Enhydra Shark 2.1-1 release notes

Table of Contents

System requirements and Upgrading

System requirements

JDK 1.4 or higher can be used for runtime (this does not apply to some client applications - they must be used with JDK 1.5 or higher).

What's new

This section lists the new or updated features associated with Enhydra Shark 2.1-1

What's new in Shark 2.1-1

These are the main changes for this release:

- DODS version 7.2-1 included
- EAF version 7.2-1 included
- Changed AdmiMiscExt API method getNextOptions() -> added another parameter to specify if asynchronous subflows should be skipped
- Configuration procedure improved to configure all distribution packages to use DB as configured in configure.properties file. If HSQL is used, web apps use local HSQL database.
- Changed SharkInterface API - getProperties now returns NameValue[] instead of java.util.Properties to enable its usage in a WebService scenario.
- Changed XPILHandler API so it can be used for WebService scenario (Node -> String, Properties->NameValue[])
- Introduced additional methods to ExecutionAdministration to be able to set Name, Description, Priority properties of Process/Activity and to set variable only in the local activity context and AdminMisc to be able to get activity result and to obtain process's activity requester. This was necessary to implement to be able to make SWC use only stateless interface.
- Extended ProcessFilterBuilder and ActivityFilterBuilder interface with methods for searching by the phrase contained in process/activity names, and by the priority less/greater than.
- ProcessFilterBuilder interface extended to support search processes by finish time
- ActivityFilterBuilder interface extended to support search processes by finish time, and to search activities by finish time
- Modified ActivityFilterBuilder interface's method addHasAssignmentForUser to accept additional parameter which specifies to retrieve activities only for accepted, only non-accepted or both type of assignments
- Some methods in ActivityFilterBuilder interface changed the name to be consistent with others
- Implemented method addProcessDescriptionContains in ActivityFilterBuilderDODS
- Added setExternalRequester method of WfRequesterInternal interface in order to ease implementation of custom process Id generation by having custum WfProcessMgrInternal implementation.
- Improved ProcessFilterBuilderDODS and ActivityFilterBuilderDODS methods for time before/after search, to take in account that if time stored as a long in a database has value Long.MAX_VALUE / 2, it should ignore it in a search
- Implemented new feature for FilterBuilders to perform case in-sensitive search. To specify if search should be case-insensitive, change newly introduced configuration parameter called *FilterBuilder.useUppercaseStringQueries* to have the value true.
- Implemented new method in CallbackUtilities interface to extract User Id out of the WMSessionHandle object - code refactored to always use this method when extracting UserId from WMSessionHandle.
- Removed methods to access Shark APIs from SharkInterfaceWrapper class in order to reduce complexity of the class and to point out its main purpose (to get handle to SharkInterface through POJO, EJB, WS). Client applications and shark wrappers ajusted accordingly.

- Changed SharkInterfaceWrapper to accept additional vendorSpecific parameter to put into WMSessionHandle's vendorSpecificData field if different than null. This should be used together with custom implementation of Shark's CallbackUtilities interface which extract user Id from WMSessionHandle information. Client applications and shark wrappers ajusted accordingly.

- StandardAssignmentManager improved:

  - to be able to properly handle activity performers given in a XPDL variable. In this case ParticipantMapping is not used but we go directly to UserGroup manager (if configured to use it) and initially assume that it is a group user and try to expand it. If we fail (or if UserGroup manager is not configured to be used), assignment is generated for the user specified by variable value.

  - in the case when there are no participant mappings, to check if there is a group with the same id as participant id, and if so to search for all the users from this group (if user group api exists).

  - to respect HUMAN participant type from XPDL when there are no participant mappings configured

  - not to add processRequesterId in a result list if there are no one to execute this activity (SharkKernel anyway does it if configured to do so)

- Added new utility methods to WMEntityUtilities class: for retrieving sub-entity, attribute of an entity or entity's attribute's value

- Improved deadline and limit checker implementation

- Few WfXML/ASAP classes changed not to directly use Shark.getInstance() but to ask for interfaces through SharkInterfaceWrapper. This enables deployment of WfXML not only on top of POJO.

- Changed logic of finished process deletion, both administrative through API and automatic through defining system property (or ext. attrib. in prof version). Now all synchronous sub-processes are also deleted.

- Introduced special "notation" for tool activities which participant is not System and that have MANUAL Start or Finish mode. This new information goes to event audit persistence layer so it is possible to perform more powerful queries on a different activity types.

- Changed JSPClientUtilities and SharkJSPClient.conf to be able to specify JNDI lookup name for UserTransaction (previously the same name as for transaction manager was used) and thus allowing more flexible deployment of JSPClient application.

- Extended WMFilter implementation to also hold information about the values of the properties used for filtering. WAPIImpl changed not to go to instance persistence neccessarily - only if it can't interpret filter expression.

- Public methods from kernel's SharkUtilities class moved to Misc and WMEntity utilities. Code adjusted according to this change.

- Kernel improved to make assignment re-evaluation more performant. This improvement is the most significant in the case old activity assignments are equal to the new ones.

- SwingAdmin improved to know how to present Calendar and GregorianCalendar data instance.

- Improved SwingAdmin termination/abortion/deletion of many processes. Now it does by 100 processes per transaction.

- SharkWebClient changes:

- new packaging for JBoss (standalone using shark through POJO, and within EAR - using shark through EJB)

- Adjusted to use newly defined XPILHandler API which now can be used via WebServices

- Switched to use only stateless APIs so it can work with Shark deployed as WebServices

- Support for converting ARIS process definitions into XPDL and importing them into system

- Now deployed with WfXML support

- Fixed bug in assignment re-evaluation, process termination, deletion - wrong filter settings

- Fixed bug in Application/Participant mapping - didn't work correctly when mapping Participants/ Application from Package level

- Fixed bug in "continuation" handling (it was not checked if next activity from the proces belongs to the logged user)

- Implementation-Version attribute included into MANIFEST of jar files

- Fixed bug in WAPIImpl's methods: abortProcessInstances, assignProcessInstancesAttribute, changeProcessInstancesState, terminateProcessInstances which appeared in the case null value is provided for WMFilter input parameter (wrong method was called on ProcessFilterBuilder interface - addProcessDefIdEquals() instead of addMgrNameEquals())

- Fixed bug in the core system: sometimes when transaction failed there were no resources in transaction cache which do have assignments from the processes also in transaction cache and those resources were not removed from global resource caches when they should. Bug is fixed by extending WfResourceInternal API and calling this new method for emptying assignments belonging to the processes with given Ids - it is called from SharkTxSynchronization.

- Fixed bug in XPDLBrowserImpl - browsing of Transition's entities within ActivitySet didn't work properly.

- Fixed bug in WAPIImple.reassignWorkitem() - if the WfResourceInternal for targetUser didn't exist, it was throwing exception, now it creates one.

- Fixed NPE in DefaultMailMessageHandler when used to receive mails

- Fixed bug in Shark cache initialization - if init cache parameters are set like: Cache.InitProcessCacheString=* Cache.InitResourceCacheString=* the caches were not initialized.

