OPRoS Component Composing Tool
User Manual

2011. 8. 29.

Document Information

Item	Affiliation	Name	Date	Sign.
Drafter	(주)엔쓰리소프트	서동민	2011-08-26	
Fellow Reviewer				
QA Reviewer				
Approver				
Version				
Date of Issue				
Status				

Revision History

Version	Revised Date	Revision Details	Writer	Checker

Table of Contents

Figure List

1. Overview

The robot component composer tool has functions to fetch the developed OPRoS components, assemble them, develop composite components and applications, and distribute the created applications.

1.1. How to Use This User Manual

This manual is written to follow easily along with the OPRoS job training, and materials for beginners of the OPRoS.

This user manual describes each item of the component composer tool in detail, adds figures to every item, and you can find a page number of the corresponding item through the table of contents.

1.1.1. Purpose

This manual explains easy to follow how to install the component composer tool, how to use it basically, and how to manage it.

1.1.2. How to Use

If you have already installed the Eclipse, refer from Chapter 3.

1.1.3. Appendix

The appendix simply explains terms easily confusable in this document.

1.1.4. Useful Information

Easy-to-use methods are provided with a mark of “TIPS: “ for each item.

2. Component Modeling and Overall Structure of the Composing Tool

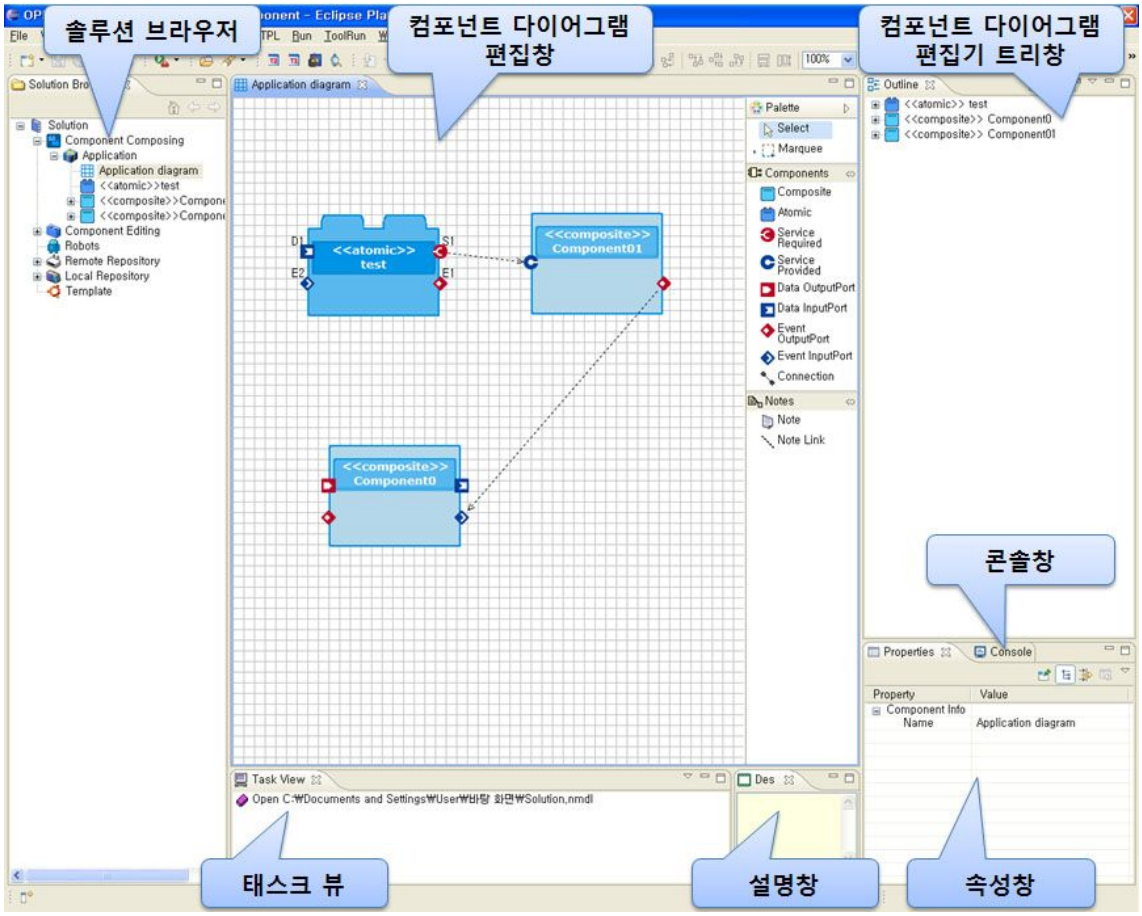

Figure 1. Overall Structure of Composing Tools

It is comprised of the solution browser, which models application and composite components and manages library folders, the description window, which can see description for each component, the component diagram editor, which can model composite components, the component diagram editor tree window, which can identify model elements of a component diagram, the console window, which shows the progress of a project's source, and the task view that shows the progress of modeling.

