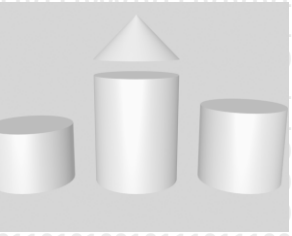


ODABA^{NG}

Remote Processing

run Software-Werkstatt GmbH
Weigandufer 45
12059 Berlin

Tel: +49 (30) 609 853 44
e-mail: run@run-software.com
web: www.run-software.com

Berlin, October 2012

Content

1 Introduction

ODABA^{NG}

ODABA^{NG} is an object-oriented database system that allows storing objects and methods as well as causalities. As an object-oriented database, ODABA^{NG} supports complex objects (user-defined data types), which are built on application relevant concepts.

ODABA^{NG} applications are characterised by a high flexibility that is achieved by supporting in addition to object (concept) hierarchy, multifarious relations between objects (master and detail relations, relations between independent objects and others). This way conditions and behaviour of objects in the real world can be represented considerably better than in relational systems.

ODABA^{NG} applications cannot only be drawn up as event-driven applications within the field of the graphical surface but also at the database level. This is one more way in which the application design is very close to the problem.

This makes ODABA^{NG} applications a favourite possibility to solve highly complex jobs as come up in administrative and knowledge areas.

Platforms

ODABA^{NG} supports windows platforms (Windows95/98/Me, Windows NT and Windows 2000) as well as UNIX platforms (Linux, Solaris).

You can build local applications or client server applications with a network of servers and clients.

Interfaces

ODABA^{NG} supports several technical interfaces:

- C++, COM as application program interface (this allows e.g. using ODABA^{NG} in VB scripts and applications)
- ODBC (for data exchange with relational databases)
- XML (as document interface as well as for data exchange)

User Interfaces

ODABA^{NG} provides special COM-Controls that easily allow building applications in Visual Basic. On the other hand ODABA^{NG} provides a special ODABA^{NG} GUI builder.

2 Remote Processing

Remote processing is a feature, which has been provided in order to submit jobs to be executed on another machine. Processes can be submitted by any application.

In order to execute submitted processes, one or more process queue manager can be started on one or more machines. A process manager may process requests from a named process queue or request from the global process queue. Processes submitted as local processes are sent to one of the named queues. Processes submitted al global processes are sent to the global queue, but the result is returned to the sending queue.

Tools

ODABA provides process queue manager (PQM) as command line utility and as GUI tool.

GUI QM

The GUI process queue manager displays the named queues and the queue status.

Console PQM

The console PQM can be started in order to process a named or the global process queue. The console PQM immediately starts processing the required queue.

Build support

The ODABA Development Environment (ODE) allows submitting several build requests to a remote server (PQM). This is typically the project server.

Processing

Processes from a queue are processed in FIFO order. In the GUI PQM you may change the position of a request in the queue by moving it up or down in the list.

When the queue is empty, the process waits until jobs are sent to the queue. The cycle time is one second, but it can be changed in the settings for the queue parameters. Default cycle time is 1 second.

3 GUI Process Queue Manager

The GUI process queue manager displays the named queues and the queue status. The GUI PQM can be called as follows:

ode90.exe .../pqm.Ini

Ini-file

The ini-file contains the definitions for the data sources. It must contain two different sections for the project and the database resources (RESSECT and DATSECT).

[SYSTEM]

DICTIONARY=C:\odaba\adk.sys
[ODE90]
RESOURCES=RESSECT
DATA=DATSECT
PROJECT=ProcessQueueManager
PROJECT_DLL=Designer
DESIGNER_RES=C:\odaba\res
DSC_Language=English

[RESSECT]

DICTIONARY=C:\odaba\adk.sys
DATABASE=C:\odaba\adk.dev
NET=YES
ONLINE_VERSION=YES
CTXI_DLL=AdkCtxi

[DATSECT]

DICTIONARY=...odaba/sample/sample.dev
DATABASE=...odaba/sample/sample.dat
NET=YES
ONLINE_VERSION=YES
ACCESS_MODE=Write

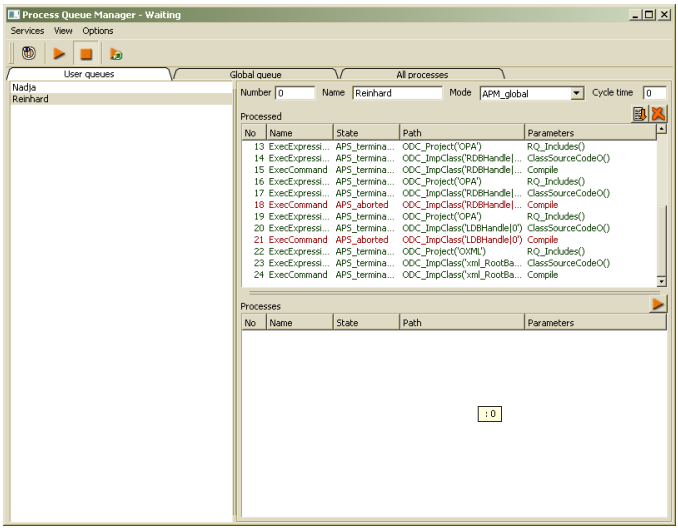
[SYSTEM] The system section refers to database system information. The minimum required is the DICTIONARY reference to the system dictionary, which is stored in the ODABA^{NG} installation folder. When running the application with a system dictionary stored on the server, server name and a port number have to be defined as well.

