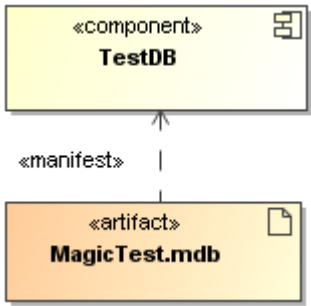
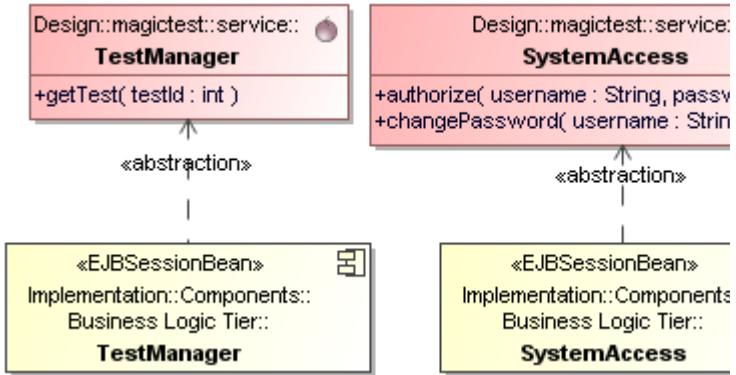
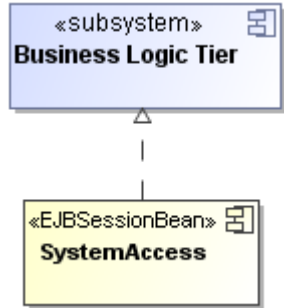
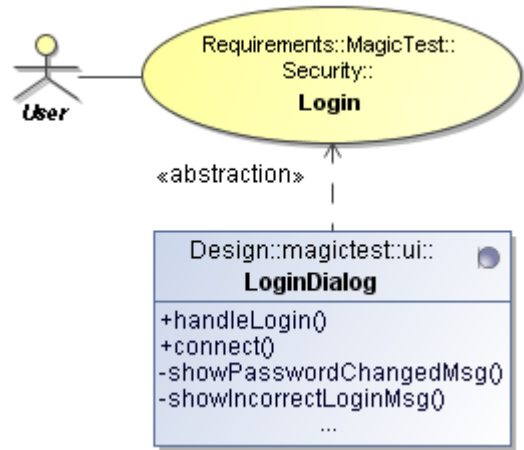


Backward traceability – specification

Property Name	Description	Applied For:	Reference Through:	Value elements type	Example
Manifested in Artifacts	The property shows which components are manifested in an artifact.	Artifact	Relationships: Manifestation	Component	 <pre> graph BT MagicTest.mdb[«artifact» MagicTest.mdb] -- «manifest» --> TestDB[«component» TestDB] </pre>
Specifying Class	The property shows the classes representing the component specification from the Design model.	Component	Relationships: Abstraction	Class	 <pre> graph BT subgraph Design TM[Design::magictest::service:: TestManager +getTest(testId : int)] SA[Design::magictest::service:: SystemAccess +authorize(username : String, passw +changePassword(username : Strin] end subgraph Implementation TM_Im[«EJBSessionBean» Implementation::Components:: Business Logic Tier:: TestManager] SA_Im[«EJBSessionBean» Implementation::Components:: Business Logic Tier:: SystemAccess] end TM_Im -- «abstraction» --> TM SA_Im -- «abstraction» --> SA </pre>
Specifying Component	The property shows the components that are realized by classifiers through component realization.	Classifier	Relationships: Component Realization, Realization	Component	 <pre> graph BT SA[«EJBSessionBean» SystemAccess] -- realization --> BLT[«subsystem» Business Logic Tier] </pre>
Specifying Use Case	The property shows the use cases (from the Requirements model) representing the class specification.	Class	Relationships: Abstraction	Use Case	 <pre> graph BT User((User)) --- Login([Requirements::MagicTest:: Security:: Login]) subgraph Design LD[Design::magictest::ui:: LoginDialog +handleLogin() +connect() -showPasswordChangedMsg() -showIncorrectLoginMsg() ...] end LD -- «abstraction» --> Login </pre>

Specifying Use Case	The property shows the use cases that specify the given use case in the higher level of abstraction. For example, the Business Use Case specifies the Requirements Use Case.	Use Case	Relationships: Abstraction	Use Case	<pre> graph BT MU([Modify User]) -.-> «abstraction» Admin([Administrate]) RU([Remove User]) -.-> «abstraction» Admin </pre>
Realized Interface	The property shows the interfaces specifying the contract, in which the related classifier conforms to.	Classifier	Relationships: Interface Realization	Interface	<pre> classDiagram class NotificationService { +send(address : String, message : String) } class NotificationServer { <<component>> } NotificationServer -- > NotificationService </pre>
Specifying Element, All Specifying Elements	<p>The Specifying Element property gathers specifying elements from the upper abstraction level.</p> <p>The All Specifying Elements property transitively gathers specifying elements from all upper abstraction levels.</p>	Element	Relationships: Abstraction, Component Realization, Interface Realization.	Element	<pre> graph TD subgraph Requirements_model [Requirements model] UC[UC] Activity[Activity] Iteration[Iteration] end subgraph Design_model [Design model] Class[Class] end subgraph Implementation_model [Implementation model] Component[Component] Artifact[Artifact] Interface[Interface] end UC --> Activity UC --> Iteration Activity --> Class Iteration --> Class Class --> Component Class --> Artifact Class --> Interface Component --> Artifact Component --> Interface Artifact --> Interface UC -.-> OwnedBehavior UC_note[OwnedBehavior] Activity -.-> Abstraction Activity_note[Abstraction] Class -.-> Manifestation Class_note[Manifestation] Component -.-> Interface Realization Component_note[Interface Realization] </pre>