2.1. Solution Browser

2.1.1. Solution Browser Structure

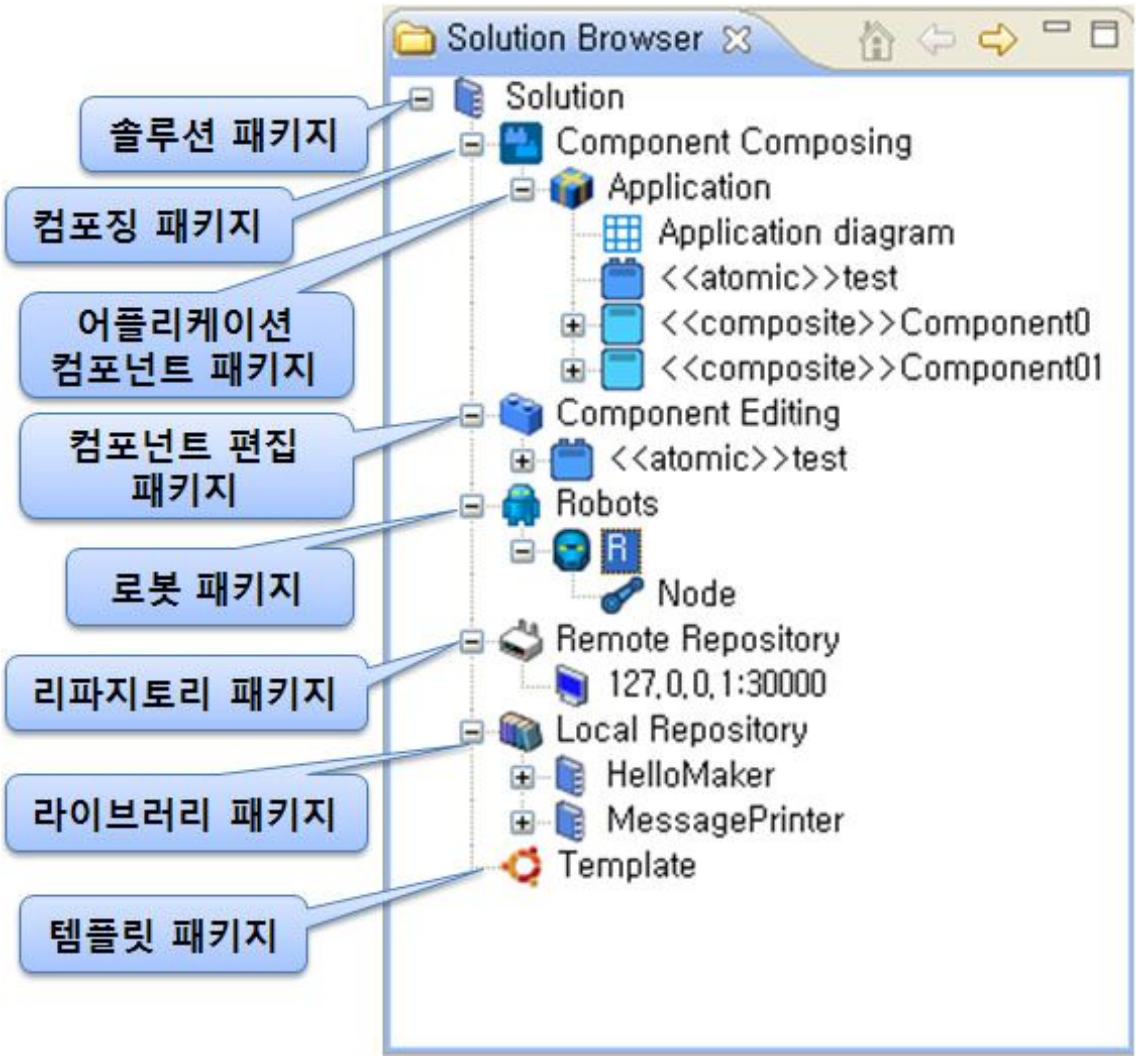

Figure 2. Solution Browser

When executing the tool, it shows the solution browser as the above structure.

- Solution Package: Saves or opens projects, and creates new projects.
- Component Composing Package: Adds new application packages and diagrams.
- Application Component Package: Models application components on the main diagram.
- Component Edit Package: Creates developable or editable components.
- Robot Package: Manages robots and each node of robots.
- Repository Package: Accesses to a remote server computer to fetch libraries

stored in the server, and shows them.

- Library Package: Fetches information in folders specified as the library folder in preferences, and shows them.
- Template Package: Fetches samples of applications or components, folders specified as the template folder in preferences, and shows them.

2.1.2. Preferences

a. Select 'Window->Preferences' on the top menu of Eclipse to open the preferences window.

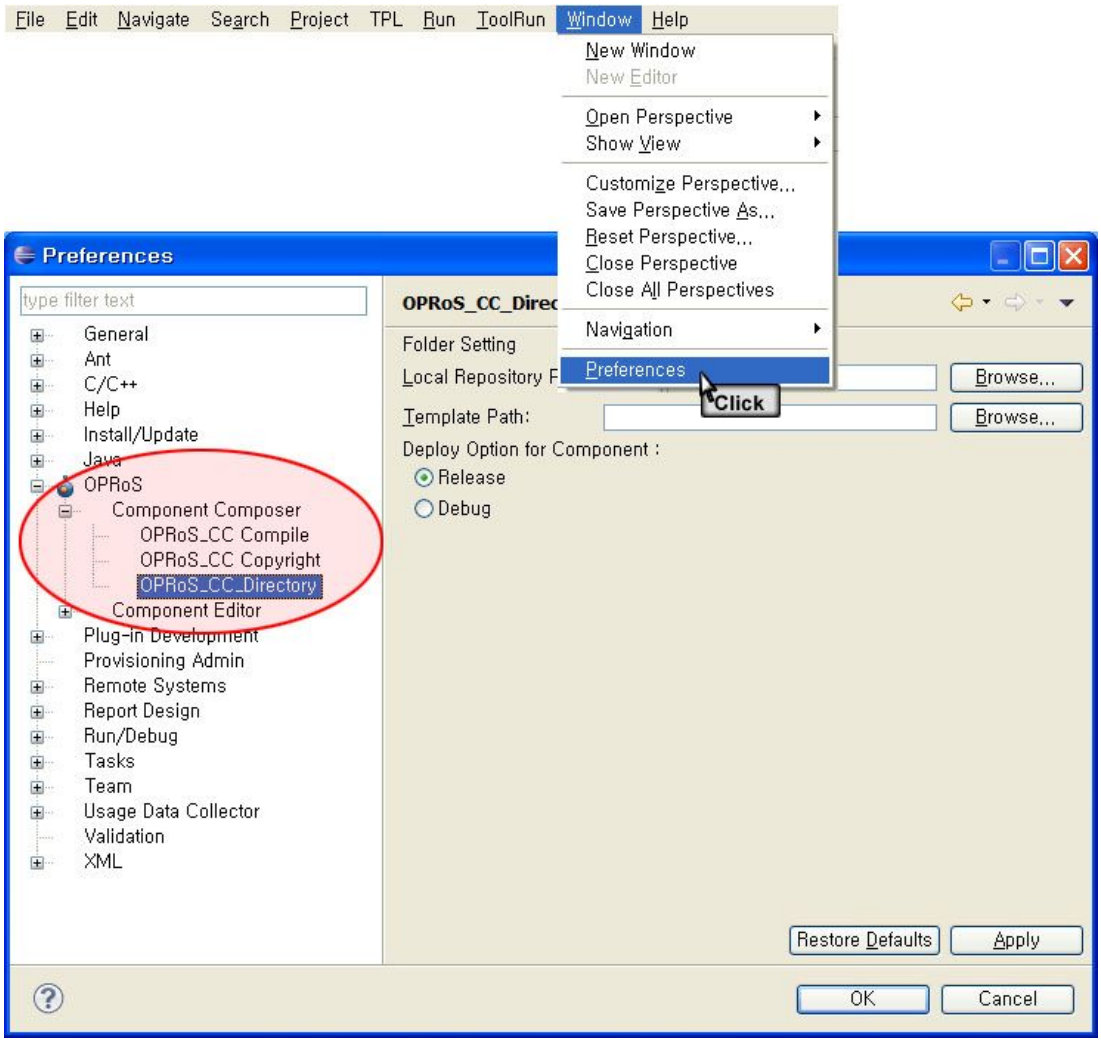

Figure 3. Preferences

b. When specifying a physical path of a library in the preferences window, the specified path is reflected into the solution browser's library package area to be shown.

c. Specify the path to the corresponding program's executable files for the executable file path.

2.1.3. Solution Package

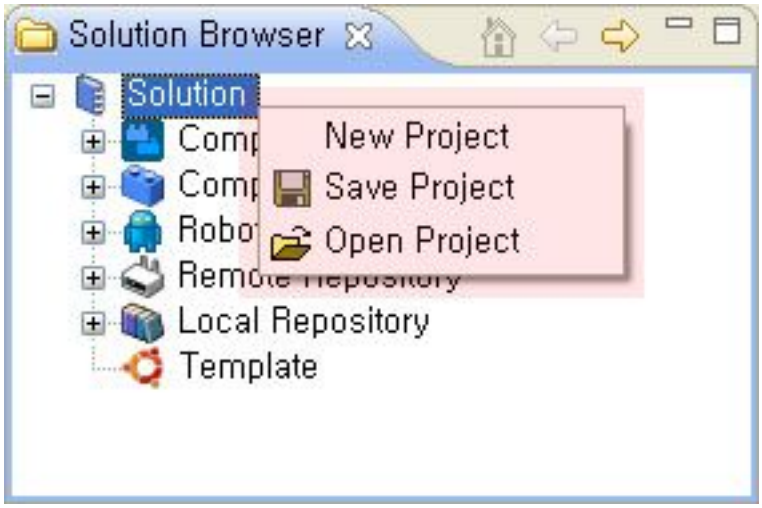

Figure 4. Solution Package

- New Project: Terminates the existing project, and creates a new project.
- Save Project: Stores the project.
- Open Project: Fetches a project.

