconfigurable connection

An Artix endpoint provides an abstract connection point between applications, as shown in [Figure 1 on page 33](#). The benefit of this abstract connection is that it allows you to change the underlying communication mechanism without recoding any of your applications. You only need to modify the contract describing the endpoint.

For example, if one of your back-end service providers is a Tuxedo application and you want to swap it for a CORBA implementation, you simply change the endpoint's contract to contain a CORBA connection to the Artix bus. The clients accessing the back-end service provider do not need to be aware of the change.

---

# Artix Contracts

---

## Overview

Artix contracts are written in WSDL. In this way, a standard language is used to describe the characteristics of services and their associated Artix endpoints. By defining characteristics such as service operations and messages in an abstract way—independent of the transport or protocol used to implement the endpoint—these characteristics can be bound to a variety of protocols and formats.

Artix ESB allows an abstract definition to be bound to multiple specific protocols and formats. This means that the same definitions can be reused in multiple implementations of a service. Artix contracts define the services exposed by a set of systems, the payload formats and transports available to each system, and the rules governing how the systems interact with each other. The simplest Artix contract defines a single pair of systems with a shared interface, payload format, and transport. Artix contracts can also define very complex integration scenarios.

---

## WSDL elements

Understanding Artix contracts requires some familiarity with WSDL. The key WSDL elements are as follows:

**WSDL types** provide data type definitions used to describe messages.

**A WSDL message** is an abstract definition of the data being communicated. Each part of a message is associated with a defined type.

**A WSDL operation** is an abstract definition of the capabilities supported by a service, and is defined in terms of input and output messages.

**A WSDL portType** is a set of abstract operation descriptions.

**A WSDL binding** associates a specific data format for operations defined in a `portType`.

**A WSDL port** specifies the transport details for a binding, and defines a single communication endpoint.

**A WSDL service** specifies a set of related ports.

## The Artix Contract

---

An Artix contract is specified in WSDL and is conceptually divided into logical and physical components.

### The logical contract

The logical contract specifies components that are independent of the underlying transport and wire format. It fully specifies the data structure and the possible operations or interactions with the interface. It enables Artix to generate skeletons and stubs without having to define the physical characteristics of the connection (transport and wire format).

The logical contract includes the `types`, `message`, `operation`, and `portType` elements of the WSDL file.

### The physical contract

The physical component of an Artix contract defines the format and transport-specific details. For example:

- The wire format, middleware transport, and service groupings
- The connection between the `portType` operations and wire formats
- Buffer layout for fixed formats
- Artix extensions to WSDL

The physical contract includes the `binding`, `port`, and `service` elements of the WSDL file.

---

## Artix Services

---

### Overview

In addition to the core Artix components, Artix also provides the following services:

- [Container](#)
- [Locator](#)
- [Session manager](#)
- [Transformer](#)
- [Accessing contracts and references](#)

These services provide advanced functionality that Artix deployments can use to gain even more flexibility.

---

### Container

The Artix container provides a consistent mechanism for deploying and managing Artix services. It allows you to write Web service implementations as Artix plug-ins and then deploy your services into the Artix Container.

Using the container eliminates the need to write your own C++ or Java server mainline. Instead, you can deploy your service by simply passing the location of a generated deployment descriptor to the Artix container's administration client.

---

### Locator

The Artix locator provides service look-up and load balancing functionality to an Artix deployment. It isolates service consumers from changes in a service's contact information.

The Artix WSDL contract defines how the client contacts the server, and contains the address of the Artix locator. The locator provides the client with a reference to the server.

Servers are automatically registered with the locator when they start, and service endpoints are automatically made available to clients without the need for additional coding.

---

### Session manager

The Artix session manager is a group of plug-ins that work together to manage the number of concurrent clients that access a group of services. This allows you to control how long each client can use the services in the group before having to check back with the session manager.

In addition, the session manager has a pluggable policy callback mechanism that enables you to implement your own session management policies.

---

**Transformer**

The Artix transformer provides Artix ESB with a way to transform operation parameters on the wire using rules written in Extensible Style Sheet Transformation (XSLT) scripts. The transformer can be used to provide a simple means of transforming data. For example, it can be used to develop an application that accepts names as a single string and returns them as separate first and last name strings.

The transformer can also be placed between two applications where it can transform messages as they pass between the applications. This functionality allows you to connect applications that do not use exactly the same interfaces and still realize the benefits of not using a canonical format without rewriting the underlying applications.

