



The Dbmlsync API

A whitepaper from Sybase iAnywhere

Author: Joshua Savill, Product Manager

Date: October 30th, 2008

This whitepaper was written in the context of SQL Anywhere 11.

CONTENTS

Introduction	3
Dbmsync API Architecture	3
Dbmsync API Documentation	5
Dbmsync API for C++	5
Dbmsync API for .NET	5
Using the DBSC_Event Structure	5
Using the Dbmsync API	6
Starting the Dbmsync Server	7
Synchronizing with the Dbmsync Server	8
Shutting Down the Dbmsync Server	8
Summary	9

FIGURES

Figure 1 – MobiLink Synchronization	3
Figure 2 – Dbmsync API and Dbmsync Server	4
Figure 3 – Sequence Flow for Dbmsync API	7

INTRODUCTION

In SQL Anywhere 11, the Dbmsync API provides a programming interface that allows applications written in C++ or .NET development languages to launch MobiLink synchronization seamlessly. Feedback information generated by the synchronization is provided back to the application.

A .NET example of the Dbmsync API can be found on the Sybase iAnywhere website here: <http://www.sybase.com/detail?id=1060464>

DBMSLYNC API ARCHITECTURE

Synchronization typically begins when the MobiLink client opens a connection with the MobiLink server. The MobiLink client uploads data changes made to the remote database since the last complete synchronization. The MobiLink server applies the data changes to the consolidated database, and then generates a download of all changes in the consolidated database since the last complete synchronization. The MobiLink client applies the changes to the remote database. Figure 1 shows the typical MobiLink architecture.

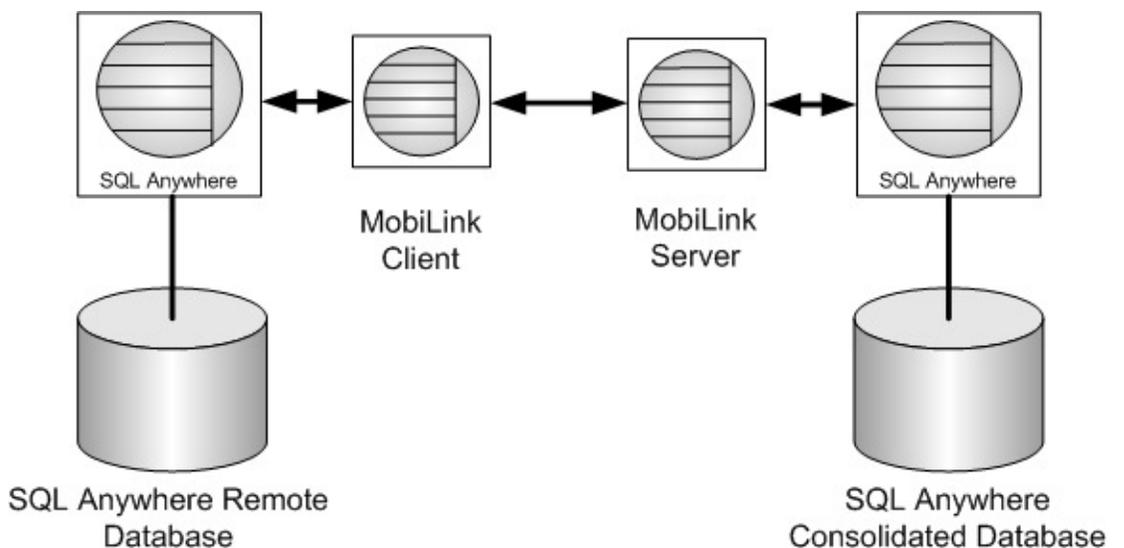


Figure 1 – MobiLink Synchronization

When using the Dbmsync API, synchronization is architecturally the same as for a typical MobiLink synchronization. The MobiLink client still communicates with the MobiLink server to perform the synchronization. The difference is the way the application communicates with the MobiLink client to initiate communication with the MobiLink server. When using the Dbmsync API, the MobiLink client is `dbmsync`.