- Fixed bug in SchedulerToolAgent - it was looking up for TransactionManager and then was casting it to UserTransaction (which was workiing with JOTM but not with JBoss's transaction manager)

What's new in Shark 2.0-2

These are the main changes for this release:

- DODS version 7.1-2 included

- Complete WfXML functionality now provided (based on ASAP 0.9 version and adjusted wfxml.xsd)

- Introduced method in MiscUtilities for converting file to byte[] and now it is used from every place we need to convert XPDL file to byte[] in order to upload/update engine
- Fixed bug in WfXmlActivityBindingImpl.java - method getProperties
- Fixed bug in SharkWebClient that prevented proper displaying of entries in worklist, processlist and reassign-list when admin user want to see other people's entries

What's new in Shark 2.0-1

These are the main changes for this release:

- DODS version 7.1-1 included
- EAF version 7.1-1 included
- Added possibility to use different DBs for different persistence layers.
- Exposure of EJB wrappers for JOnAS as WEB Services
- New Wrapper that exposes shark API as "pure" AXIS web services provided as WAR file for Tomcat.
- Complete WfXML functionality now provided (based on ASAP 0.9 version and adjusted wfxml.xsd)
- DODS EventAudit and InstancePersistence model changes (SQL scripts changed accordingly)
- Kernel (WfActivityImpl and WfProcessImpl) improvement - order of initialization changed so callback can work from any event audit (processes are now put into transaction caches and activities are put inside process's caches at a right time)
- Improved/fixed implementation of WAPI.terminateProcessInstances and WAPI.abortProcessInstances - now we do not allow not to specify valid process definition id and we terminate only processes based on this definition
- Added two more JUnit tests
- Improved profiling logs.
- Enhanced Limit and Deadline checker implementation - you can specify the exact times when limit/deadline checking should happen.
- Enhanced DefaultMailMessageHandler:

  - now using "," instead of ";" as separator for attributes where multiple values are allowed (e.g. to,cc,bcc addresses)

  - added possibility to specify names of attached URL, File and Variable attachments, as well as names for to, cc and bcc addresses

  - added possibility to specify charset for to, cc, bcc names, subject and to the content of non-multipart emails without mime type defined
- Implemented SMIMEMailMessageHandler capable of sending cripted and signed mails.
- Enhanced functionallity of XPIL interface: possibility to request for Date types in a certain form (DateTime/Date/Time), possibility to specify if you want to include/omit specific variables by type or

Id when retrieving list of variables for process/activty; when value is null, returning "" for the value attribute

- New configuration parameter in the case shark caches are used to specify to cache closed processes or not (now by default closed processes are not cached - previously they were always cached)
- New configuration parameter for optimization when we are sure there is only one thread at a time dealing with one process instance
- Possible to configure shark with Enhydra's configuration object (so it is now possible to configure it directly from web.xml)
- Enhanced XSLT tool agent
- Added tool agents: ExecuteSqlTool and DigestPasswordTool
- Changed configuration in OracleConf.xml file which greatelly improves performance when using Oracle database (it is assumed new Oracle driver is used - otherwise it won't work)
- Improvements of SharkWebClient application:

  - moving XSLT logic to EAF post-processor

  - possibility to show XForm for Worklist and Processlist

  - WEBRowSet hanling moved to BasicXFormsFactory interface

  - improved profiling info

  - improved handling of XForms hidden fields (putting null values for corresponding variables)

  - optional Quartz initialization

  - possibility to define list of pages only accessible from localhost (e.g. ProcessMonitoring, ...)

  - ShowDocumentPO - enhanced functionality (can accept MIME type) plus improved security (can specify HTTPReferers to be accepted)

  - XPILProcessVariableHandlerPO - improved to be able to specify which variables to include/ommit

  - commit now happens before writting DOM
- Fixed bug that could cause problems when used Date and primitive array variables (they were not cloned when passing it from process to activity and from activity to toolagent ...).
- Fixed handling of primitive array variables.
- Fixed bug in AsapObserverBindingImpl.stateChanged().

What's new in Shark 2.0-beta8

These are the main changes for this release:

- DODS CVS version 20070125 included
- EJB wrappers now available for JBoss, JOnAS and Geronimo. Selection of EAR to be built is done by editing configuration.properties file - by default it is configured to build EAR for JBoss.

Geronimo's implementation works without client-side control over the user transaction (using dummy user transaction for Geronimo from the client side).

- Exposure of EJB wrappers for JBoss as WEB Services.
- SwingAdmin and SharkConsoleClient applications can work with Shark also through WEB Service interface (EJBs for JBoss exposed as WEB Services)
- Shark.conf separated into two files: SharkClient.conf and Shark.conf. One is used for client and the another for server side. If client type is POJO, server side file is referenced from the client side file. Client side configuration file is used by all shark client applications comming with Shark project, and there you can define if client application is accessing shark integrated as POJO, deployed as EJBs in some EJB container or through WEB Services as well as other parameters necessary to access shark.
- Shark WEB Client application improved
- Shark Console Client now can also work in "command" mode. Several basic commands are supported (package upload, process termination/deletion etc.)
- Client API heavily changed to remove duplicated method names and complex Java classes usage (because of Web Service implementations) - API implementations and client applications adjusted respectively
- ASAP & WfXML bug fixes and improvements (fixed problems with AXIS generation of Java classes, JUnit SharkWebServiceTestCase improved).
- Standard WfXML interoperability plug-in improved - it can accept new property called *Interoperability.IgnoreTerminateAndAbortRemoteExceptions*. When this property is set to "true", exceptions that could happen during termination or abortion of remote process will be ignored.
- XPIL (XML Process Instance Language) schema extended to support byte[] variable.
- XPIL API extended with several new methods.
- Internal API changes (Security API, UserGroup, ParticipantMapping, ...)
- introduced separate SQL scripts for MSQL2005 (because of XML type), and scripts for MSQL2000 and Oracle adjusted to support XML types for XMLs upon 4000 chars
- Fixed bug in WAPIImpl.getWorkitem() method - didn't return 100% correct info
- Fixed bug in PackageAdmin methods for uploading/updating package via byte[] parameter (the bug occured when there are external package dependencies).

What's new in Shark 2.0-beta7

These are the main changes for this release:

- DODS 7.0-5 included
- Updated libraries: axis1.4, jawe2.1, hsql1.8
- SwingAdmin community version added to project. Works with Shark embedded through POJO or deployed as EJB

- swing admin application now reads properties to determine if it will work with Shark embedded throuh POJO or with Shark's EJB service, as well as some other properties to specify which tabs will be visible, and which group of users can use which functionallity, ...

- swing admin application is improved regarding performance

- swing admin application has another section with the list of processes instantiated by particular user

- WEB Admin application became part of Shark project
- Added Shark Console Client application (works both with embedded shark and shark deployed as EJB) with basic functionallity to upload XPDL, instantiate/terminate/delete process, obtain worklist, change variables and complete activity; startup script for the application provided (runCC script)
- Added more examples for testing shark
- ASAP and WfXML services refactored, improved and put back into shark
- Stateless and Stateful session beans
- CORBA wrapper partially restored (only core OMG API + reduced SharkConnection API)
- Added new XSLT tool agent
- Added new Storage tool agent which utilizes DODS to store various data into DB
- Added utility module for working with XPDLBrowser API
- Added new SharkClientUtilities module with new utility class for shark handling, and new utilitiy class for special Limit checking (aborting processes which limit expired), and improved Deadline checking
- DODS implementation of UserGroup, Participant mapping and Application mapping plug-in API moved to community version
- New Simple Cache implementation defined (utilizing HashMap)
- New Shark client extension API for advanced engine handling (no implementation in community version)
- Added draft of new API with several methods for obtaining XPIL (XML Process Instance Language) representation of the process instance, activity instance, worklist, ...

  It is used from Shark WEB Admin application.

- SharkInterface API:

  - new method to obtain plug-in component implementation

  - new methods to obtain new extension APIs

  - new method to obtain new XPIL API

  - all methods are now throwing Exception

- New Admin application API for User Group/Participant Mapping/Application Mapping/Repository Handler for client applications (this is not engine API)

- Added Default Implementation of Admin API:

  - DODS User Group implementation

  - DODS Participant Mapping implementation

  - DODS Application Mapping implementation

  - default implementation of Repository manager

- WAPI extended with method to obtain single WMProcessDefinition

- WMProcessDefinition, WMProcessInstance, WMActivityInstance, WMWorkItem structures extended to handle description property.

