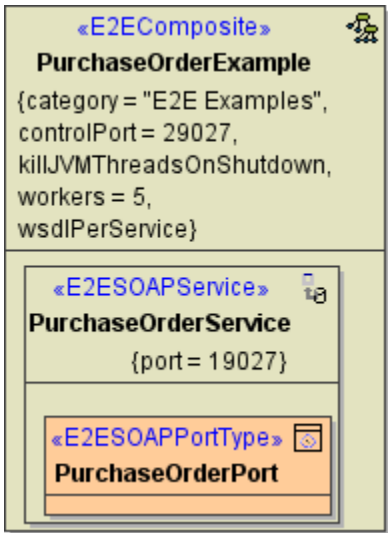


Persistent State Components

As persistent state is an integrated part of the Bridge, there is no requirement for additional components. The Components Wizard is used to create this diagram. For more details, refer to [The Components Wizard](#).

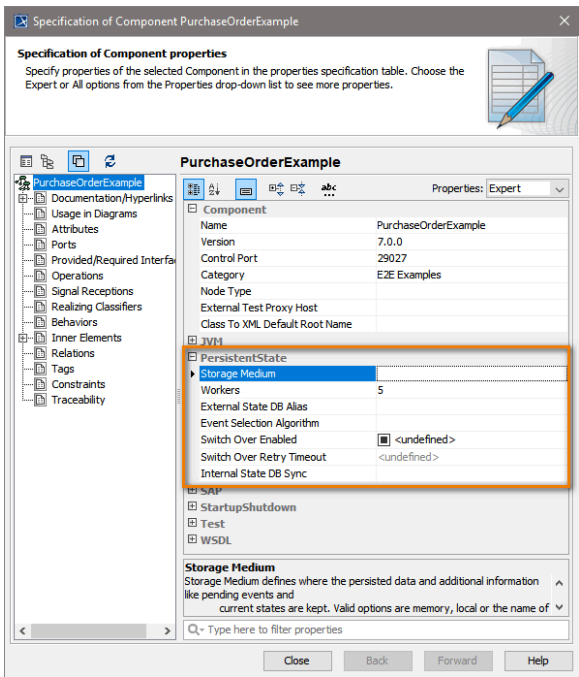
The picture below shows the component diagram of the Purchase Order example.






Related Pages:


- [SQL Adapter](#)
- [Performance Considerations of Persistent State](#)
- [License for Running xUML Services](#)
- [Load Balanced Persistent State](#)

In the specification dialog of the **xUML service composite**, you can configure the persistent state engine.



Attribute	Type	Description	Value	Description
Persistent State				
Storage Medium	String	Storage Medium defines where the persisted data and additional information like pending events and current states are kept. See section Data Storage for a discussion of the options. When using a database instance as storage medium, make sure to configure components and deployment of the SQL adapter as described in section SQL Adapter .	memory	The persistent state data is kept in memory. On service shutdown, data is written to the file system.
			local (default)	The persistent state data is kept in a local SQLite database that is delivered with the Bridge.

			external	The persistent state data is kept in an external database that has to be defined as a service backend. Use External State DB Alias to specify the database. This storage medium is needed to implement a load balanced persistent state setup .
			volatile	The persistent state data is kept in memory. On service shutdown, data is lost.
Workers	Integer	<p>Workers defines the maximum number of parallel threads used to process pending events. The default setting is 5, which is used if none or 0 workers are configured. The implications of more or less workers are discussed in Performance Considerations of Persistent State.</p> <div>  Each active worker requires one license slot (concurrent connection) to process activities. For more information on licensing and concurrent connections, refer to License for Running xUML Services. </div>		
External State DB Alias	String	<p>If you are using an external storage medium, you can specify the DB alias in External State DB Alias.</p> <p>See also External Properties State DB Alias.</p>		
External Properties State DB Alias	String	<p>Runtime 2019.8 Builder 7.6.0 Specify the database alias of the external database you want to store external persistent data to. For more information on external data, refer to Persistent State Classes > External Persistent Data.</p>	<tagged value not specified> (default)	If not specified, external persistent data will be stored to the main persistent state database (internal or external database as defined with External State DB Alias).
			database alias	The external persistent data will be stored to the database indicated by the alias. The main persistent state database can be still internal, though.
Event Selection Algorithm	String	<p>Runtime 2019.2 Builder 7.4.0 The xUML Runtime processes the persistent state events in a defined order. Select the event selection algorithm the xUML Runtime should use to define this order of events. Refer to Performance Considerations of Persistent State > Event Selection Algorithm for more details on when to use which algorithm.</p> <div>  Services using this tagged value (all options but the default <not specified>) will not start on a Runtime below 2019.2. </div>	<tagged value not specified> (default)	Leave the decision which algorithm to use to the xUML Runtime. If future versions provide changed algorithms, they will be taken into account automatically.
			Default	Basically the same as <tagged value not specified>: Leave the decision which algorithm to use to the xUML Runtime. If future versions provide changed algorithms they will be taken into account automatically. Additionally, the tagged value will appear on the Bridge in the Settings tab of the xUML service and can be changed there.
			Favour Signals	Process events according to signal appearance. This is the algorithm used up to Runtime 2019.1.
			Favour Objects	Process events according to object age. This results in processing older objects first. Signals to a selected object are processed in the order they arrived.
Switch Over Enabled	Boolean	<p>This flag enables the automatic fail over mechanism for clustered persistent state databases. If the persistent state database becomes inoperative, the E2E xUML Runtime will try to open a connection to compensatory database of the cluster. See also option Switch Over Retry Timeout.</p> <div>  This option is available for clustered Oracle databases only. </div>	false (default)	fail over mechanism not enabled
			true	fail over mechanism enabled
Switch Over Retry Timeout	Integer	<p>During fail over, the E2E xUML Runtime will try to create a new database connection to a compensatory database (see Switch Over Enabled). If this fails, the xUML Runtime will try to open a new connection every second until the timeout (in seconds) is reached. Default is 600 seconds.</p>		

Internal State DB Synch	String	<p>Internal State DB Synch defines the level of file system synchronization performed on the internal persistent state database.</p> <div>  <p>For production setup, we recommend using FULL synch.</p> </div>	OFF (default)	SQLite continues without syncing as soon as it has handed data off to the operating system. If the xUML service crashes, the data will be safe, but the database might become corrupted if the operating system crashes or the computer loses power before that data has been written to the disk surface. Very fast.
			NORMAL	The SQLite database engine will still sync at the most critical moments, but less often than in FULL mode (see below). There is a very small (though non-zero) chance that a power failure at just the wrong time could corrupt the database in NORMAL mode. But in practice, you are more likely to suffer a catastrophic disk failure or some other unrecoverable hardware fault. Up to 50 times slower than OFF.
			FULL	The SQLite database engine will ensure that all content is safely written to the disk surface prior to continuing. This ensures that an operating system crash or power failure will not corrupt the database. FULL synchronous is very safe, but it is also slower than NORMAL. FULL is the most commonly used synchronous setting.
			EXTRA	Also synchronize the DB transaction journal after every commit. Equals FULL in E2E Bridge context.