There are two versions of the Dbmsync API: C++ and .NET.

The C++ version is implemented in the DLL `dbmsynccli11.dll`. Applications that use the C++ version need to include the header `dbmsynccli.hpp` in the C++ code and link against the import library `dbmsynccli11.lib`. The `dbmsynccli11.dll` must be deployed with the application.

The .NET version is implemented in the *iAnywhere.MobiLink.Client.dll* assembly. Applications using the .NET version must include a reference to the assembly in the .NET project. The *iAnywhere.MobiLink.Client.dll* assembly must be deployed with the application.

To use the Dbmsync API, the client application instantiates and calls methods provided in the DbmsyncClient class. The DbmsyncClient class communicates with the dbmsync server, using TCP/IP, to initiate a synchronization. The dbmsync server performs the synchronization by connecting to the MobiLink server and SQL Anywhere remote database. The dbmsync server is actually the dbmsync client running in a listening mode waiting for synchronization requests. The dbmsync server and application must be running on the same machine. Figure 2 shows the architecture of the Dbmsync API and the dbmsync server.

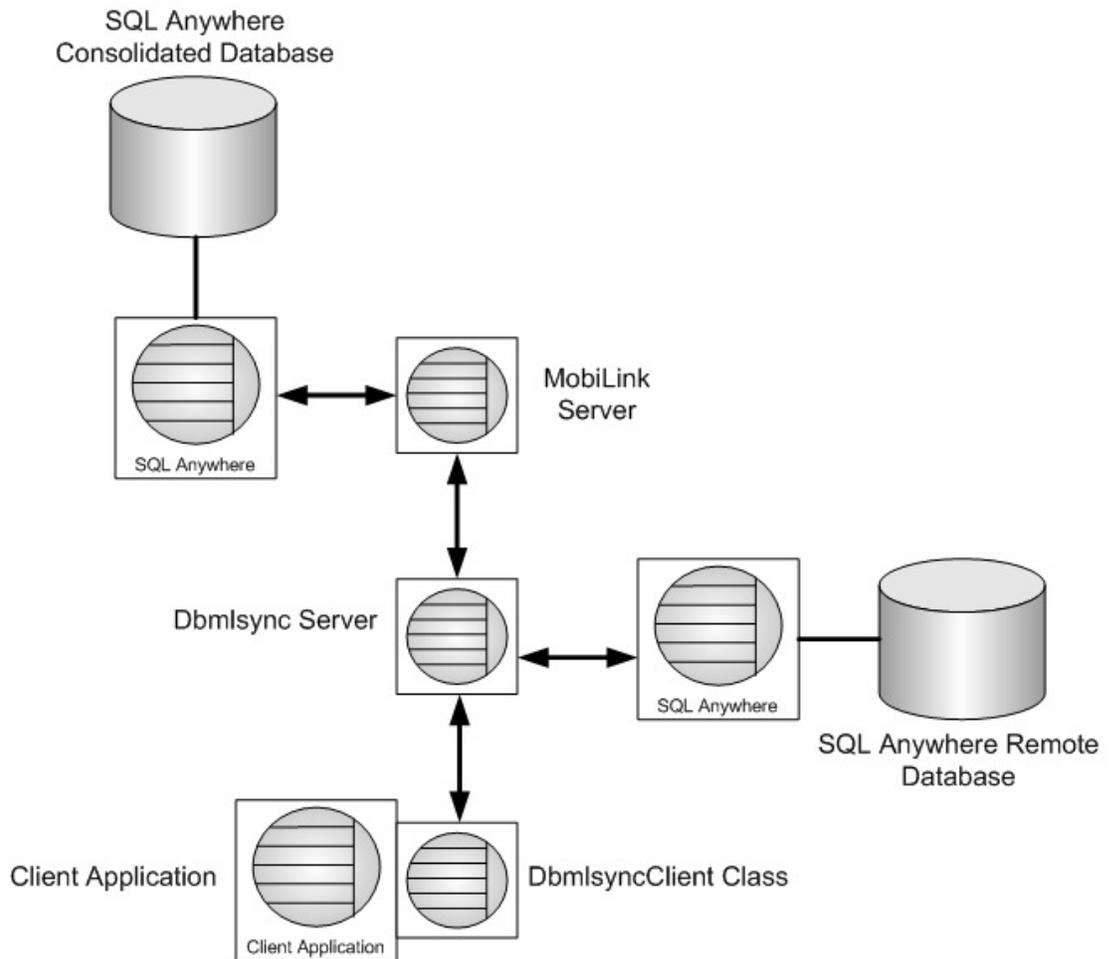


Figure 2 – Dbmsync API and Dbmsync Server

During a synchronization, the dbmsync server provides a series of events that contain information and feedback about the progress of the synchronization. These events are sent from the dbmsync server to the DbmsyncClient class and queued. The GetEvent method, which is part of the DbmsyncClient class, is called by the client application to retrieve the next queued event.

In this architecture, the dbmsync server can only be connected to and synchronize a single remote database. Client applications have the ability to share a dbmsync server, however only one synchronization can be performed on a remote database at a time. If the dbmsync server receives a synchronization request while performing a synchronization, the request is queued and executed by the dbmsync server when the first synchronization is complete.

DBMSYNC API DOCUMENTATION

The Dbmsync API programming interface is fully explained in the SQL Anywhere 11 documentation.

DBMSYNC API FOR C++

The Dbmsync API methods for the C++ implementation of the DbmsyncClient class are documented in the following section of the SQL Anywhere 11 documentation:

[MobiLink - Client Administration](#) »
[SQL Anywhere Clients for MobiLink](#) »
[Dbmsync API](#) »

Dbmsync API for C++

Online:

<http://dcx.sybase.com/index.php#http%3A%2F%2Fdcx.sybase.com%2F1100en%2Fmliclienten11%2Fmc-dbmsyncserv-s-4174851.html>

DBMSYNC API FOR .NET

The Dbmsync API methods for the .NET implementation of the DbmsyncClient class are documented in the following section of the SQL Anywhere 11 documentation:

[MobiLink - Client Administration](#) »
[SQL Anywhere Clients for MobiLink](#) »
[Dbmsync API](#) »

Dbmsync API for .NET

Online:

<http://dcx.sybase.com/index.php#http%3A%2F%2Fdcx.sybase.com%2F1100en%2Fmliclienten11%2Fmc-dbmsyncserv-s-4311724.html>

USING THE DBSC_EVENT STRUCTURE

While the dbmsync server is executing a synchronization it generates a series of events that contain information and feedback about the progress of the synchronization. Each event is sent from the dbmsync server to the DbmsyncClient class and stored in a DBSC_Event structure. The DBSC_Event structures are queued during the synchronization. A call to the GetEvent method retrieves the next DBSC_Event structure that is queued. If no DBSC_Event structure can be retrieved, the GetEvent method waits until the next one is available, or times out based on the specified expired time.

The types of events generated during a synchronization can be controlled by a series of properties that are applied when the GetEvent method is called. The SetProperty method can enable and disable certain types of events during the synchronization. Disabling events that are not required can improve performance of the overall synchronization.

The DBSC_Event Structure is documented in the following sections of the SQL Anywhere 11 documentation:

[MobiLink - Client Administration](#) »
[SQL Anywhere Clients for MobiLink](#) »
[Dbmsync API](#) »
[Dbmsync API for C++](#) »

DBSC_Event structure

Online:

<http://dcx.sybase.com/index.php#http%3A%2F%2Fdcx.sybase.com%2F1100en%2Fmliclienten11%2Fmc-dbmsyncserv-s-5226880.html>

[MobiLink - Client Administration](#) »

[SQL Anywhere Clients for MobiLink](#) »

[Dbmsync API](#) »

[Dbmsync API for .NET](#) »

DBSC_Event structure

Online:

<http://dcx.sybase.com/index.php#http%3A%2F%2Fdcx.sybase.com%2F1100en%2Fmliclienten11%2Fmc-dbmsyncserv-s-5226880net.html>

USING THE DBMSYNC API

The DbmsyncClient class needs to be instantiated before using the Dbmsync API.