- WMProcessInstance structure extended to handle factory name property (name of corresponding WMProcessDefinition)

- Some of XPDLs examples are updated; Workflow Patterns XPDL got two new patterns 'Discriminator' and 'N-out-of-M Join' (thanks to Fuad Abinader)

- Added new WfMC API for handling event audits (taken from OBE and adjusted)

- AdminMisc API:

  - new methods for handling event audits (based on WfMC spec and interface taken from OBE); NOTE: still draft version

  - removed metods getProcessContext and getActivityContext

  - signatures of methods that were returning java.util.Map are changed to return String[][]

  - introduced new method to retrieve all usernames

- PackageAdministration API:

  - introduced new method to get WMEntity representation for the Package (XPDL) specified by Id and Version

  - signature of several methods changed to return WMEntity of the Package (i.e. when uploading, updating, closing package)

- UserGroup plug-in API:

  - introduced new method to get all groups for the specified user

  - new metod to get password for the user

  - new method to validate user

- Assignment plug-in API got new methods to get UserGroup and Participant mapping plug-in implementation

- ToolAgentManager plug-in API got new method to get Application mapping plug-in implementation

- WMEntity: introduced equals method

- RootException removed from signature of all internal APIs

- Improvements and bug fixes in WAPIImpl
- Changes in WAPIImpl:

  - methods for obtaining definition, process, activity, workitem are now returning null if there is no such entity, instead of throwing exception

  - assignActivityAttribute and assignWorkitemAttribute are now setting (OMG's defined) result of the activity to include provided variables
- Added possibility to use filters for basic filtering of the result obtained through XPDLBrowserImpl of corresponding interface
- Exception handling improved to reduce exception wrapping at different layers which resulted in un-readable stack trace
- SMTP Event Audit implementation refactored and put back into shark
- Mail Tool Agent's DefaultMailMessageHandler implementation changed. Now it can accept many different parameters, and it recognize parameters provided from shark through the Ids of FormalParameters of corresponding XPDL Application definition
- LRU caches changed - now size -1 means unlimited cache
- Fixed bug in WMProcessInstanceState - state "not started" was defined as "suspended"
- Fixed bug in WMWorkItemState - wrong values for states
- Fixed default kernel implementation of activity (WfActivityImpl) to properly work with WfXML
- Fixed bug in WfActivityImpl for creating assignments when XPDL participant is expression

What's new in Shark 2.0-beta5

These are the main changes for this release:

- DODS 7.0-4 included
- Improved version of WEB WorklistHandler application WAR included.
- Fixed bug in AssignmentFilterBuilderDODS in method addPackageIdEquals().
- Fixed bug in ProcessFilterBuilderDODS in method setOrderByResourceRequesterId()

What's new in Shark 2.0-beta4

These are the main changes for this release:

- DODS 7.0-3 included
- UserGroup interface extended with additional methods for querying Users/Groups based on additional criteria.

- LDAPUserGroup implementation now supports ActiveDirectory (structure type 2 in configuration)
- Many improvements on worklist handler based on XPIL schema (work in progress)
- Improved utility for Deadline checking (passing lookup name parameter for obtaining User transactions)
- Improved DefaultMailMessage handler: now it is able to send e-mails with attachments. Also, additional parameter added to define wheter to authenticate when sending e-mails or not.
- Fixed bugs in methods for building queries for actiivty variable search in ActivityFilterBuilder DODS implementation.
- Fixed bug in ExecutionAdmin.checkDeadline(WMSessionHandle,WMFilter)

What's new in Shark 2.0-beta3

These are the main changes for this release:

- implemented possibility to have multiple Event Audit plug-ins
- DODS 7.0-2 included
- Instance persistence and event audit persistence information carriers re-defined as the final classes.
- Implemented simple support for XPDL Schema type (represent it as w3c object inside shark) - instance persistence and event audit data models changed accordingly

  Note
  not performing validation

- Corrected shark's default behavior regarding usage of external packages (instantiation of sub-flows, and usage of application definitions):
  - If the sub-flow/application definition is from the same package, instantiating/using the same version as the version of the main process
  - if the sub-flow/application is from an external package, instantiating/using the last updated version which XPDL version is equal or less than the main process's XPDL version

    Note
    validation is still not implemented

- worklist handler based on XPIL schema (work in progress)
- improved InitialValue handling for Date and custom Java class instances (using script interpreter)
- extended FilterBuilders to support specifying the start for querying (in combination with limits will be used for pagination)

What's new in Shark 2.0-beta2

These are the main changes for this release:

- Simple JSP shark client example included.
- DODS 7.0-1 included
- Time profiling now possible for DODS instance persistence layer.
- New API method in ExecutionAdministration for "exception injection" - client is now able to inject exception to shark's activity, and after its completion shark will try to follow exception transition (attempt to re-implement SchedulerToolAgent to use this new feature).
- Improved ActivityFilterBuilder and AssignmentFilterBuilder APIs - added ommited methods that were contained in the last CVS version of old shark 1.1-x.
- Extended internal plug-in APIs: Instance Persistence, UserGroup, Callback and Logging
- Fixed: SQL scripts for creating Postgres DB - now it works with Postgres8.1.
- Fixed: NPE bug in HistoryRelatedAssignmentManager.
- Fixed: bug in InstancePersistenceManager regarding the result of activity.

What's new in Shark 2.0-beta1

This is a major release, with several very crucial changes in the concept:

- Full support for JTA. Shark doesn't handle transactions anymore, it's up to the client application to control the transaction.
- new DODS 7.0-beta1with full JTA support included.
- shark project now contains only pure engine implementation (no client applications, CORBA wrapper, EJB wrapper, Web Service wrapper, ...).
- Data model slightly changed; improved model for variable persistence in order to be able to persist arrays in a natural way.
- Non OMG client API greatelly changed.
- Removed client API for XPDL repository management (it's not shark's job to handle XPDL repository).
- Removed client API for UserGroup handling (it's not shark's job to handle users and groups)
- Removed client API for Participant Mapping (it's not shark's job to handle participant mapping)
- Removed client API for Application Mapping (it's not shark's job to handle application mapping)
- Added client API for XPDL browsing - now you can obtain any information about XPDL deployed on shark.
- Added client API that greatlly conforms to WfMC spec for interface 2 in the sence of stateless methods (it is mostly taken from OBE)
- Expression builder interface greatelly changed, enhanced and renamed to Filter builder. It is possible to do ORDER BY on many columns, and to set database limits.

- Shark switched to optimistic locking of process instances. It is now possible that more than one thread access the same process instance at a time.
- Modul for handling XPDL removed from shark, it is now included as a jar file from JaWE project.
- Internal APIs greatelly changed
- Removed following internal APIs: Authentication, Limit, ProcessLocking, Script Map persistence, Transaction and User transaction
- Participant mapping and UserGroup internal APIs shorten, and used only from Assignment API.
- Application mapping internal API shorten, and used only from Tool agent factory API
- Introduced time profiler for measuring time spent inside shark methods (useful for finding bottlenecks).
- Improved handling of variables: shark now also accepts Integer and Short for Long variable, Float, Long, Integer and Short for Double variable, and java.sql.Date, java.sql.Timestamp and java.util.Calendar for java.util.Date variable
- "automatic start"/"manual finish" mode combination allowed for Tool activities with non-system participant performer
- New configuration parameters for not creating default assignment, for allowing to set the priority out of the OMG spec specified range, for determining to store arrays as BLOBs or in an enhanced variable model
- Optimized and improved Swing admin application (UserGroup, XPDL Repository, Participant and Application mapping are now Swing admin APIs, included new JaWE version, no more memory leak, ability to monitor processes based on all XPDL versions ...)
- ToolAgent loader and Scheduler tool agent are enhanced
- Improved XPDL model and validation
- MySQL data model fixed, and switched to innodb
- Fixed: bug in SharkUtilities -> Shark's 'transaction' cache wasn't filled when user required a process that was in LRU cache.
- Fixed: participant mapping now can be used when shark is configured to work without user/group implementation.
- Fixed: bug related to assignment deletion when shark is configured to delete other assignments when some user accepts one.
- Fixed: hashCode methods introduced to WfXXXWrapper classes.
- Fixed: several bugs in DODS's Expression Builders (now DODS's Filter Builders).