2.1.4. Component Composing Package

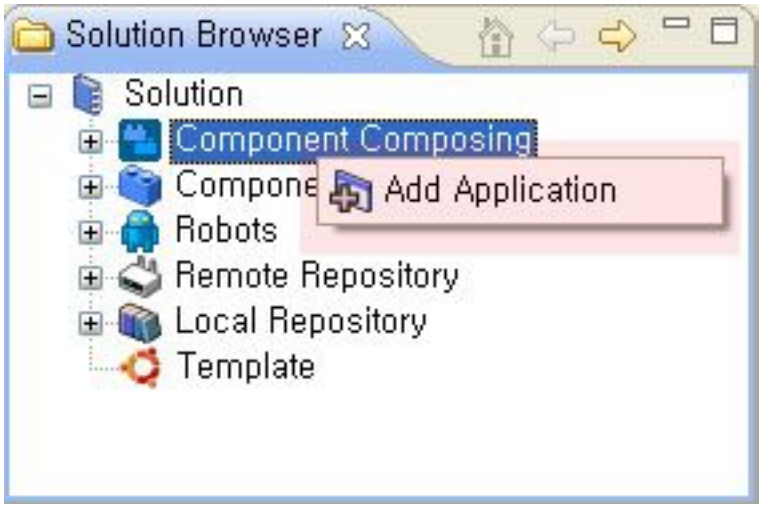

Figure 5. Component Composing Package Popup Menu

- Add Application: Creates a new application package and a dialog.

2.1.5. Application Component Package

a. Package Popup Menu

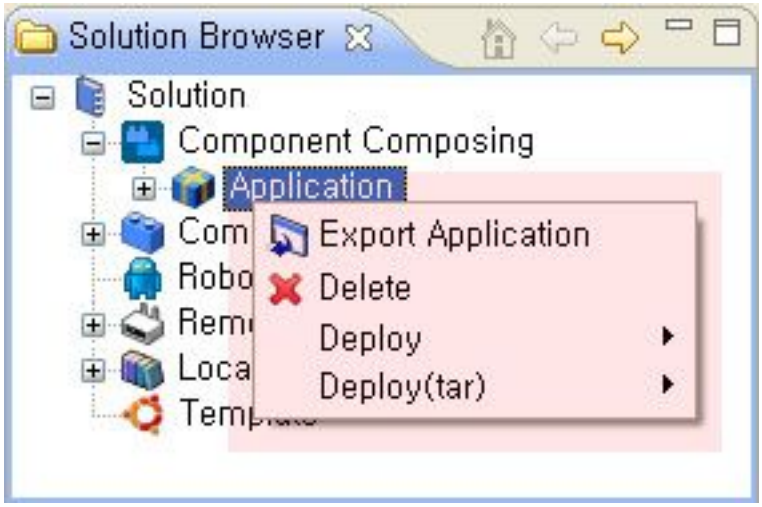

Figure 6. Application Component Package Popup Menu

- Delete: Deletes the selected package.
- Deploy: Deploys the selected package.

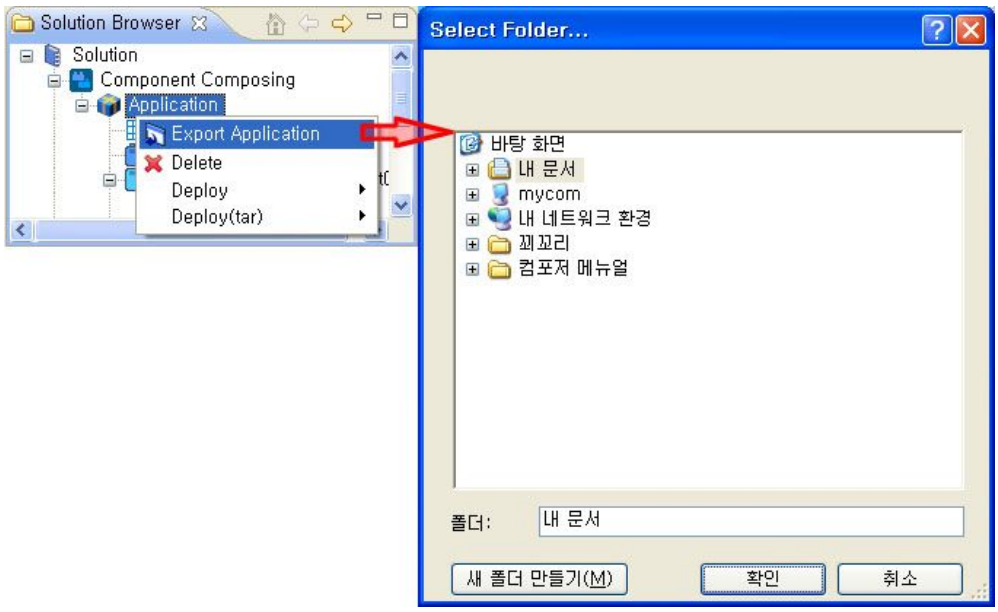

Figure 7. Export Application

- Export Application: Able to distribute the selected application after choosing the specified folder.

b. Component Popup Menu

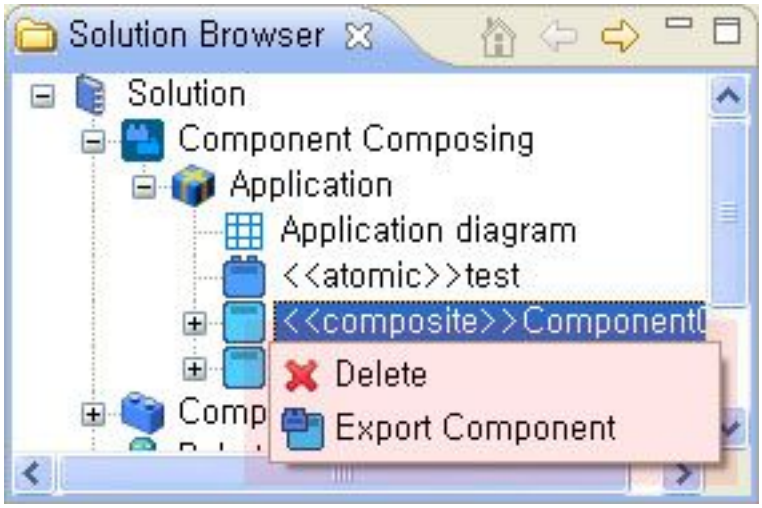

Figure 8. Component Popup Menu

- Delete: Deletes the selected component.
- Export Component: In the case of a composite component, it is exported as a XML form to the outside.