---

**Accessing contracts and references**

Accessing contracts and references in Artix ESB refers to enabling client and server applications to find WSDL service contracts and references. Using the techniques and conventions of Artix avoids the need to hard code WSDL into your client and server applications.

---

**For more information**

For more information on Artix services, see [Configuring and Deploying Artix](#).

---

# Solving Problems with Artix ESB

---

## Overview

Artix ESB allows you to solve problems arising from the integration of existing back-end systems using a service-oriented approach. It allows you to develop new services using C++ or Java, and to retain all of the enterprise levels of service that you require.

There are three phases to an Artix ESB project:

1. The design phase, where you define your services and define how they are integrated using Artix contracts.
2. The development phase, where you write the application code required to implement new services.
3. The deployment phase, where you configure and deploy your Artix solution.

---

## Design phase

In the design phase, you define the logical layout of your system in an Artix contract. The logical or abstract definition of a system includes:

- the services that it contains
- the operations each service offers
- the data the services will use to exchange information

Once you have defined the logical aspects of your system, you then add the physical network details to the contracts.

The physical details of your system include the transports and payload formats used by your services, as well as any routing schemes needed to connect services that use different transports or payload formats.



The Artix command-line tools automate the mapping of your service descriptions into WSDL-based Artix contracts. These tools allow you to:

- Generate Artix contracts from:
  - ◆ CORBA IDL
  - ◆ A description of tagged data
  - ◆ A description of fixed record length data
  - ◆ A COBOL copybook
  - ◆ A Java class
- Add the following bindings to an Artix contract:
  - ◆ CORBA
  - ◆ Fixed record length
  - ◆ SOAP
  - ◆ Tagged data
  - ◆ XML

---

## Development phase

You must write Artix application code if your solution involves creating new applications or a custom router, or involves using the Artix session management feature. The first step in writing Artix code is to generate client stub code and server skeleton code from the Artix contracts that you created in the design phase. You can generate this code using the Artix command-line tools.

After you have generated the client stub code and server skeleton code, you can develop the code that implements the business logic you require. For most applications, Artix-generated code allows you to stick to using standard C++ or Java code for writing business logic.

Artix ESB also provides advanced APIs for directly manipulating messages, for writing message handlers, and for other advanced features your application might require. These can be plugged into the Artix runtime for customized processing of messages.

---

## Deployment phase

In the deployment phase, you configure the Artix runtime to fine-tune the Artix bus for your new Artix system. This involves modifying the Artix configuration files and editing the Artix contracts that describe your solution to fit the exact circumstances of your deployment environment.

This phase also includes the managing of the deployed system. This might involve, for example, using an enterprise management tool along with the Artix command interface. These tools allow you to further fine-tune your system.

# Understanding WSDL

*Artix contracts use WSDL documents to describe services and the data they use.*

---

**In this chapter**

This chapter discusses the following topics:

WSDL Basics	page 44
Abstract Data Type Definitions	page 46
Abstract Message Definitions	page 49
Abstract Interface Definitions	page 52
Mapping to the Concrete Details	page 56

# WSDL Basics

---

## Overview

Web Services Description Language (WSDL) is an XML document format used to describe services offered over the Web. WSDL is standardized by the World Wide Web Consortium (W3C) and is currently at revision 1.1. You can find the standard on the W3C website at [www.w3.org/TR/wsdl](http://www.w3.org/TR/wsdl).

---

## Elements of a WSDL document

A WSDL document is made up of the following elements:

- `import` allows you to import another WSDL or XSD file.
- Logical contract elements:
  - ◆ `types`
  - ◆ `message`
  - ◆ `operation`
  - ◆ `portType`
- Physical contract elements:
  - ◆ `binding`
  - ◆ `port`
  - ◆ `service`

These elements are described in “[WSDL elements](#)” on page 36.

---

## Abstract operations

The abstract definition of *operations* and *messages* is separated from the concrete data formatting definitions and network protocol details. As a result, the abstract definitions can be reused and recombined to define several endpoints. For example, a service can expose identical operations with slightly different concrete data formats and two different network addresses. Alternatively, one WSDL document could be used to define several services that use the same abstract messages.

---

## The portType

A *portType* is a collection of abstract operations that define the actions provided by an endpoint.

---

**Concrete details**

When a `portType` is mapped to a concrete data format, the result is a concrete representation of the abstract definition. A *port* is defined by associating a network address with a reusable *binding*, in the form of an endpoint. A collection of ports (or endpoints) define a service.