What's new in Shark 1.1-2

This is mainly the bug fix release, although it contains some new features:

- The great contribution by David Forslund: CORBA server for Shark that is based on POA.

You can choose which server you want to start, an old one - using old scripts (run or runAll to also start admin), or POA, using runPOA or runAllPOA scripts

The admin client is also modified to be able to use both CORBA server implementations.

It is strongly adviced to set this option to "true", and if you are using connection for a long time (not calling disconnect method()), use doneWith() method to release memory for the CORBA objects you don't need any more.

- CORBA admin client is modified to be able to use both CORBA server implementations.
- Shark can be configured to work with data sources.
- new DODS 6.5, with bug fixes, and with support for data sources is included
- Fixed: bug in DODS regarding concurrent modification exception.
- Fixed: bug in SecurityManager - it was reading wrong configuration parameter (Shark.conf file also fixed)
- Fixed: DODS EventAudit and InstancePersistence managers fixed to support persisting of null variables when using Oracle DB
- Fixed: WfActivityIterator and DODSPersistence manager to properly retrieve activities in a certain state when using WfProcess.get_activities_in_state() method.
- Fixed: BaseIteratorWrapper not to throw null pointer exceptions when there is no expression set.

What's new in Shark 1.1

This is a major release, with a lot of new features, performance improvements, bug fixes, ...:

- Shark implements ASAP interface, and part of WfMC's Interface 4 (Wf-XML) for managing XPDLs. It can be deployed as a WEB service, and one can communicate with shark through WfXML and ASAP protocol for engine interoperability. You can use JaWE 1.4.2 version to connect to shark engine deployed as Wf-XML WEB service, get the list of Process definitions, download XPDL, upload new XPDL or update existing, and all of that using interface 4 defined by WfMC.

  Also, new internal API is defined for executing "remote" subflow processes through ASAP interface. Now it's possible to make a reference to a "remote" subflow in XPDL. The definition of such subflow process does not exist in XPDL, and user need only to enter ActualParameters - see new shark_retailer.xpdl example which was used in Wf-XML demo in Miami, where shark communicated through Wf-XML interface with Fujitsu and TIBCO/Staffware engines. There are three new parameters in Shark.conf for setting up the usage of this new API.
- Defined Security API for checking if user can perform certain operations on working instances (processes, activities, ...). Kernel (if it is configured to use Security API) always asks security manager if user is allowed to perform operation. Also, simple implementation of this API is implemented.
- AdminMisc interface (both POJO and CORBA) extended with may new methods, like the ones for getting extended attributes for various XPDL entities, retrieving activity instance Id and resource username out of assignment Id (it can be used from ToolAgent API which gets assignment Id as a parameter), getting the time of process/activity creation, start(accept) and finish, ...
- It has support for HSQL 1.7.2 which is now included in distribution.

- Introduced storing of External process Requester class name into DB. External requesters should be stateless.
- External requester (if set for some process instance) will be notified on any process event audit (before, it was notified only about state event audit).
- Changed package names for CORBA API (our CORBA admin application and other CORBA client applications changed accordingly)
- MANUAL finish mode for tool activities is now allowed. If specified, the activity won't be completed after executing tool agent, but it's up to tool agent implementation to finish this activity. The contributed Scheduler tool agent does exactly this.
- Major improvements on the performance.
- Separate logger for Scripting
- Implemented caching of LDAP user/group attributes in LDAP implementation of UserGroup API
- Implemented possibility to use custom Java classes as process variables through CORBA API (of course, these classes must be serializable)
- Handling of Date and HashMap variables through Admin applications
- Allowed importing and executing of recursive process definitions (subflow activity instantiates the process of the same process definiton as its own process)
- DB structure for instance persistence changed a lot. New redundant fields are introduced to gain more performance.
- DB structure for storing process and activity variables changed. Now, not all variables are stored into LONGVARBINARY field, but Date variable is stored into TIMESTAMP field, Boolean into BIT field, Long into BIGINT field, Double into DOUBLE field, String into VARCHAR field (if it does not contain too many characters, otherwise it goes into LONGVARBINARY), and custom Java classes are stored into LONGVARBINARY field.
- Provided configuration that defined the maximum number of String variable characters that will be persisted into VARCHAR field (the limit is 4000). If String variable has less characters than the maximum specified, it will be stored into VARCHAR field, and otherwise it'll be stored into LONGVARBINARY field.
- Introduced optional data model for Process and Activity variables which is more optimized for the use cases where there are not so many large String variables and custom Java classes variables. There is a configuration switch to tell DODS instance persistence layer which model to use. The usage of optional model is especially useful with Oracle DB.
- New implementation of event audit manager for notifying users about task arrival via mail is provided. It is called SMTP event audit manager.
- New implementation of event audit manager for notifying "listeners" about activity changed is provided.
- ToolAgentLoader class is provided, that enables you to add new Tool agents into shark system in runtime. The location where they need to be put is configurable - there's a new property called "ToolAgentPluginDir".
- Defined new SchedulerToolAgent, which is a proxy for calling other tool agents. It opens a new Thread for their execution, and it uses XPDL design with Automatic activity having Automatic start and Manual

finish mode. It is a general solution for long-running automatic activities. The XPDL example for its usage is also provided: test-Scheduler.xpdl.

- DeadlineAdministration (both, POJO and CORBA) interface was extended with new methods for checking deadlines, that are much more optimized. Also, new interface for retrieving deadline info for some activity is introduced.
- New configuration parameter introduced to define if Asynchronous deadline will happen only once, or every time when deadline checker determines (based on deadline condition) that Asynchronous deadline should be triggered.
- New method called "checkDeadlinesWithTermination" is introduced to client DeadlineAdministration interface for special cases when process needs to be terminated if the deadline happens. This method is especially efficient when deadlines are stored in DB (not re-evaluated). Also, the performance for checking all deadlines is optimized when deadline information is stored in DB (deadlines are not re-evaluated).
- ExecutionAdministration interface (both, POJO and CORBA) was extended with new methods for deleting process instances, which are now more optimized, with a methods for getting Input signature of the process (only IN and INOUT formal parameters of the process definition), and with methods for retrieving single variable, or a list of process/activity variables.

  Also, the method for deleting single process instance is improved - when you delete single process, it also deletes all of the nested synchronous sub-processes.
- PackageAdministration interface was extended with new methods for retrieving the content of any version of some XPDL.
- Instance persistence API is greatly changed in order to have shark much more performant and reliable. Also, indexes are improved.
- Shark transaction caches are introduced. This gain a lot of performance to shark, especially when client application controls creation/release/commit of the transaction, and it also makes possible to have a callback from shark components to shark even when shark LRU caches are not used.
- BeanShellToolAgent and JavaScriptToolAgent now have additional context variables: "procInstId" (Id of the process that called tool agent), "assId" (Id of assignment for which tool agent is called - it is consisted of username and activity Id, and you can use AdminMisc to extract username and activity Id out of assignment Id) and "sharkTransaction" (shark's transaction) which opens a new perspective of their usage (callbacks to Shark) - be careful not to define the XPDL variables having Ids as these new context variables.
- Encryption of the password stored into UserGroup model is implemented.
- New utility in SharkTests for uploading XPDLs is provided.
- SwingAdmin's patched in order to be used in multihead environment. Also, they are now using newly defined XXXIteratorExpressionBuilder interfaces to achive better performance and more optimized methods for process deletion and deadline checking. Translation to Spanish for Swing Administrator application(s) is provided. The Map and Data Java type process variables now can be displayed in SwingAdmin(s).
- New configuration parameter "Deadlines.useProcessContext" introduced. It is used to define if you want to use process or activity context when deadline expression is evaluated.
- Included finalize() method in ExecutionAdmin and SharkConnectionImpl to avoid minor memory leak in the case of improper usage of these classes (if user does not call disconnect() after he finishes using it)

- Improved logic for automatic deletion of processes (if user configures shark to delete processes immediatelly after they are finished). Now, if process finishes, and it's state is aborted or terminated, it won't be automatically deleted.