2.1.6. Component Edit Package

a. Package Popup Menu

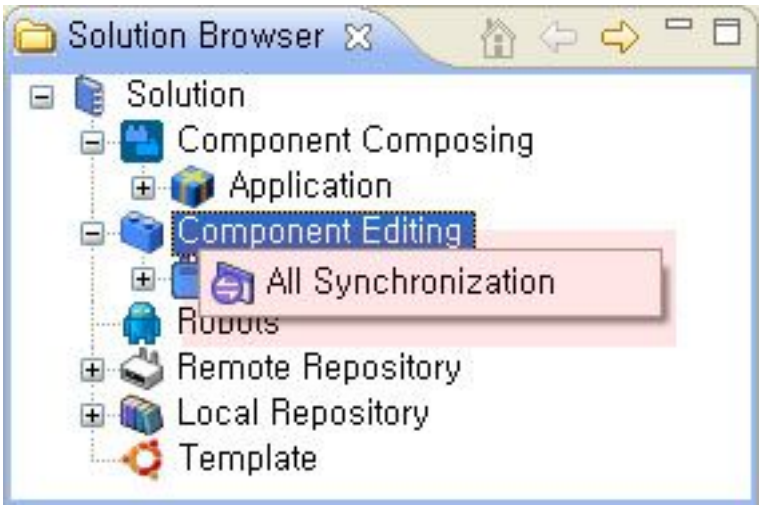

Figure 9. Component Edit Package Popup Menu

- All Synchronization: Synchronizes the composer and editor component information.

b. Component Popup Menu

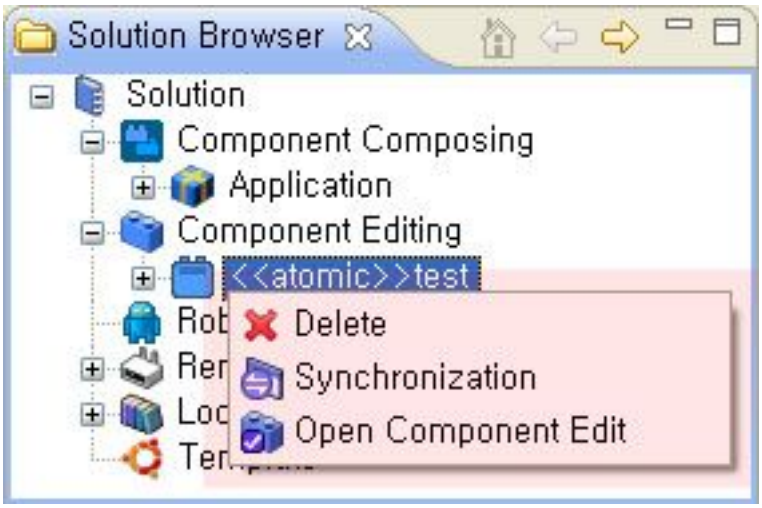

Figure 10. Component Edit Component Popup Menu

- Delete: Deletes the selected component.
- Synchronization: Synchronizes the selected component with component information on the editor.
- Open Component Edit: Creates a project with the selected component on the component editor, and changes the tool into the component editor.

2.1.7. Repository Package

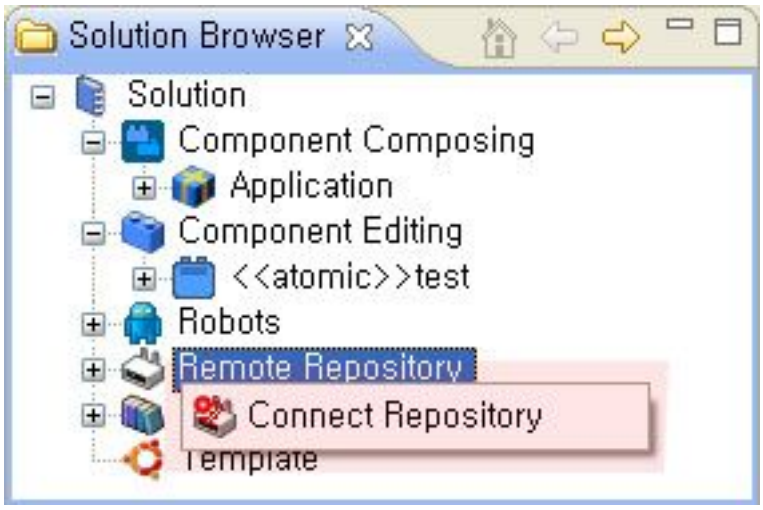

Figure 11. Repository Package Popup Menu

- Connect Repository: Enters information on server computer's IP and ports to add a repository server.

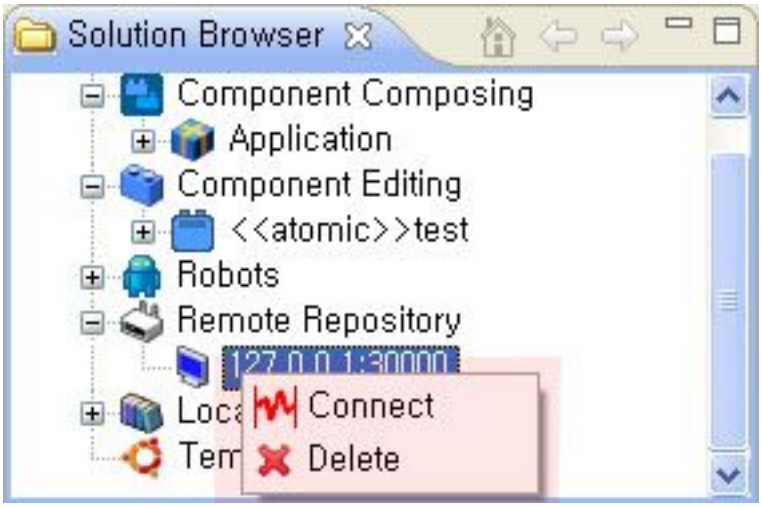

Figure 12. Repository Popup Menu

- Connect: Connects to a repository server to read a library, and shows it.
- Delete: Deletes the repository server.

2.1.8. Library Package

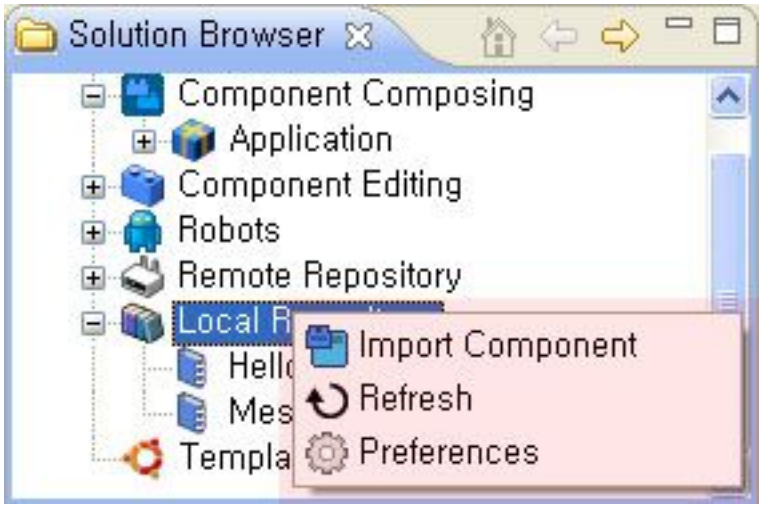

Figure 13. Library Package Popup Menu

- Import Component: It can fetch atomic and composite components in other folders to the selected folder.
- Refresh: If content of the folder storing components is changed, it reads new information on the changed folder and reflects it.
- Preferences: Refer to the preferences

2.1.9. Template Package

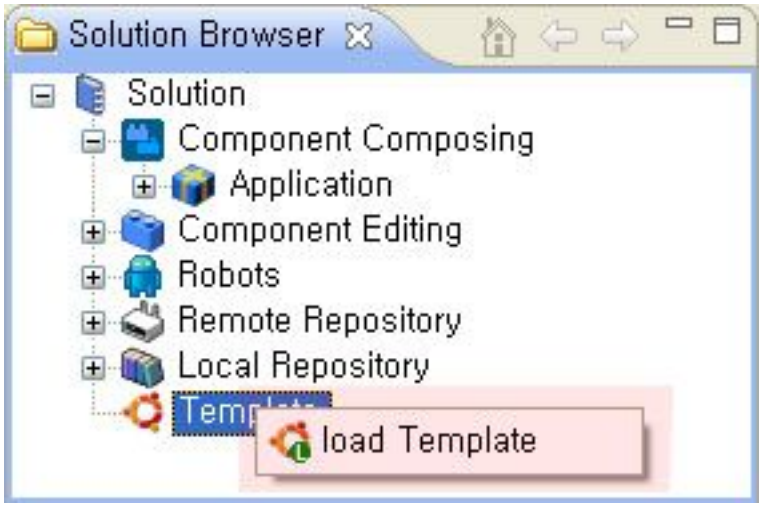

Figure 14. Template Package Popup Menu

- Load Template: Fetches template components from the path specified in ‘Preferences – Template Path.’