For C++, include the header *dbmsynccli.hpp* found in the *%SQLANY11%\SDK\Include* folder and link against *dbmsynccli.lib* found in the *%SQLANY11%\SDK\Lib\X86* folder. Deploy the application with the *dbmsynccli.11.dll* found in the *%SQLANY11%\bin32* folder

For .NET, reference must be made to the iAnywhere.MobiLink.Client component/DLL. Typically this assembly is already integrated into the .NET components list, but if not there, it can be found in the *%SQLANY11%\Assembly\V2* directory.

Figure 3 shows the sequence flow when using the Dbmsync API for synchronization.

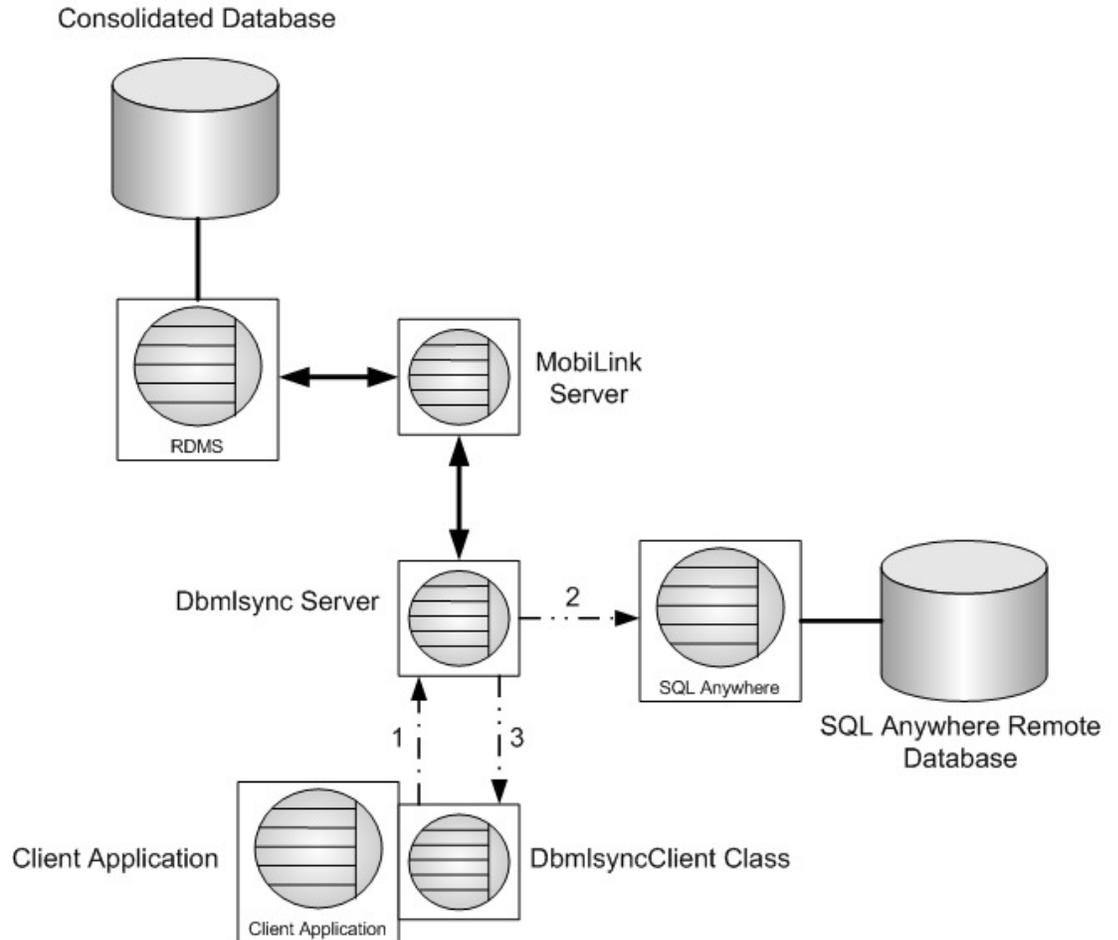


Figure 3 – Sequence Flow for Dbmsync API

The sequence of flow for the Dbmsync API is broken into the 3 phases. During phase 1, the client application instantiates the DbmsyncClient class, starts the dbmsync server, and establishes a connection with the dbmsync server.