- Defined new Common API module, with ExpressionBuilder API. SharkConstants class moved from kernel to this module.

- Defined API (both, POJO and CORBA), and DODS implementation for ExpressionBuilders, which can be found in SharkUtilities/ExpressionBuilders. These builders enables you to "writte" query expressions for WfXXXIterators which will be executed directly on DB. The implementation of WfXXXIterators improved to support fast querying on DB, if expression for the WfXXXIterator is made using our ExpressionBuilder implementation, and instance persistence layer changed to intoduce new methods for this purpose.

- ExecutionAdministration API (both POJO and CORBA) now have additional methods for retrieving WfActivity, WfProcess and WfAssignment Iterators, which now can be used to make queries over all entries in DB (previously, you could only make queries for the instances belonging to the object which had a method to retrieve specific Iterator - i.e., you were only able to use WfActivityIterator in the context of some specific process, and now you can iterate on all activities in DB).

- Kernel API and implementation changed to perform more efficient activity acceptance/rejecting of assignments.

- Optimized performance for retrieving assignments through WfResource.get_sequence_work_item() method. There was an logical error in the code, that caused that this method always retrieves assignments from DB, instead of getting it from cache (if caches are used). Also, persistence layer is further optimized to make this retrieval even faster.

- The way how the kernel (by default) handles assignments changed: when one user accepts an assignment, the assignments are immediately removed from DB (not only hidden for that user), and if he rejects assignment, the assignments are again re-evaluated. Old approach can also be used if you configure kernel properly (there is a property called "SharkKernel.deleteOtherAssignments").

- User, can now reassign its accepted assignment to another user, no matter if such assignment already existed before. Suspended activities can't be accepted/rejected.

- New configuration property for assignments introduced. It determines if kernel will create assignments (default is true) or not.

  There are situations when assignment creation is not necessary, and this is the case when you always execute activities directly using change_state() WfActivity method.

  The new property is called SharkKernel.createAssignments

- All kernel implementations of client interface "managers" now have protected constructor (AdminMiscImpl, DeadlineAdmin, ...), so one can easily provide it's own implementation extending default one.

- CORBA API changed:

  - removed Administration API, and all the methods from this API are moved to SharkInterface API

  - added doneWith() method on SharkInterface, SharkConnection and ExecutionAdministration interfaces (it is used to release server memory for given CORBA object))

- CORBA memory leak fix: there is a new configuration parameter for CORBA server called "CORBAServer.TrackAndDisconnect". Default server value is false. If set to true, whenever

you call disconnect() on some interface (PackageAdministration, SharkConnection, AdminMisc, ExecutionAdministration, ...), all CORBA objects created using this connection are released (disconnected from ORB).

It is strongly adviced to set this option to "true", and if you are using connection for a long time (not calling disconnect method()), use doneWith() method to release memory for the CORBA objects you don't need any more.

- Major refactoring of XPDL code in order to achieve faster upload of XPDLs and better engine startup times (startup times are now better up to 5 times, depending on XPDL). All classes changed to implement new logic, and now there is a nicer API for retrieving appropriate XPDL information.

  Kernel classes greatly chaned due to the previous changes.

- Change to RepositoryPersistence API and implementations in order to achieve faster upload of XPDLs and better engine startup times.

- Changes to AssignmentManager and LimitAgent APIs - there is no more parameter that represents ExtendedAttributes string, all relevant information can be retrieved through callback to shark if needed. Implementations are changed accordingly.

- New option for persisting (or not) OldEventAuditData. Default is to persist, and shark can be configured not to persist it - some performance can be gain if not persisting it.

- Configuration.properties file (for switching to another DB) is changed, and now it is much easier to switch amongst DBs - each vendor now has it's own username and password properties.

  Also, you can choose if shark will be configured to use caches (both DODS and Shark) or not. (Of course, this is useful only if you are configuring shark through Shark.conf file).

- New feature for shark configuration: XPDL spec does not say that OTHERWISE transition should be executed only if no other transition condition is evaluated to true (in the case of XOR split). So, if you i.e. put OTHERWISE transition to be the first outgoing transition of some activity, other transition's condition won't be even considered. You can configure shark to deviate from the spec, so that OTHERWISE transition is evaluated and executed only if no other transition condition is evaluated to true. To do that, you should set the property "SharkKernel.handleOtherwiseTransitionLast" to true.

- New feature for remote subflow invocation (through ASAP): if you put some variable Id as subflow URL, it'll be evaluated in runtime.

- Shark instance persistence layer and kernel prepared for storing LimitTime information. In the next release (after 1.1) we'll have some improvements regarding Limits.

- new DODS 6.4 included

- .bat and .sh scripts changed to allow 128M heap for Java VM

- Fixed: shark didn't persist initial state EventAudit for activity

- Fixed: shark didn't persist assignment EventAudit in the case of using activity's change_state() method

- Fixed: bug in WfProcess.result() method - in the case when there were a variable with the name that begins the same as resulting OUT formal parameter, it was retrieved instead of wanted variable

- Fixed: bug that appeared in fully automatic processes with AND split and XOR join or implicit AND Join

- Fixed: WfProcessMgr.category() method was returning language specific value of category (not the one specified by XPDL schema)
- Fixed: WfProcessMgr.context_signiture() now returns all variables from process definition (not only ones declared as FormalParameters) - as it is defined in OMG spec.
- Fixed: bug in AbstractToolAgent - the cache that was used for performance was not synchronized, and sometimes it caused unnecessary exceptions in multithread environment
- Fixed: bug in the case of asynchronous subflows when parent process is deleted before subflow finishes
- Fixed: bug regarding deletion of parent process of asynchronous subflow
- Fixed: bugs in InstancePersistence layer that rarely occured when using shark methods with SharkTransaction parameter in a row
- Fixed: bug fix that occured when updating "deeply" nested external package
- Fixed: when there was an AND split, and than XOR join on some ending activity of the process, shark didn't interpret this situation well
- Fixed: generation of the script for creating MySQL's instance persistence tables - RThis is mainly the bug fix release, although it contains some new features:The great contribution by David Forslund: CORBA server for Shark that is based on POA.You can choose which server you want to start, an old one - using old scripts (run or runAll to also start admin), or POA, using runPOA or runAllPOA scriptsThe admin client is also modified to be able to use both CORBA server implementations.It is strongly adviced to set this option to "true", and if you are using connection for a long time (not calling disconnect method()), use doneWith() method to release memory for the CORBA objects you don't need any more.CORBA admin client is modified to be able to use both CORBA server implementations.Shark can be configured to work with data sources.new DODS 6.4-2, with bug fixes, and with support for data sources is includedFixed: bug in DODS regarding concurent modification exception.Fixed: bug in SecurityManager - it was reading wrong configuration parameter (Shark.conf file also fixed)Fixed: DODS EventAudit and InstancePersistence managers fixed to support persisting of null variables when using Oracle DBFixed: WfActivityIterator and DODSPersistence manager to properly retrieve activities in a certain state when using WfProcess.get_activities_in_state() method.Fixed: BaseIteratorWrapper not to throw null pointer exceptions when there is no expression setEAL type was used instead of BIGINT type to represent long values
- Fixed: bugs related to process deletion: deleteClosedProcesses method(s) in ExecutionAdmin, DODSPersistenceManager, ...
- Fixed: bug with deadlines on block activity (when using caches, it didn't always work properly) and bug with retrieving deadline information from DB
- Fixed: DODS templates for Oracle - it was not possible to update large BLOB values in DB
- Fixed: bug in DODSActivity class (activity's accepted and activated time were not correctly persisted - they were switched)
- Fixed: bug in PackageAdmin - sometimes you could get an exception if updating XPDL that references other XPDLs
- Fixed: XPDLStraightParticipantMapping assignment manager, so that when the performer expression is an empty string, it does not create an assignment for a user with username being empty string, but rather for the user who created the process.