2.2. Component Diagram Editor

2.2.1. Layout

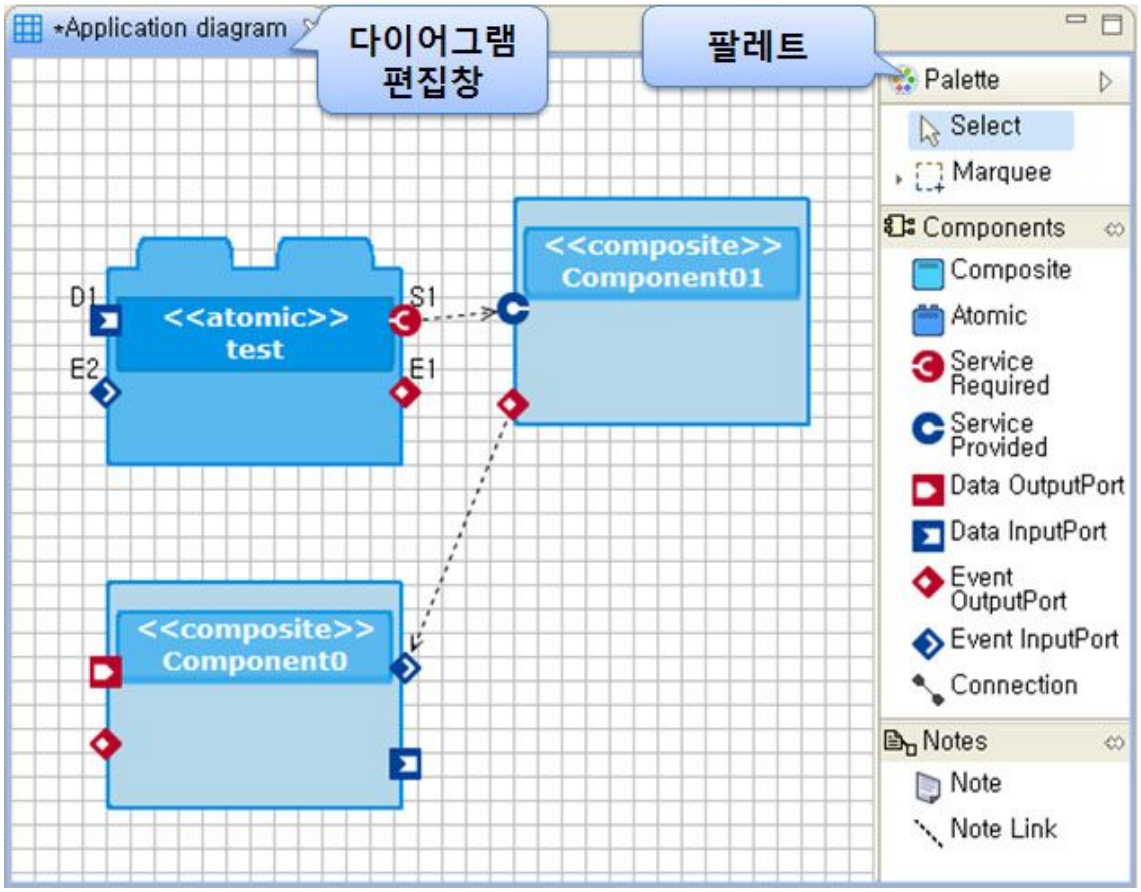

Figure 15. Component Diagram Editor

- Palette: It is a tool that collects components, ports and relations available to use on the diagram, which users can select components and relations to be created from the palette to create them on the diagram edit window.
- Diagram Edit Window: It is an edit window that users can model application and composite components by adding objects in the palette or solution browser.

a. Palette

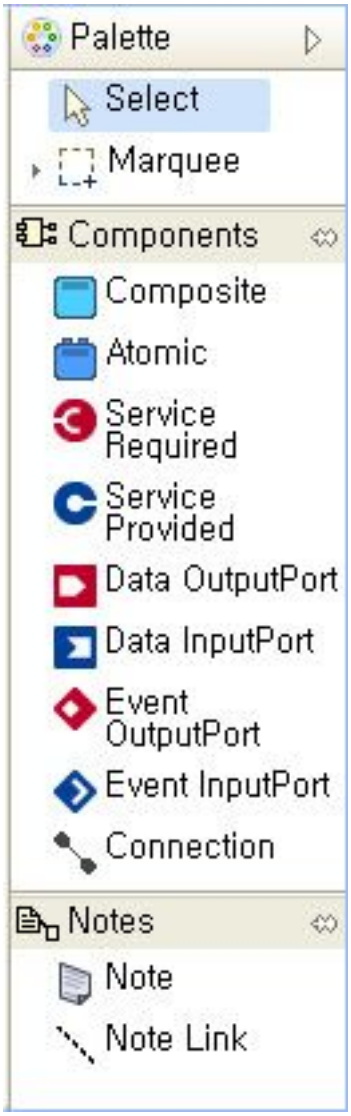

Figure 16. Palette

- Select: Used when selecting a single object on the diagram edit window.
- Marquee: Used when selecting multiple objects on the diagram edit window.
- Composite: Used when creating a composite component on the diagram edit window.
- Atomic: Used when creating an atomic component on the diagram edit window.
- Service Required: Used when adding a 'Service Required' to the selected component on the diagram edit window.
- Service Provided: Used when adding a 'Service Provided' to the selected component on the diagram edit window.
- Data OutputPort: Used when adding a 'Data OutputPort' required to the selected

component on the diagram edit window.

- Data InputPort: Used when adding a 'Data InputPort' to the selected component on the diagram edit window.
- Event OutputPort: Used when adding an 'Event OutputPort' to the selected component on the diagram edit window.
- Event InputPort: Used when adding an 'Event InputPort' to the selected component on the diagram edit window.
- Connection: Used when connecting a component's port.
- Note: Used when adding a note.
- Note Link: Used when connecting objects (components, ports) related to a note.