In phase 2, the client application calls the Sync method to request that the dbmsync server perform a synchronization. The dbmsync server will establish a connection with the database at this time. All other synchronization requests are queued.

While the synchronization is being performed, phase 3 shows the dbmsync server generating a series of events that contain information and feedback about the progress. Each event is sent from the dbmsync server to the DbmsyncClient class and stored in a DBSC_Event structure. The DBSC_Event structures are queued. The GetEvent method is used to retrieve queued DBSC_Event structures.

STARTING THE DBMSYNC SERVER

Here is a .NET example showing how to start the dbmsync server:

```
using System;
using System.Collections.Generic;
using System.Text;
using iAnywhere.MobiLink.Client;

namespace StartServer {
```

```

class Program {
    static void Main(string[] args) {
        DbmlsyncClient client = DbmlsyncClient.InstantiateClient();
        DBSC_StartType starttype;

        client.Init();
        client.StartServer(2700, "-c \"DSN=api_rem\"", 3000, out
starttype);
        client.Fini();
    }
}

```

SYNCHRONIZING WITH THE DBMLSYNC SERVER

Here is a .NET example showing how to request a synchronization with the dbmlsync server. The dbmlsync server is already running with StartServer .NET execution above:

```

using System;
using System.Collections.Generic;
using System.Text;
using iAnywhere.MobiLink.Client;

namespace Sync {
    class Program {
        static void Main(string[] args) {
            DbmlsyncClient client = DbmlsyncClient.InstantiateClient();
            DBSC_Event evt = new DBSC_Event();

            client.Init();
            client.Connect("localhost", 2700, "dba", "sql");
            client.Sync("profile1", "Verbosity=HIGH");

            //Loop until DBSC_EVENTTYPE_SYNC_DONE
            while (client.GetEvent(ref evt, 3000) ==
DBSC_GetEventRet.DBSC_GETEVENT_OK) {
                //do stuff
                if (evt.type ==
DBSC_EventType.DBSC_EVENTTYPE_SYNC_DONE) {
                    break;
                }
            }
            client.Disconnect();
            client.Fini();
        }
    }
}

```

SHUTTING DOWN THE DBMLSYNC SERVER

Here is a .NET example showing how to shut down the dbmlsync server.

```

using System;
using System.Collections.Generic;
using System.Text;

```

```

using iAnywhere.MobiLink.Client;

namespace StopServer {
    class Program {
        static void Main(string[] args) {
            DbmlsyncClient client = DbmlsyncClient.InstantiateClient();

            client.Init();
            client.Connect("localhost", 2700, "dba", "sql");

            client.ShutdownServer(DBSC_ShutdownType.DBSC_SHUTDOWN_ON_EMPTY_QUEUE);
            client.WaitForServerShutdown(5000);
            client.Fini();
        }
    }
}

```

SUMMARY

This document outlines how the Dbmlsync API works and provides documentation links to explain the API. Also provided are .NET examples to show how to start the dbmlsync server, initiate synchronization, and shutdown the dbmlsync server.

Copyright © 2008 iAnywhere Solutions, Inc. All Rights Reserved. Sybase, Afaia, SQL Anywhere, Adaptive Server Anywhere, MobiLink, UltraLite, and M-Business Anywhere are trademarks of Sybase, Inc. All other trademarks are property of their respective owners.

Copyright

© Copyright 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects S.A. in the United States and in other countries. Business Objects is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.