Because WSDL was intended to describe services offered over the Web, the concrete message format is typically SOAP and the network protocol is typically HTTP. However, WSDL documents can use any concrete message format and network protocol. In fact, Artix contracts bind operations to several data formats and describe the details for a number of network protocols.

---

**Namespaces and imported descriptions**

WSDL supports the use of XML namespaces defined in the `definition` element as a way of specifying predefined extensions and type systems in a WSDL document. WSDL also supports importing WSDL documents and fragments for building modular WSDL collections.

---

**Example**

[Example 1 on page 86](#) shows a simple WSDL document.

---

# Abstract Data Type Definitions

---

## Overview

Applications typically use data types that are more complex than the primitive types, like `int`, defined by most programming languages. WSDL documents represent these complex data types using a combination of schema types defined in referenced external XML schema documents and complex types described in `types` elements.

## Complex type definitions

Complex data types are described in a `types` element. The W3C specification states that XSD is the preferred canonical type system for a WSDL document. Therefore, XSD is treated as the intrinsic type system. Because these data types are abstract descriptions of the data passed over the wire, and are not concrete descriptions, there are a few guidelines on using XSD schemas to represent them:

- Use elements, not attributes.
- Do not use protocol-specific types as base types.
- Define arrays using the SOAP 1.1 array encoding format.

WSDL does allow for the specification and use of alternative type systems within a document.

## Example

The structure, `personalInfo`, defined in [Example 1](#), contains a `string`, an `int`, and an `enum`. The `string` and the `int` both have equivalent XSD types and do not require special type mapping. The enumerated type `hairColorType`, however, does need to be described in XSD.

### Example 1: *personalInfo* structure

```
enum hairColorType {red, brunette, blonde};

struct personalInfo
{
    string name;
    int age;
    hairColorType hairColor;
}
```

[Example 2](#) shows one mapping of `personalInfo` into XSD. This mapping is a direct representation of the data types defined in [Example 1](#). `hairColorType` is described using a named `simpleType` because it does not have any child elements. `personalInfo` is defined as an `element` so that it can be used in messages later in the contract.

**Example 2:** *XSD type definition for `personalInfo`*

```
<types>
  <xsd:schema targetNamespace="http://iona.com/personal/schema"
    xmlns:xsd1="http://iona.com/personal/schema"
    xmlns="http://www.w3.org/2000/10/XMLSchema" />
  <simpleType name="hairColorType">
    <restriction base="xsd:string">
      <enumeration value="red" />
      <enumeration value="brunette" />
      <enumeration value="blonde" />
    </restriction>
  </simpleType>
  <element name="personalInfo">
    <complexType>
      <sequence>
        <element name="name" type="xsd:string" />
        <element name="age" type="xsd:int" />
        <element name="hairColor" type="xsd1:hairColorType" />
      </sequence>
    </complexType>
  </element>
</types>
```

Another way to map `personalInfo` is to describe `hairColorType` in-line as shown in [Example 3](#). With this mapping, however, you cannot reuse the description of `hairColorType`.

**Example 3:** *Alternate XSD Mapping for `personalInfo`*

```
<types>
  <xsd:schema targetNamespace="http://iona.com/personal/schema"
    xmlns:xsd1="http://iona.com/personal/schema"
    xmlns="http://www.w3.org/2000/10/XMLSchema" />
  <element name="personalInfo">
    <complexType>
      <sequence>
        <element name="name" type="xsd:string" />
        <element name="age" type="xsd:int" />
      </sequence>
    </complexType>
  </element>
</types>
```

**Example 3:** *Alternate XSD Mapping for personalInfo (Continued)*

```
<element name="hairColor">
  <simpleType>
    <restriction base="xsd:string">
      <enumeration value="red"/>
      <enumeration value="brunette"/>
      <enumeration value="blonde"/>
    </restriction>
  </simpleType>
</element>
</sequence>
</complexType>
</element>
</types>
```



---

# Abstract Message Definitions

---

## Overview

WSDL is designed to describe how data is passed over a network. It describes data that is exchanged between two endpoints in terms of abstract messages described in `message` elements.

Each abstract message consists of one or more parts, defined in `part` elements.

These abstract messages represent the parameters passed by the operations defined by the WSDL document and are mapped to concrete data formats in the WSDL document's `binding` elements.

