



Remote Processing





run Software-Werkstatt GmbH Weigandufer 45 12059 Berlin

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

Berlin, October 2012

Content

1	Introduction	4
		4
	Platforms	4
	Interfaces	4
	User Interfaces	4
2	Remote Processing	5
	Tools	5
	Processing	5
3	GUI Process Queue Manager	6
	Ini-file	6
4	Console Process Queue Manager	9
-	ini file	9
	queue_name	9
5	Remote Build Processes in ODE 1	10
		10

1 Introduction

ODABA ^{NG}	ODABA ^{NG} is an object-oriented database system that al- lows storing <u>objects</u> and <u>methods</u> as well as <u>causalities</u> . 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 (Win- dows95/98/Me, Windows NT and Windows 2000) as well as UNIX platforms (Linux, Solaris).
	You can build local applications or client server applica- tions 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 data- bases)
	 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.

- ToolsODABA 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.lni

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 ODA-BA^{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).

🔜 Proce	ess Qu	ieue Ma	nager - V	Vaitir	ig															- 🗆 3	×
Services	View	Option	15																		
8			6																		
(Us	ser queu	s			G	lobal qu	eue		\sim		All p	rocesse	s							
Nadja							Numbe	er O	N	ame	Reinhard			Mode	APM do	bal		-	yde time	0	1
Reiniard	1														,			_		Faile	1
							Proces	sed		_		-				_					4
							No	Name		Stal	te	Pa	ith			Par	ameters				3
							13	ExecEx	pressi	APS	_termina	. 00	C_Proj	ect('OP#	4') DRHaadial	RQ.	_Include:	s() Codo	00		
							19	ExecCo	pressi mmand	APS	_termina	. UL	C Imp(lass(R Tlass(R	DBHandlej.	Car	onile	code	00		
							16	ExecEx	pressi	APS	_termina	. 00	C_Proj	ect('OP/	4)	RQ	Include	s()			
							17	ExecEx	pressi	APS	_termina.	. OE	C_Imp(lass('R	DBHandle	Cla	ssSource	Code	0()		
							18	ExecCo	mmand	APS	_aborted	00	C_Imp(Ilass('R	DBHandle).	Cor	npile	-0			1
							20	Exects	pressi pressi	APS	_termina bermina	. UL	C Imp("lass(") f	+))BHandlel0	עא רוא	_include: ssSource	s() Code	00		Ш
							21	ExecCo	mmand	APS	aborted	00	C_Imp(lass('L	BHandle 0) Cor	npile		-0		Ш
							22	ExecEx	pressi	APS	_termina	. OE	C_Proj	ect('OXI	4Ľ)	RQ	_Include:	s()			Ш
							23	ExecEx	pressi	APS	_termina	. 00	C_Imp(lass('xi	nl_RootBa.	Clas	ssSource	Code	00		Ш
							24	EXOLUL	minanu	APS	_ternina	. 01	vc_nub	-1855(XI	n_kootba.	cu	npue			•	1
							Proces	ses													1
							No	Name		Stal	te	Pa	ith			Par	ameters				Ĩ
															:0						-

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

Process Services \	Queue Manager - W /iew Options	aiting		_	-							- 🗆 י
	User queues	\mathcal{V}	Global que	ue	V	A	I processes)			
Nadja Reisbard			Number	0	Name Rei	inhard		Mode	APM_glob	al 💌	Cycle time	0
Keinaru			D									
			Process	ea	6		o					
			12.0	Name	State	croin a	Path ODC Droig	eb('ODA'		Parameters		
			14 8	ExecExpress	si APS_ter	mina	ODC_PIOJE ODC_ImpC	lass('RD)	/ BHandlel	ClassSourceCo	deO()	
			15 8	ExecComma	nd APS_ter	mina	ODC_ImpC	lass('RDI	BHandlej	Compile	0	
			16 8	ExecExpress	si APS_ter	rmina	ODC_Proje	ct('OPA')	RQ_Includes())	
			17 8	:xecExpress ExecExpress	si AP5_ter si APS her	mina	ODC_impc ODC_Proje	lass(RU chí'OPA'	BHanciej	ClassSourceuce RO_Includes()	odeO()	
			20.8	ExecExpress	si APS_ter	mina	ODC_ImpC	lass('LDE	/ 3Handle(0')	ClassSourceCo	odeO()	
			21 6	ExecComma	nd APS_ab	orted	ODC_ImpC	lass("LDE	(0) Handle	Compile		
			22 8	ExecExpress	si APS_ter	mina	ODC_Proje	ct('OXMI	.)	RQ_Includes()	1.00	
			1 25 0	EXECEXDITESS	si AP5 cer	mina	ODC IMPC	asst xm	Rootsa	Classbourcecu	odeuu	
			Process	es								
			No I	Name	State		Path			Parameters		
Dutput						_						8
error] faile	d to compile RDBHar	ndle.cpp										
ERROR FO	JUND (L:topatOPAtter	np\gerr.tmp}										
	And the ADD DI Langella server.	((0.0))		a bills af a sa	101							

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]

C:\WINNT\system32\cmd.exe - I:\opa\procs\DBProcessQueue.bat	- 🗆 🗡
ODC_Project OPA: generating header files for modified implementation classes of current project done	1
UDC_impClass LB_DHHeader: creating source code tile done ODC impClass EB DAHaader: compilion sourcecode done	
ODC_Project OPA; generating header files for modified implementation classes of current project done	
ODC_ImpClass RDBHandle: creating source code file done	_
ODC_Project OPA: generating header files for modified implementation classes of current project done	
ODC_ImpClass RDBHandle: creating source code file done	
ODC_IMPLIARS RUBHANDE: COMPILING Sourcecode and (errors found) ODC Project OPA: generating header files for modified implementation classes of current project done	
ODC_ImpClass LDBHandle: creating source code file done	
ODC_impClass LDBHandle: compiling sourcecode done (errors found) ODC_Project OXHL: generating headow files fow modified implementation classes of curvent project	
ODC_InpClass xnl_RootBase: creating source code file done	
ODC_impClass xnl_RootBase: compiling sourcecode done ODC_meinet Hilitian: generating baden files fan medified implementation slasses of support pusiest de	
ODC_Inglate difficults. generating header files for Modified Implementation classes of current project us	ille
ODC_ImpClass UtilityServices: compiling sourcecode done (errors found)	
UDC_impliass UtilityServices: creating include file done ODC Project Utilities: generating bedge files for modified implementation classes of current project do	ine
ODC_ImpClass UtilityServices: creating source code file done	
ODC_ImpClass UtilityServices: compiling sourcecode done	
	×

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 PRO-CESS_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.