[ODE90] The ODE90 section contains information for the ODABA^{NG} GUI runtime environment. It refers to sections for resource database and database locations and contains some details fir the Index Services application. This section must not be changed.

[RESSECT] This section defines the connection to the application resource database, which is the adk.dev database provided on the ODABA^{NG} installation folder. This section must be updated, when ODABA^{NG} had been installed on a different location as the default location or when running the application in a Unix or Linux environment.

[DATSECT] This data section defines the connection to the application database by defining the dictionary and the database. When indexing a resource database (as in the example above), the dictionary is the system dictionary adk.sys provided in the ODABA^{NG} installation folder.

Usually, paths for dictionary and database must be replaced by the application database (DATABASE) and the application resource database (DICTIONARY).

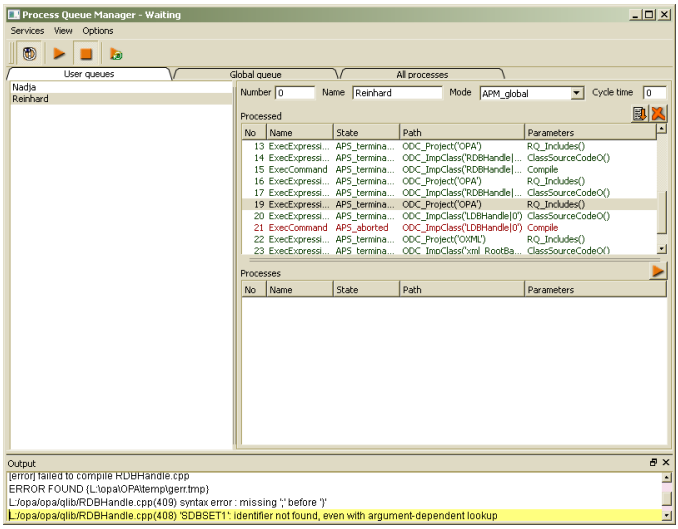


Execution for a queue can be started by clicking on the start button on top (running global queue) or pressing the start button for the local queue (above the process list for the local queue, on the right side)

The process queue manager can execute only one queue at the time. Before starting another queue, the running queue must be stopped.

The upper process list for the local queue contains the requests already processes. You may remove requests from the result queue by deleting those one by one or the whole collection by activating the context menu. Deleting all processes from the list, which have been exe-

cuted successfully, you may press the delete-OK button above the result list. Lines for processes failed are displayed with red colour.

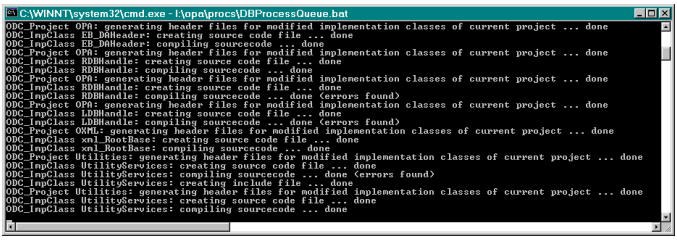


The GoTo button above the result list will remove the process from the result list and displays the process protocol in the output area.

4 Console Process Queue Manager

The console PQM can be started in order to process a named or the global process queue. The console PQM immediately starts processing the required queue. The console PQM can be called with following parameters:

DBProcessQueue *ini_file* [*queue_name*]



The console PQM runs until it is terminated by pressing the enter key. In order to complete the process currently processed, do not use ^c for terminate the console application.

ini_file

The configuration or ini-file contains the definitions for the data sources, object collections to be indexed and text fields.

[SYSTEM]

DICTIONARY=C:\odaba\adk.sys

[DBProcessQueue]

DICTIONARY=C:\odaba\adk.sys
DATABASE=...odaba/sample/sample.dev
NET=YES
ACCESS_MODE=Write

[SYSTEM]

The system section refers to database system information. The minimum required is the DICTIONARY reference to the system dictionary, which is stored in the ODABA^{NG} installation folder.

queue_name

A queue name can be passed in order to process a named (local) queue, only. The queue name is the name of the queue to be processed. In order to process the global queue, no queue name is passed.

5 Remote Build Processes in ODE

Build processes

The ODE supports remote processing for build jobs. In order to activate remote processing, the PROCESS_QUEUE option must be set under Options. This can be done in the ini-file or in the ODE settings.

Submitted jobs are stored in a named queue. The queue name is the name of the currently selected configuration (or user name). By default, each ODE application logs in with the system user name.

LOCAL queue

Setting the option to LOCAL allows processing a local process queue in a separate thread or on a remote process server.

In order to execute processes on a remote machine, a PQM must be started on the selected machine passing the queue name in the program call (console PQM) or activating the corresponding queue in the GUI PQM.

GLOBAL queue

Setting the option to GLOBAL allows processing requests in the global queue. The global queue collects requirements from all local queues, which submit global processes. Processing sequence is FIFO.

In order to execute processes on a remote machine, a PQM must be started on the selected machine without passing a queue name in the program call (console PQM) or activating the global queue in the GUI PQM.