---

## Messages and parameter lists

For simplicity in describing the data consumed and provided by an endpoint, WSDL documents allow abstract operations to have only one input message, the representation of the operation's incoming parameter list, and only one output message, the representation of the data returned by the operation.

In the abstract message definition, you cannot directly describe a message that represents an operation's return value. Therefore, any return value must be included in the output message.

Messages allow for concrete methods defined in programming languages like C++ to be mapped to abstract WSDL operations. Each message contains a number of `part` elements that represent one element in a parameter list.

Therefore, all of the input parameters for a method call are defined in one message and all of the output parameters, including the operation's return value, are mapped to another message.

**Example**

---

For example, imagine a server that stores personal information as defined in [Example 1 on page 46](#) and provides a method that returns an employee's data based on an employee ID number.

The method signature for looking up the data would look similar to [Example 4](#).

**Example 4:** *Method for Returning an Employee's Data*

```
personalInfo lookup(long empId)
```

This method signature could be mapped to the WSDL fragment shown in [Example 5](#).

**Example 5:** *WSDL Message Definitions*

```
<message name="personalLookupRequest">
  <part name="empId" type="xsd:int" />
</message>
<message name="personalLookupResponse">
  <part name="return" element="xsd1:personalInfo" />
</message>
```

**Message naming**

---

Each message in a WSDL document must have a unique name within its namespace. Choose message names that show whether they are input messages (requests) or output messages (responses).

## Message parts

Message parts are the formal data elements of the abstract message. Each part is identified by a `name` attribute and by either a `type` or an `element` attribute that specifies its data type. The data type attributes are listed in [Table 2](#).

**Table 2:** *Part Data Type Attributes*

Attribute	Description
<code>type="type_name"</code>	The data type of the part is defined by a <code>simpleType</code> or <code>complexType</code> called <code>type_name</code>
<code>element="elem_name"</code>	The data type of the part is defined by an <code>element</code> called <code>elem_name</code> .

Messages are allowed to reuse part names. For instance, if a method has a parameter, `foo`, which is passed by reference or is an in/out, it can be a part in both the request message and the response message. An example of parameter reuse is shown in [Example 6](#).

**Example 6:** *Reused Part*

```
<message name="fooRequest">
  <part name="foo" type="xsd:int"/>
</message>
<message name="fooReply">
  <part name="foo" type="xsd:int"/>
</message>
```

---

# Abstract Interface Definitions

---

## Overview

WSDL `portType` elements define, in an abstract way, the operations offered by a service. The operations defined in a `portType` list the input, output, and any fault messages used by the service to complete the transaction the operation describes.

---

## PortTypes

A `portType` can be thought of as an interface description. In many Web service implementations there is a direct mapping between `portTypes` and implementation objects. `PortTypes` are the abstract unit of a WSDL document that is mapped into a concrete binding to form the complete description of what is offered over a port.

`PortTypes` are described using the `portType` element in a WSDL document. Each `portType` in a WSDL document must have a unique name, specified using the `name` attribute, and is made up of a collection of operations, described in `operation` elements. A WSDL document can describe any number of `portTypes`.

---

## Operations

Operations, described in `operation` elements in a WSDL document, are an abstract description of an interaction between two endpoints. For example, a request for a checking account balance and an order for a gross of widgets can both be defined as operations.

Each operation within a `portType` must have a unique name, specified using the required `name` attribute.

## Elements of an operation

Each operation is made up of a set of elements. The elements represent the messages communicated between the endpoints to execute the operation.

The elements that can describe an operation are listed in [Table 3](#).

**Table 3:** *Operation Message Elements*

Element	Description
input	Specifies a message that is received from another endpoint. This element can occur at most once for each operation.
output	Specifies a message that is sent to another endpoint. This element can occur at most once for each operation.
fault	Specifies a message used to communicate an error condition between the endpoints. This element is not required and can occur an unlimited number of times.

An operation is required to have at least one `input` or `output` element. The elements are defined by two attributes listed in [Table 4](#).

**Table 4:** *Attributes of the Input and Output Elements*

Attribute	Description
name	Identifies the message so it can be referenced when mapping the operation to a concrete data format. The name must be unique within the enclosing port type.
message	Specifies the abstract message that describes the data being sent or received. The value of the <code>message</code> attribute must correspond to the <code>name</code> attribute of one of the abstract messages defined in the WSDL document.