What's new in Shark 1.0

- The possibility to have deadlocks when used from multiple threads/VMs is minimized (some kind of ordering of DB writes is implemented).
- Added new functionality of handling exception transitions as defined by WfMC specification.
- Added new functionality of handling Deadlines. Shark now has defined client API, and its implementation for handling Activity deadlines. This API is supposed to be used by shark client to periodically ask shark to check deadines. Shark can be setup to re-evaluate deadlines every time deadline check is performed, or to initially calculate deadline times and store it into DB, and when asked to check deadlines, deadline limit is retrieved from DB. Shark comes with an example XPDL processes contained in deadlineexamples.xpdl file, that shows ASYNC and SYNC deadline handling.

  In shark deadline expressions along with all process variables, you can use special variables called:

  1. PROCESS_STARTED_TIME - the time when the process is started

  2. ACTIVITY_ACTIVATED_TIME - the time when process flow comes to activity and assignments for the activity are created

  3. ACTIVITY_ACCEPTED_TIME - the time when the first assignment for the activity is accepted

  NOTE: If activity is being rejected after its acceptance, or it is not accepted at all, the ACTIVITY_ACCEPTED_TIME is set to some maximum value in the future

  IMPORTANT:

  - There shouldn't be process variables (DataField or FormalParameter entities from XPDL) that have the same Id as the one of previously listed - The Java type of these variables is java.util.Date.

  - deadline expression result must be java.util.Date

  - if shark is setup to not re-evaluate deadlines, but to initially evaluate deadline limit times, ACTIVITY_ACCEPTED_TIME should not be used in expressions because it will contain some maximum time in the future.

  When starting Shark CORBA server, it can be configured if it will open a thread for checking Deadlines.
- Defined new EventAudit API - InstancePersistenceAPI is splited, and its part for handling Event audits is put into EventAudit API . Default implementation of this API is used only to store EventAudit information into DB, but one can give implementation which would i.e. also send email notifications to the users that are getting assignments. You can configure shark not to use this API implementation (by commenting entry EventAuditManagerClassName in Shark.conf file if you initialize shark this way) , in which case event won't be written in and they also won't be created in memory.
- Defined new RepositoryPersistence API. It is used to store the information on xpdl packages loaded into shark. Previously we didn't have separate API for that, and it was called internal repository, which was implemented as a folder where we were putting loaded xpdl definitions. Now, we have two different implementations of this API: FileSystem and DODS implementation, and they take care of XPDL versioning -> it is possible to load several XPDLs with the same Id (this way we implemented update of packages). If you configure shark to use FileSystem implementation, it works more or less like it was before (except that we have xpdl versioning). If you configure shark to use DODS implementation, xpdls will be stored into DB. This way, it is easy to configure several VMs that use shark on the same DB (you do not need to map disk drive like in the case of FileSystem implementation). By default (in standard Shark.conf), we use DODS implementation.

- Shark again has the posibility to update Packages. You can update some xpdl definition already loaded into shark with a definition that has the same package Id. So far, this package update concerns only newly created processes, and the old ones are running based on its own (old) xpdl definition. After updating package, It is possible to instantiate processes based on new and old definitions. You can try package update by using Shark's Admin application (you can previously upload the package, with the same Id as the one you want to update, into external repository, which can also be done by the use of admin application).

- Defined and implemented new API methods in PackageAdmin for synchronizing, refreshing and clearing shark's XPDL cache. Now, shark is more suitable for the use from many VMs at a time. If one VM opens new package, or updates an existing one, call to synchronizeXPDLCache method of other VMs synchronizes in-memory cached XPDLs with the state of XPDL internal Repository

- Shark's XPDL handling code (inherited from JaWE) is cleared from Swing-related stuff, plus it is cleared from some other unnecessary stuff, and optimized to achive better shark performance.

- Major shark's code optimization has been done, and this version is much faster that beta2 version, especially regarding execution of ToolAgents.

- UserGroup API (client and internal one) is enhanced with new methods.

- LDAP implementation of internal UserGroup API is changed. By the usage of this implementation, shark can access two different kinds of LDAP structures.

- Three new internal Mapping APIs with their own Transaction API (available to the client) are defined - MapPersistenceAPI is splitted into three APIs. Also, now we have nice relational model for storing mapping information for Participant and Application mappings. Now it is possible to hold Participant mapping information in one DB, and Application mapping information in another one.

- Changes to client interface used for mappings (CORBA interface didn't change). Now there are special transactions for each type of mappings (regarding to issue declared above).

- The problem with building Shark on Linux by the use of IBM's Java1.4.1 is solved.

- Now, the process can be instantiated using WfProcessMgr.create_process() with WfRequester parameter set to null.

- The use of nested XPDL TypeDeclaration is allowed (i.e. first TypeDeclaration declares type1 to be a BasicType-String, and second TypeDeclaration declares type2 to be of type1).

- Modified Assignment API to be more usefull.

- Included new HistoryRelated implementation of Assignment API - great contribution by Rich Robinson. You can use it by commenting standard AssignmentManager and uncommenting HistoryRelated assignment manager entries in Shark.conf (if you are configuring shark this way), and test it with Publish Document proces from test-JavaScript.xpdl.

- Included new XPDLStraightParticipantMapping implementation of Assignment API, that can be used if one wants to create assignments for users which have the same Ids as participants Ids in XPDL. Also, if you configure shark to work with this new implementation of Assignment API, you can create XPDLs where activity's performer will depend on some variable, and when starting process, you can set this variable value, and user with Id equal to this value will get an assignment.

- The oracle problem with cursors and BLOBS is solved (DODS runtime and templates are modified, new oracle.jar should be included in classpath when using Oracle DB)

- Added possibility to start shark as CORBA WINDOWS NT SERVICE, and as *nix daemon.
- Instance persistence interface is modified to fix a possible bug that could occur if there is some activity definition inside block activity that has the same Id as the activity defined directly in the process definition.
- New MailToolAgent is included:

  - in Shark.conf file you have some settings for MailToolAgent (in fact for the DefaultMailMessageHandler)

  - when calling this tool agent, AppMode 0 means sending and 1 means receiving mail

  - when calling this tool agent, AppName is the name of MailMessageHandler class to use. We have default implementation

