



Antheus Technology, Inc.

Agora SDK 3.00 for .NET Framework 2.0

Demonstrations

The Agora SDK includes four demonstrations that cover the most relevant functionality. The C# source code of these demonstrations is included in the SDK to illustrate the developer on how to use the technology, and the executable can be used to show the technology to end users.

- **Verification** allows you to load from file or live capture two fingers and compare them, and view the result in a graphical representation.
- **WsqCodec** allows you to load from file or live capture one finger image and immediately see the WSQ compressed image side by side
- **Rolled** allows you to capture a rolled fingerprint using a Cross Match Verifier 310.
- **MatchDemoSql** allows you interact with Scheduler and its MatchServer(s), to connect to an SQL Server 2005 database, enroll a person with one to ten fingers and simple demographic data, and then search with one or more fingers against the templates stored in the MatchServer(s) memory through the Scheduler. The data is stored in the SQL Server 2005 database.

Assemblies

The Agora SDK 3.00 includes the following .NET Framework 2.0 assemblies within the namespace Antheus:

- **Analysis**, the class AgrFeature has methods Encode to extract the fingerprint minutiae template and Quality to determine quality, plus other classes to store information
- **Matching**, the class AgrVerify includes two methods with two algorithms to match fingerprints: Matching and Verification; and VerificationDraw to plot the similarity; then there is a class to store similarity information
- **Compress**, the class AgrWsqa uses FBI certifier WSQ compression of BMP and RAW, and decompression; and classes to store compress and decompress information
- **Identification**, the class MatchClientSql includes methods to communicate a 1:N identification transaction with the Scheduler / MatchServers
- **Plain**, the class Plain has methods to command the plain impression capture of a Cross Match Verifier 300 LC2 and Verifier 310
- **Capture**, has numerous classes manipulate fingerprint images, to capture plain and rolled prints with Cross Match Verifier 300 and 310, NitGen and FlashBus frame grabber; and interface for rolled capture
- **Display**, has classes to view the finger image and its options

Scheduler-MatchServer

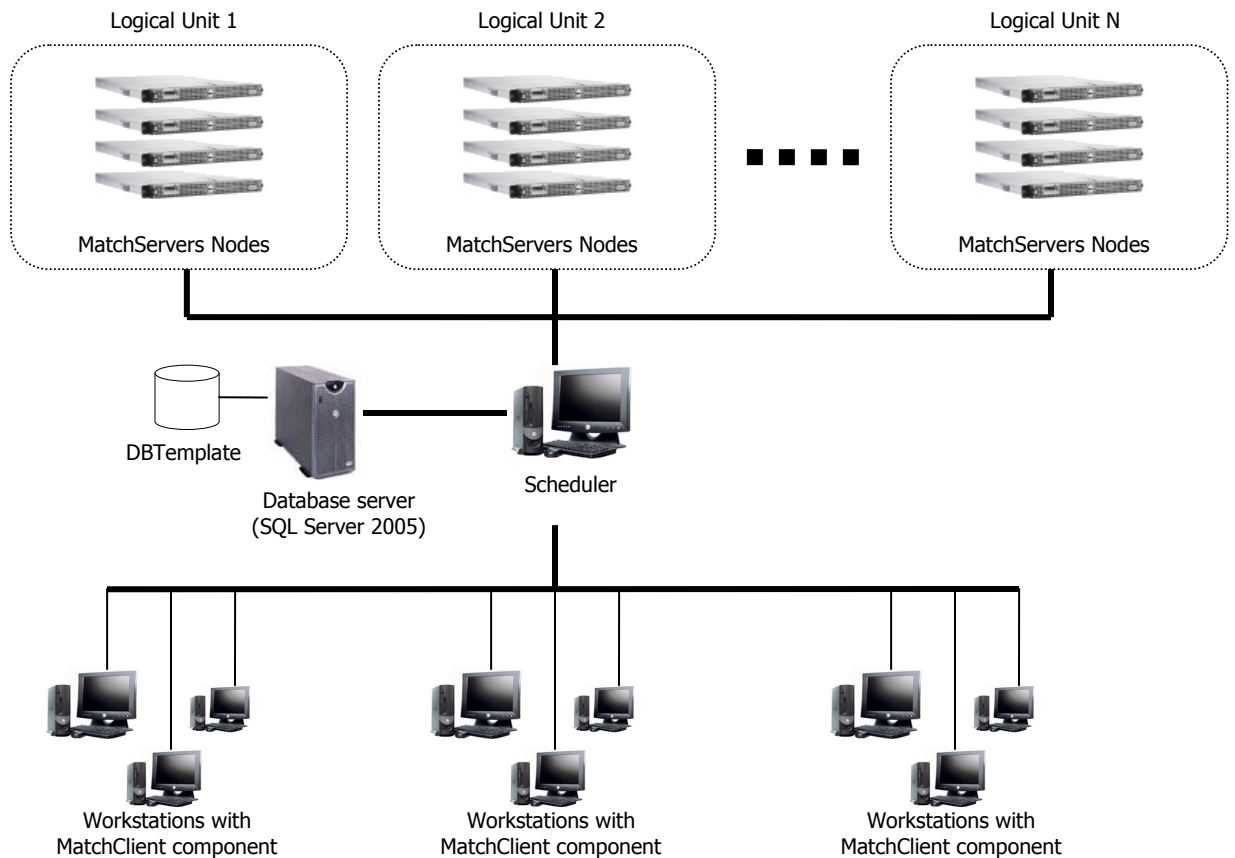
This section explains how to setup and use the **Scheduler** and the **MatchServer(s)**. It contains two sub sections:



Antheus Technology, Inc.

- The **Reference** section explains how to use the Scheduler and the MatchServer.
- The **Setup** section explains how to setup a sample system that consists of one Scheduler and two MatchServers

The Scheduler controls one or more clusters called **Logical Units** (LU) that may contain one or more MatchServer Nodes (MS) working in parallel in cluster architecture. The workstations send tasks to the Scheduler. The Scheduler checks which LU is not busy and assigns the task. The MS nodes on that LU, each with its portion of the database in RAM, works on the task and returns a result. The Scheduler updates the database (**DBTemplate**) and returns information to the workstation MC. The Scheduler uses one instance of a Microsoft SQL Server 2005 database to store the identity records. The Scheduler controls one or more LU working in parallel. Each LU contains all templates in RAM distributed among the MS in that cluster.





Antheus Technology, Inc.

Since all the templates are in RAM there is minimum disk I/O and this improves performance dramatically. The templates are loaded in RAM when the MS is started and subsequently loaded.

If a given configuration demands more performance more MS may be added to a LU, or a new LU may be configured. This ensures scalability in two ways. More MS in a cluster LU will maintain desired performance as the database grows because all the templates are redistributed into more MS; and more LU's will maintain performance when the number of transactions per day grow when more workstations are added to the system because the Scheduler will tap LU resources in idle state more often.

The Scheduler receives the tasks requested by the MC components in client workstations software applications, and schedules its distribution to the LUi. In its scheduling role when handling one task request, the Scheduler checks to see which LU are in a "free" state at the time, assigns one task to one LU, and changes the state of that LU to "busy". In this process the Scheduler optimizes the use of the resources. Once it gets back the result of the task, it dispatches the result to the client workstation that generated the task request, and "frees" the Logical Unit for its next task.

The Scheduler communicates with the MatchServers through TCP/IP port 1433.

The benefits of this architecture are:

- Scalability. As the database grows, and the number of client computer grows, the customer may want to maintain a response time. Additional LU may be added, and / or additional MS nodes may be added to each LU to accomplish the needed response time.
- Speed. If the customer needs faster response time, the same applies. Add Logical Units and / or add MatchServer Nodes to the Logical Units.
- Reliability. Additional LU may be set as stand-by and waiting to be used in case of a catastrophic failure.
- Lower total cost of ownership. The project may start with one LU with one MS node and grow when needed. The system is strictly software based, and there is no proprietary hardware. All computers are commercial off-the-shelf type (COT), based on Intel type CPU's and running Windows operating systems, on a standard TCP/IP protocol network. The system could be managed by regular information technology (IT) personnel with no need for specialists.

Sample implementations

A small, a medium and a large fictitious project to show examples of configurations that fulfill customer needs. The purpose of this section is to illustrate on the possible applications of the Agora technology in real life personal identification projects.