b. Diagram Edit Window

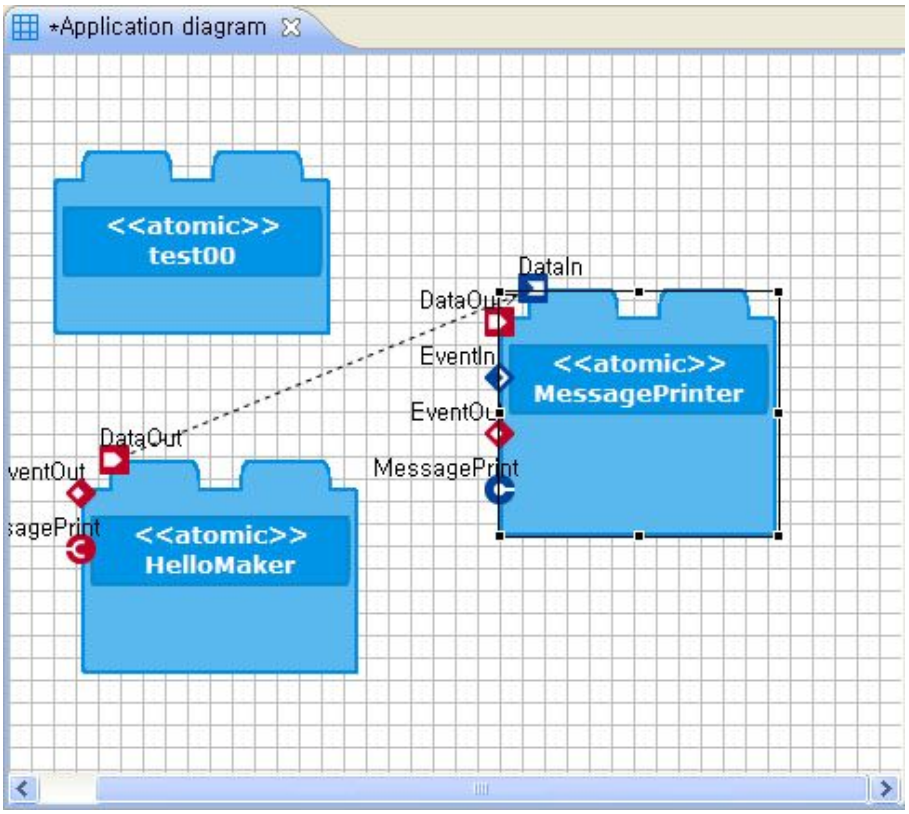

Figure 17. Diagram Edit Window

2.2.2. Create Composite Components

a. Select the Composite from the palette to create a new composite component.

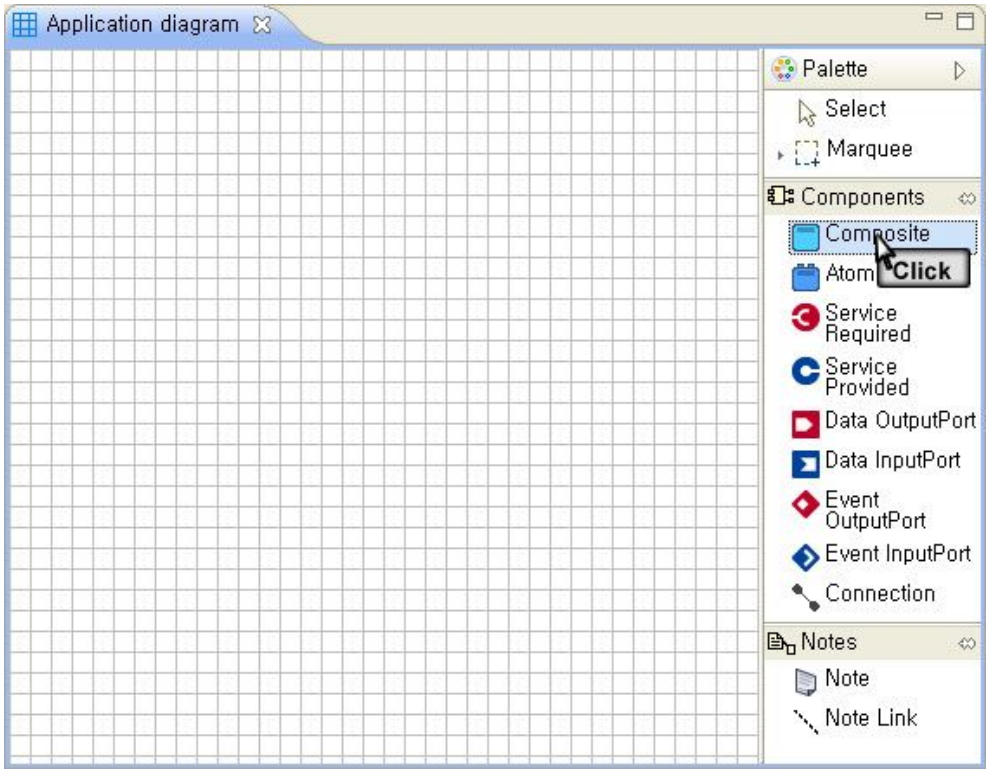

Figure 18. Create Composite Components 1

b. Click the location to draw the component on the diagram edit window.

c. A basic component is created on the diagram edit window as below.

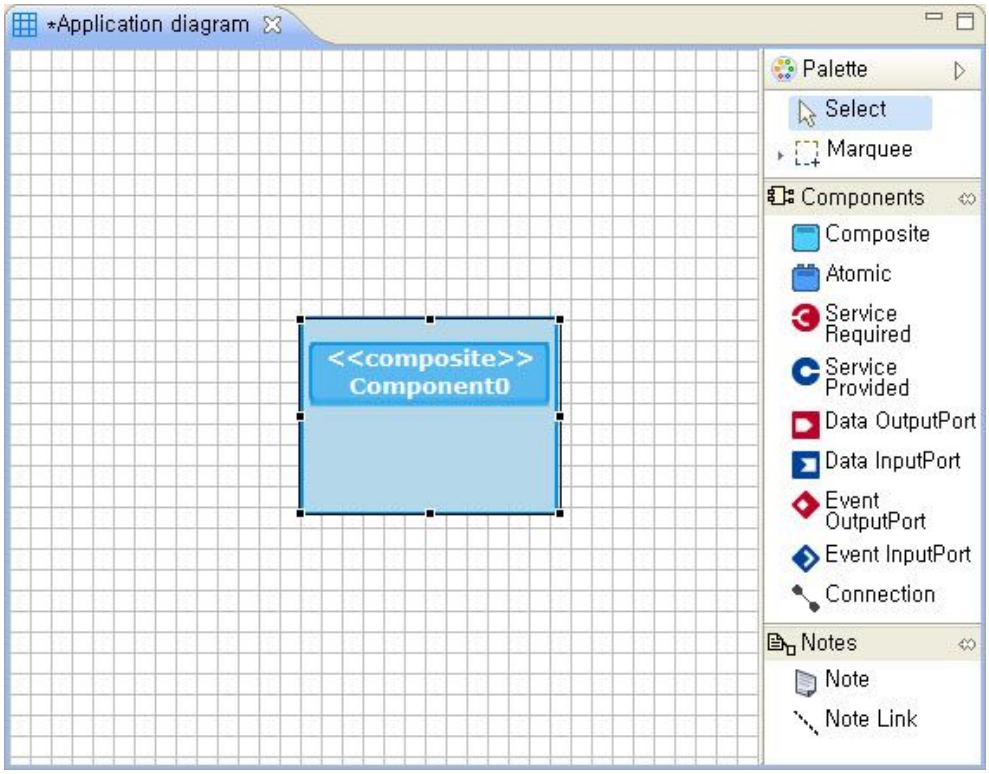

Figure 19. Create Composite Components 2

2.2.3. Create Atomic Components

a. Select the Atomic from the palette to create a new atomic component.