  - DefaultMailMessageHandler, which needs 3 String IN parameters (destination address, mail subject and mail content) when sending mails, and 1 OUT or INOUT type parameter (message subject) when receiving mails. The example of usage for this tool agent is in test-JavaScript.xpdl -> Mail Handling process.
- New SOAP Tool agent is included, and example of its use is given in test-JavaScript.xpdl->Do math process-> "arbitraryop" activity. If using Admin, in previous "Enter various parameters" activity, you can select which mathematical operation offered by the web service you want to perform, and the possible choices are: Add, Subtract, Multiply, Divide (be careful, it is a typo in WSDL so you have to enter "Subtract" not "Substract" as expected).
- In AdminMisc interface (both POJO and CORBA), methods for getting extended attributes of following XPDL entities: Package, WorkflowProcess, DataField, Participant, Application and Activity are added
- New methods for administrative deletion of finished processes are added in ExecutionAdministration interface. Now, you can delete finished processes and after that unload their package (if there are no running process instances or other packages depending on this package).
- New methods for getting all package versions, and for closing particular version is added in PackageAdministration (also, old method for closing package by Id tries to close all package versions).
- The shark is logging which managers are instantiated, and it also displays a problem if initialization fails because of some manager class is not specified, or manager didn't configure properly.
- The RootError class is introduced, and DefaultToolAgent (and some other classes) do not throw RuntimeException anymore, but RootError with the cause.
- Updated Architecture picture and documentation.
- Removed support for QED and InstantDB databases (they couldn't really be used in shark because of some types that they don't support)
- Introduced Client LimitAPI (both POJO and CORBA) for polling for activity/process limits (similar as client Deadline API).
- Standard internal Limit API implementation is created - it doesn't create new Threads like Timer based implementation. Currently, it does nothing special but just logs to the console when time Limit has passed.
- The usage of "user.dir" property is completely removed from shark.

- Introduced connect methods for POJO admin interfaces that didn't have it. The call to that method is not mandatory, and it is used just to present yourself to shark, so that shark could log (in the future when we improve logging) who did which administrative operation.
- New client CacheAdministration API is written. User can now setup shark's caches at the runtime, as well as he can see the size and the number of objects within caches.
- Defined getProperties method in CORBA's Shark interface, and fixed bug in appropriate POJO implementation of this method.
- When rollback happens, the more informative stack trace is printed/logged.
- WfXXXIterator implementations become independent of Bean Shell interpreter. Now, it is possible to set "query grammar", so that expressions passed to iterators can be written in Java-like style, JavaScript and Python style (the last is not tested). If one does not set grammar, BeanShell is used by default as it was previously.
- WfXXXIterator implementations got additional criteria for querying (i.e. WfProcessIterator implementation now can perform search based on Requester Id, on the time when process starte, on the time of last process state, and on number of active activities).
- There was a big project restructuring. This mainly concerns with the source's input directory, and binary output directory structure. The most important change from user aspect is easier switch to another DB. Now, you do not have to edit 3 files (Shark.conf, recreateDB and LoaderJob.olj), but only configure.properties file that can be found in the root of binary output. In this file, you have to set some properties, and call configure script to switch to another DB. Also, lib directory has more informative structure, execution scripts (except the configuration script) are now in the bin subfolder of binary output and Shark.conf is now in conf subfolder of binary output.
- Shark can use interfaces or abstract classes as variables - you should define XPDL variables so that they have null for initial value, and concrete implementation of variable can be created by executing some tool agent.
- XPDL examples are improved and extended to show more shark capabilities.
- Improved logging and exception handling
- ToolAgent API is extended to have possibility to get an info from ToolAgent (client interface also has this possibility)
- Corresponding Hibernate implementations for all DODS implementations of shark's persistence APIs are contributed by Vladislav Pernin from Open Wide, but they are only included in source distribution since there is no one to maintain them.
- Fixed bug related to getting the last state time of the process (it was not written to DB at all).
- Fixed bug related to use of configuration parameter for deleting processes from DB after its completion.
- Fixed bug related to assignment reevaluation.
- Fixed bugs that could occur when aborting/terminating/suspending/resuming block activities.
- Fixed Shark.configureFromJar() method, so that shark can be used in AIX WebSphere environment.
- Fixed shark kernel bug considering DOS and Unix file separators (slashes and back-slashes) and passing external repository relative path of xpdl file (when repository has some structure).

- Fixed: shark didn't allowed process state transition from "open.not_running.not_started" to "closed.terminated" or "closed.aborted" and it is allowed by spec.

- Admin applications are changed to show new possibilities: deadline checking, limit checking, updating packages, synchronizing XPDL caches, unloading packages, deletion of finished processes, assignment reevaluation, runtime cache administration, ...

What's new in release 1.0-beta2

- Shark is not CORBA engine anymore, but is a library that can be deployed as CORBA service, in EJB container, used directly from a WEB or Swing application, ...

- Shark became very, very modular and configurable. It is consisted of many configurable and replacable parts, and even kernel can be extended or replaced.

- Shark can be configured in serveral ways, including configuration through configuration file and Java's Properties.

- Shark is now fully transactional engine (when performing some operation through client interface, either all consequences of this operation are written to DB, or everything is rolled back) - it is always consistent, and in valid state.

- Shark can be used in a claster (from many VMs targeting the same DB)

- In a runtime, shark is JDK 1.3.1 compatible.

- Default shark's configuration (kernel and internal APIs) does not open new threads - everything is done from the client thread

- There are pretty well defined set of APIs, both client, and internal ones for different shark components. Based on these internal APIs, one could write its own implementations, and use it in shak. There are many APIs and various implementations defined, i.e. Assignment API (standard implementation), Authentication API (DODS and LDAP implementation), Caching API (LRU implementation), InstancePersistence API (DODS implementation), Working (kernel) API (standard implementation), LimitAgent API (Timer implementation), Logging API (standard log4j implementation), MapPersistence API (DODS implementation), ProcessLocking API (Memory and DODS implementations), Scripting API (standard implementation with JavaScript, BeanShell and Python interpreters), SecurityAPI (standard implementation), ToolAgent API (standard implementation with several tool agents), Transaction API (DODS implementation), UserGroup API (DODS and LDAP implementations), UserTransaction API (DODS implementation).

- CORBA Client API changed accordingly to the new Shak's client API

- External transactions are supported - Client API is duplicated with transaction parameters (instance of transaction interface). Kernel simply uses this transaction instead of creating its own and does NOT commit it or roll it back...

- Tool agent support based on WfMC specification of Interface3 - you can write your own tool agents, and they can be executed by Shark server. Tool Agents that come with Shark enable you to perform some simple math operations, start some system application, send mails, perform arbitrary scripts through the Java Script interpreter or Bean Shell interpreter, ... Who ever writtes its own tool agents or Tool agent applications, he must be awared that default shark kernel suposes that they are executing quickly inside client transaction (new threads are not opened for executing tool agents).

- There has been implemented support to use complex Java objects as process variables. You can define XPDL to use some Java class as data type through ExternalReference, and then use this type in XPDL. In runtime, shark will create the object of this class and use it (of course, this class must be in class path when starting shark).
- Shark comes with some predefined sql scripts, and support to easily configure shark to be used with DB other then default HSQL.
- Shark engine comes with several examples of its usage:
  - Directly from Swing Admin application
  - Deployed as CORBA service, and used through CORBA Swing applications
  - Deployed as war file in some web container, and used through the JSP worklisthandler application
- Documentation is generated in html and pdf format using docbook.
- There is a QuickStart document that can lead you through all shark documentation
- Implemented event history support.
- Implemented possibility to enable/disable creation of certain processes
- Implemented possibility to query processes by the state of their variables.
- Package handling events are no longer persisted to database, but logged to file.
- In default implementation, If there is no mapping, the user that created the process gets all workitems.
- Shark does not hand over its internal kernel memory to clients or to it internal API implementations.
- There is a switch for re-evaluation of assignments at startup
- You can define a Tool activity with MANUAL start, and AUTOMATIC finish mode
- The information on finished processes is by default kept in DB

Shark project jar files

Shark engine:

- activation.jar (version 2.3.1)
- ant.jar (version 1.5)
- antcontrib-1.0b1.jar (version 1.5)
- axis.jar (version 1.4)
- bsh-1.2b8.jar (version 1.2b8)
- commons-cli-1.0.jar (version 1.0)
- commons-discovery.jar (version 0.2)