It is not necessary to specify the `name` attribute for all input and output elements; WSDL provides a default naming scheme based on the enclosing operation's name.

If only one element is used in the operation, the element name defaults to the name of the operation. If both an `input` and an `output` element are used, the element name defaults to the name of the operation with `Request` or `Response`, respectively, appended to the name.

---

### Return values

Because the `portType` is an abstract definition of the data passed during an operation, WSDL does not provide for return values to be specified for an operation. If a method returns a value, it is mapped into the output message as the last part of that message. The concrete details of how the message parts are mapped into a physical representation are described in [“Bindings” on page 56](#).

---

### Example

For example, in implementing a server that stores personal information in the structure defined in [Example 1 on page 46](#), you might use an interface similar to the one shown in [Example 7](#).

**Example 7:** *personalInfo Lookup Interface*

```
interface personalInfoLookup
{
    personalInfo lookup(in int empID)
    raises(idNotFound);
}
```

This interface could be mapped to the portType in [Example 8](#).

**Example 8:** *personalInfo Lookup Port Type*

```

<types>
...
  <element name="idNotFound" type="idNotFoundType">
    <complexType name="idNotFoundType">
      <sequence>
        <element name="ErrorMsg" type="xsd:string" />
        <element name="ErrorID" type="xsd:int" />
      </sequence>
    </complexType>
  </types>
<message name="personalLookupRequest">
  <part name="empId" type="xsd:int" />
</message>
<message name="personalLookupResponse">
  <part name="return" element="xsd1:personalInfo" />
</message>
<message name="idNotFoundException">
  <part name="exception" element="xsd1:idNotFound" />
</message>
<portType name="personalInfoLookup">
  <operation name="lookup">
    <input name="empID" message="personalLookupRequest" />
    <output name="return" message="personalLookupResponse" />
    <fault name="exception" message="idNotFoundException" />
  </operation>
</portType>

```

---

# Mapping to the Concrete Details

---

## Overview

The abstract definitions in a WSDL document are intended to be used in defining the interaction of real applications that have specific network addresses, use specific network protocols, and expect data in a particular format. To fully define these real applications, the abstract definitions discussed in the previous section must be mapped to concrete representations of the data passed between applications. The details describing the network protocols in use must also be added.

This is accomplished in the WSDL `bindings` and `ports` elements. WSDL binding and port syntax is not tightly specified by the W3C. A specification is provided that defines the mechanism for defining these syntaxes. However, the syntaxes for bindings other than SOAP and for network transports other than HTTP are not defined in a W3C specification.

---

## Bindings

Bindings describe the mapping between the abstract messages defined for each `portType` and the data format used on the wire. Bindings are described in `binding` elements in the WSDL file. A binding can map to only one `portType`, but a `portType` can be mapped to any number of bindings.

It is within the bindings that you specify details such as parameter order, concrete data types, and return values. For example, a binding can reorder the parts of a message to reflect the order required by an RPC call. Depending on the binding type, you can also identify which of the message parts, if any, represent the return type of a method.

---

## Services

To define an endpoint that corresponds to a running service, the `port` element in the WSDL file associates a binding with the concrete network information needed to connect to the remote service described in the file. Each port specifies the address and configuration information for connecting the application to a network.

Ports are grouped within `service` elements. A service can contain one or many ports. The convention is that the ports defined within a particular service are related in some way. For example, all of the ports might be bound to the same `portType`, but use different network protocols, like HTTP and WebSphere MQ.



# Index

## A

Apache CXF 29  
Artix  
  bus 34  
  contracts 36, 37  
  locator 38  
  session manager 38  
  transformer 39

## B

bindings 36, 56  
bus 34

## C

C++ Runtime 29  
contracts 36, 37  
CORBA IDL 41

## D

deployment phase 42  
design phase 40  
development phase 42

## E

EAI 27  
enterprise application integration, see EAI  
enterprise service bus, See ESB

## I

IDL 41

## J

Java Runtime 29

## L

locator 38

## M

messages 36

## O

operations 36, 52

## P

ports 36  
portTypes 36, 44, 52

## R

runtimes  
  C++ 29  
  Java 29

## S

service-oriented architecture, see SOA  
services 36, 56  
session manager 38  
SOA 26  
SOAP 26

## T

transformer 39  
types 36

## W

W3C 44  
Web Services Description Language, see WSDL  
World Wide Web Consortium, see W3C  
WSDL 36, 43–56  
  defined 44

## X

XSD 46