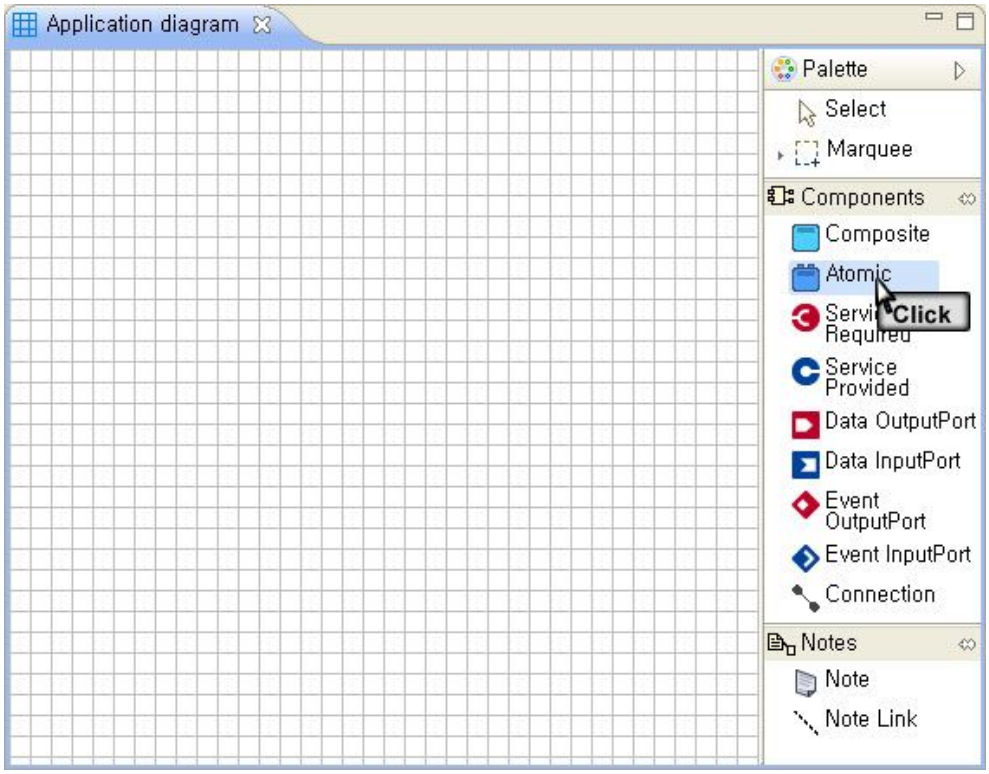

Figure 20. Create Atomic Components 1

b. Click the location to draw the atomic component on the diagram edit window.

c. When the 'New OPRoS Component' window is appeared to ask about information on the atomic component to be created, enter input items and click OK.

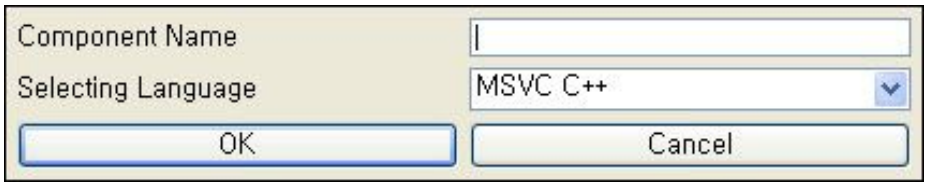

Figure 21. Create Atomic Components 2

- Component Name: Name of the component to be created
- Selecting Language: Select a development language between MinGW C++ and MSVC C++

d. A basic atomic component is created on the diagram edit window as below.

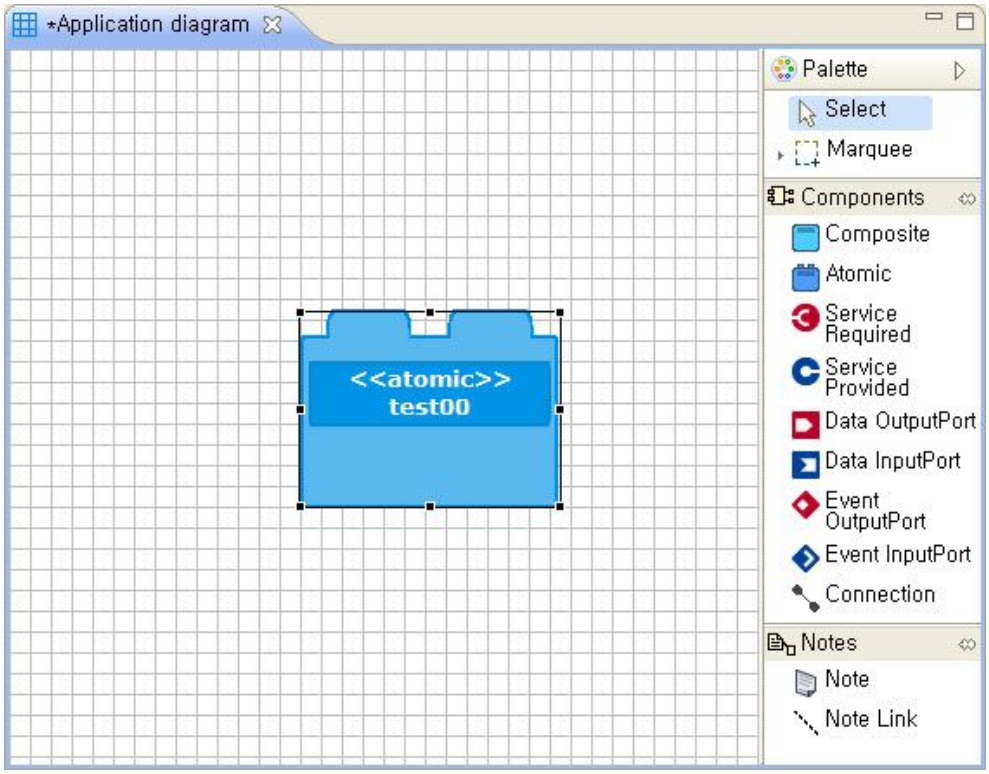

Figure 22. Create Atomic Components 3

e. In a different way, select an atomic component from the solution browser.

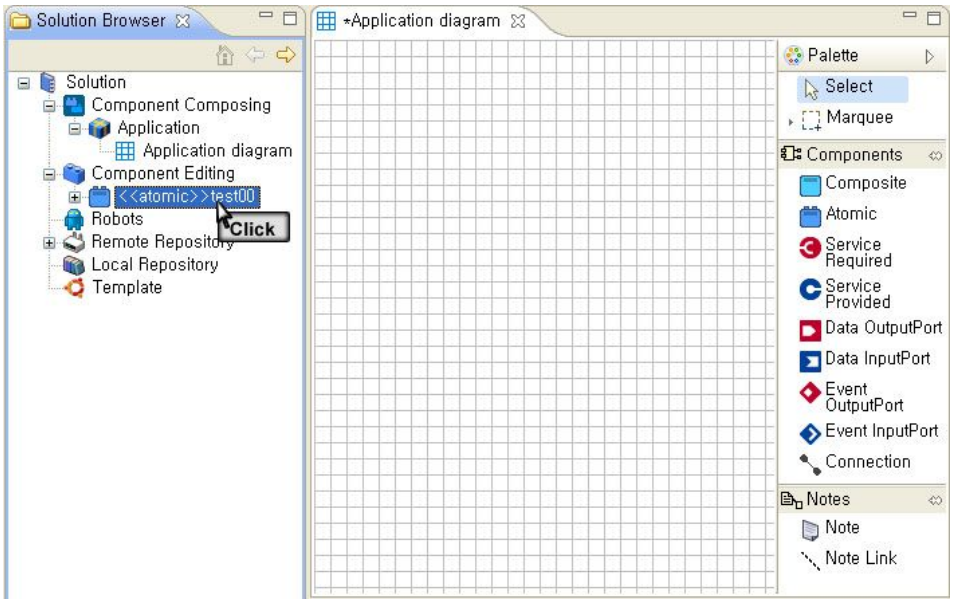

Figure 23. Creates Atomic Components 4

f. Drag and drop it to the diagram edit window while clicking the mouse.