- commons-logging.jar (version 1.0.4)
- concurrent.jar
- connector-1_5.jar
- dbmanager.jar (version 7.1-1)
- dbmanager-api.jar (version 7.1-1)
- dsconnection.jar (version 7.1-1)
- eaf_api.jar (version 7.1-1)
- eafconfreader.jar (version 7.1-1)
- eaflog4j.jar (version 7.1-1)
- stdcaches.jar (version 7.1-1)
- stdconnection.jar (version 7.1-1)
- stdtransaction.jar (version 7.1-1)
- tro-jta.jar (version 7.1-1)
- tro-localcontext.jar (version 7.1-1)
- hsqldb.jar (version 1.8)
- howl.jar
- jaxrpc.jar (version 1.1)
- jotm.jar
- jotm_iiop_stubs.jar
- jotm_jrmp_stubs.jar
- jta-spec1_0_1.jar
- js.jar (1.5R4)
- jython.jar (version 2.1)
- junit.jar (version 3.8.1)
- log4j.jar (version 1.2.11)
- mail.jar (version 1.2)
- objectweb-datasource.jar
- ow_carol.jar
- Octopus.jar (version 3.6-1)
- OctopusConf.jar (part of shark project)

- optional.jar (version 1.5)
- sharkadminapi.jar (part of shark project)
- sharkapppersistence-dods.jar (part of shark project)
- sharkapppersistence-dodslayer.jar (part of shark project)
- sharkassignment-historyrelated.jar (part of shark project)
- sharkassignment-standard.jar (part of shark project)
- sharkassignment-xpdlstraightparticipantmapping.jar (part of shark project)
- sharkcaching-lru.jar (part of shark project)
- sharkcaching-simple.jar (part of shark project)
- sharkclientapi.jar (part of shark project)
- sharkcommonapi.jar (part of shark project)
- sharkcorbaclientapi.jar (part of shark project)
- sharkeventaudit-dods.jar (part of shark project)
- sharkeventaudit-dodslayer.jar (part of shark project)
- sharkeventaudit-notifying.jar (part of shark project)
- sharkeventaudit-smtp.jar (part of shark project)
- sharkinstancepersistence-dods.jar (part of shark project)
- sharkinstancepersistence-dodslayer.jar (part of shark project
- sharkinteroperability-wfxml.jar (part of shark project)
- sharkinternalapi.jar (part of shark project)
- sharkkernel-standard.jar (part of shark project)
- sharklogging-eaf.jar (part of shark project)
- sharklogging-standard.jar (part of shark project)
- sharkpartmappersistence-dods.jar (part of shark project)
- sharkpartmappersistence-dodslayer.jar (part of shark project)
- sharkrepositorypersistence-dods.jar (part of shark project)
- sharkrepositorypersistence-dodslayer.jar (part of shark project)
- sharkrepositorypersistence-filesystem.jar (part of shark project)
- sharkscripting-standard.jar (part of shark project)
- sharksecurity-standard.jar (part of shark project)

- sharktoolagent-standard.jar (part of shark project)
- sharkusergroup-dods.jar (part of shark project)
- sharkusergroup-dodslayer.jar (part of shark project)
- sharkutilities-dods.jar (part of shark project)
- sharkutilities-dodslayer.jar (part of shark project)
- sharkutilities-geronimo.jar (part of shark project)
- sharkutilities-map.jar (part of shark project)
- sharkutilities-misc.jar (part of shark project)
- sharkutilities-wmentity.jar (part of shark project)
- sharkutilities-xpil.jar (part of shark project)
- sharkwebservice-asapapi.jar (part of shark project)
- sharkwebservice-wfxmlapi.jar (part of shark project)
- twexpdl.jar (version 2.1-1 - part of jawe project)
- util.jar (version 7.1-1)
- wsdl4j.jar (version 1.5.1)
- xalan.jar (version 2.5)
- xapool.jar
- xercesImpl.jar (version 2.8)
- xml-apis.jar (version 2.8)

Installation, platform, and system information

These release notes apply to Enhydra Shark 2.0-2.

Installation

Enhydra Shark 2.0-2 is a complete release, and should not be installed over an existing Enhydra Shark installation.

Platform support

Through the open-source development process, Enhydra Shark has been used on a wide variety of platforms and with many different JDBC-supported databases.

Databases

It is tested with following databases:

- DB2
- HypersonicSQL
- MSQL
- MySQL
- Oracle
- PostgreSQL

JDKs

- Sun JDK 1.3.1 or higher on Windows 2000 and Linux.

Contributions

Many people contributed to shark project with their questions, ideas, testing and "bug discovering", and we want to mention some of them who even wrote some shark parts, or send some patches. We are appologizing to all the people we forgot to mention here:

- Andy Zeneski <jaz@ofbiz.org>

  Contributed LimitAgent API, and its simple Timer implementation, as well as pointed to some bugs, send some patches, and had a lot of useful suggestions.

- David Forslund <dwf@lanl.gov>

  CORBA patches, improvements, suggestions, POA implementation of Shark CORBA server.

- Paloma Trigueros <paloma.trigueros@mad.tecsidel.es>

  Contributed classes for JavaClassToolAgent for sending e-mails. Also contributed classes for JavaClassToolAgent for sending/receiving JMS messages (which are not included in the project yet.), and started to write Mail tool agent

- Rich Robinson <robinsonrc@hotmail.com>

  Contributed History related implementation of Assignment API, whcih reads XPDL extended attributes and look at what shark has already been doing before. Also, he sent some client application/utilities for shark beta1 and shark beta2.

- Rick Ross <rross@genvault.com>

  Contributed several tutorials of how to use shark. They can be found at shark's home page under documentation.

- shijun <havedreams@163.com>

Sent some pressure tests for shark beta1

- Vladislav Pernin (vladislav.pernin@openwide.fr)

  Contributed Hibernate layers similar to DODS layers for all APIs that have such DODS layers (more than 10 layers). Unfortunatelly, these layers are not worked out completely, and there is nobady to maintain them (especially InstancePersistence layer), and are included only as a source code.

- ####### ###### ########## <sharapov@mcc.elektra.ru>

  Sent CORBA web client for beta1 version of shark

- Dirk Hoffmann <dh.discuss@web.de>,

  ToolAgentLoader for being able to add new Tool Agents at runtime. Also, he sent some patches for shark.

- Mathias Holst <holst@igd-r.fraunhofer.de>

  Implementation of Shark JSP client - not included in the release.

- Abe Achkinazi <aachkinazi@bdnacorp.com>

  Implementation of SchedulerToolAgent

- Daniel Frey <qwer12341@gmx.ch>

  Notifying EventAuditManager. A lot of suggestions for shark improvement.

- Schimmack, Roland <SC@ProCom.de>

  Many useful suggestions regarding CORBA implementation.

- Wolters, Oliver <WT@ProCom.de>

  Many useful suggestions regarding CORBA implementation.

- Till Kothe <Till.Kothe@de.ibm.com>

  Improvements of SMTP Event Audit Manager

- Javier Reyes Gómez <Javier.Reyes@iconmedialab.es>

  DODS patches for Oracle DB.

- Ivan Roza Medina <Ivan.Roza@iconmedialab.es>

  DODS patches for Oracle DB, many useful suggestions.

- Peter Niederlag <peter@niederlag.de>

  Many useful suggestions.

- Vojtech Huser <huser56@minfor.net>

  Many useful suggestions.

- Günther Wieser <gwieser@creative-it.com>

Many useful suggestions.

- Gretchen Moran <gmoran@pentaho.org>

  Many useful suggestions, DeadlineInfo API implementation, support for Firebird (not yet included)

- Geeta Ramani<GRamani@intellicare.com>

  Many useful suggestions, supporting users on mailing list, discovering bugs, ...