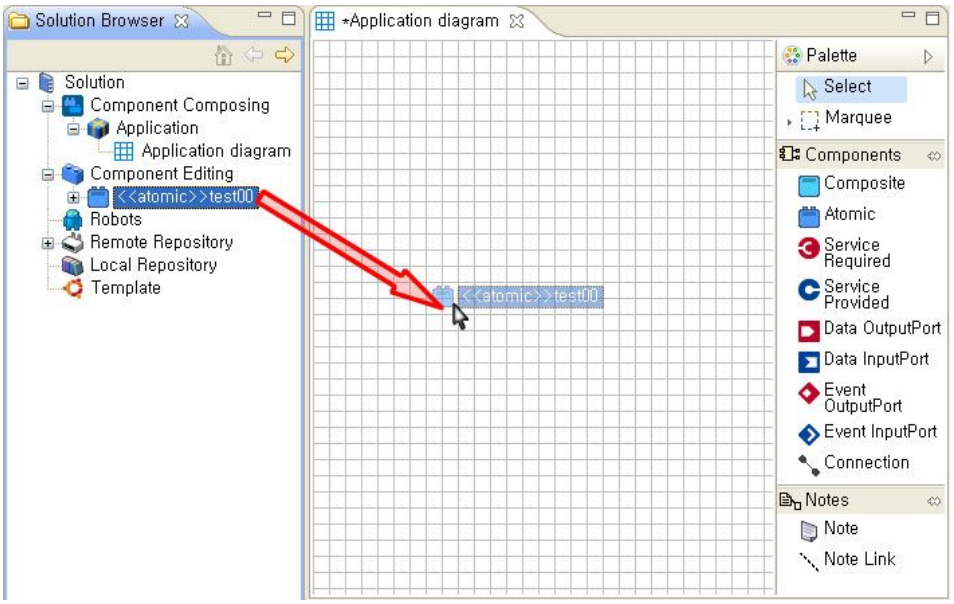

Figure 24. Create Atomic Components 5

g. When releasing the mouse button, an atomic component is drawn on the diagram edit window.

2.2.4. Create Composite's Ports

a. Select a port to be added from the palette.

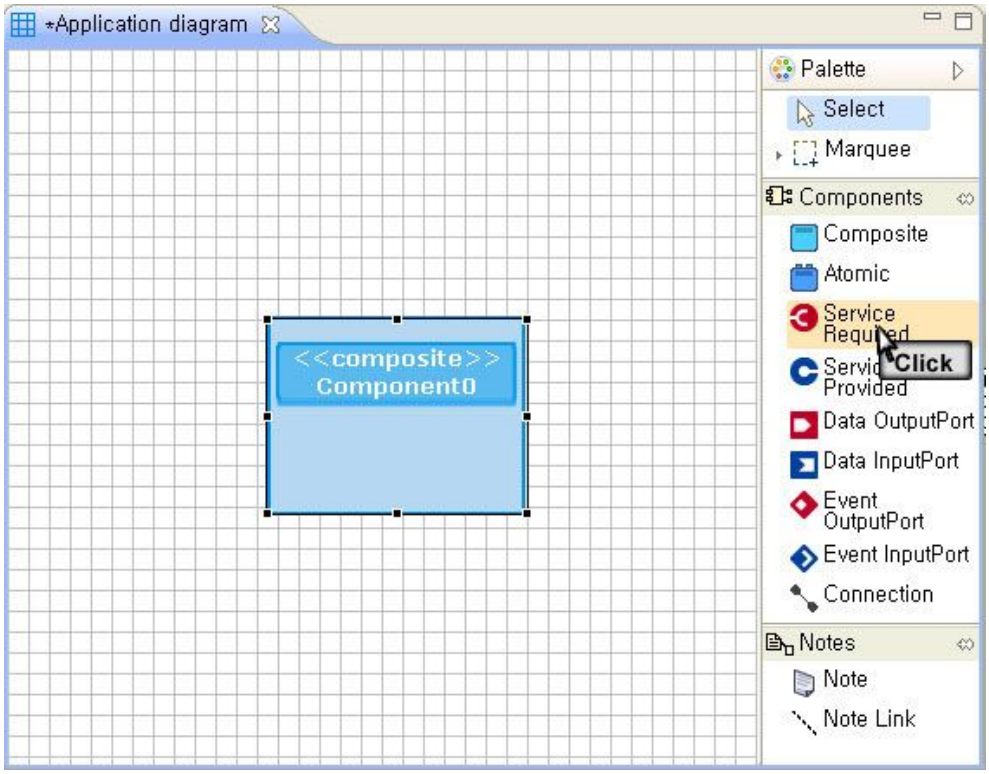

Figure 25. Create Composite's Ports 1

b. Click the component, which you want to add a port, on the diagram edit window.

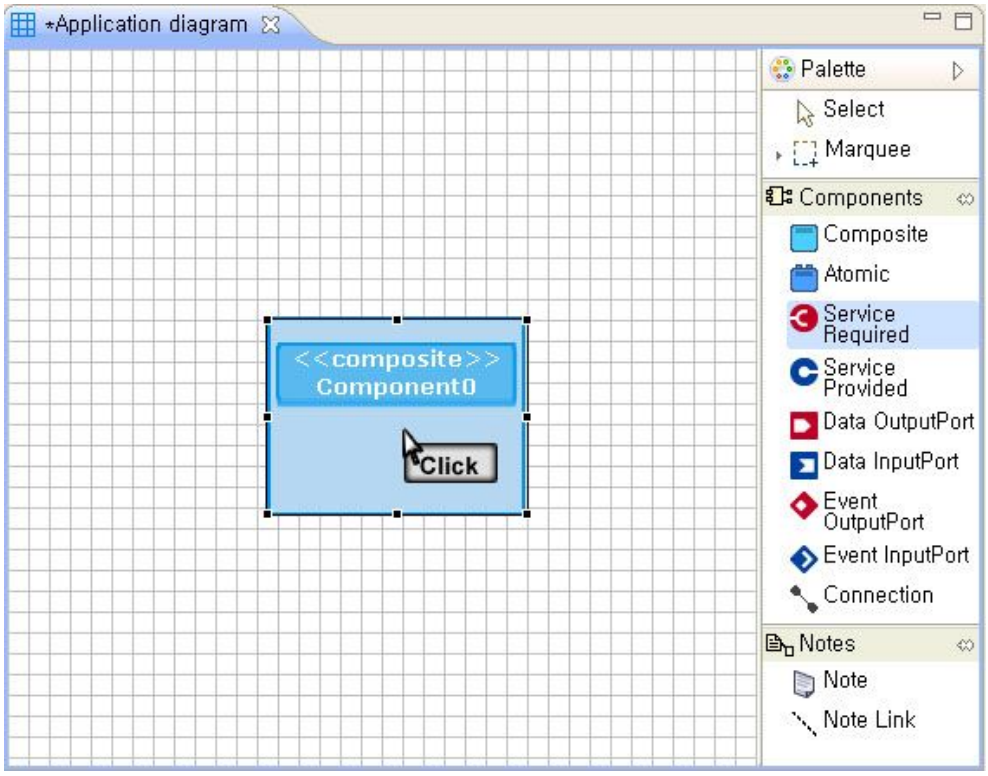

Figure 26. Create Composite's Ports 2

c. The corresponding port is added on the top left of the component.

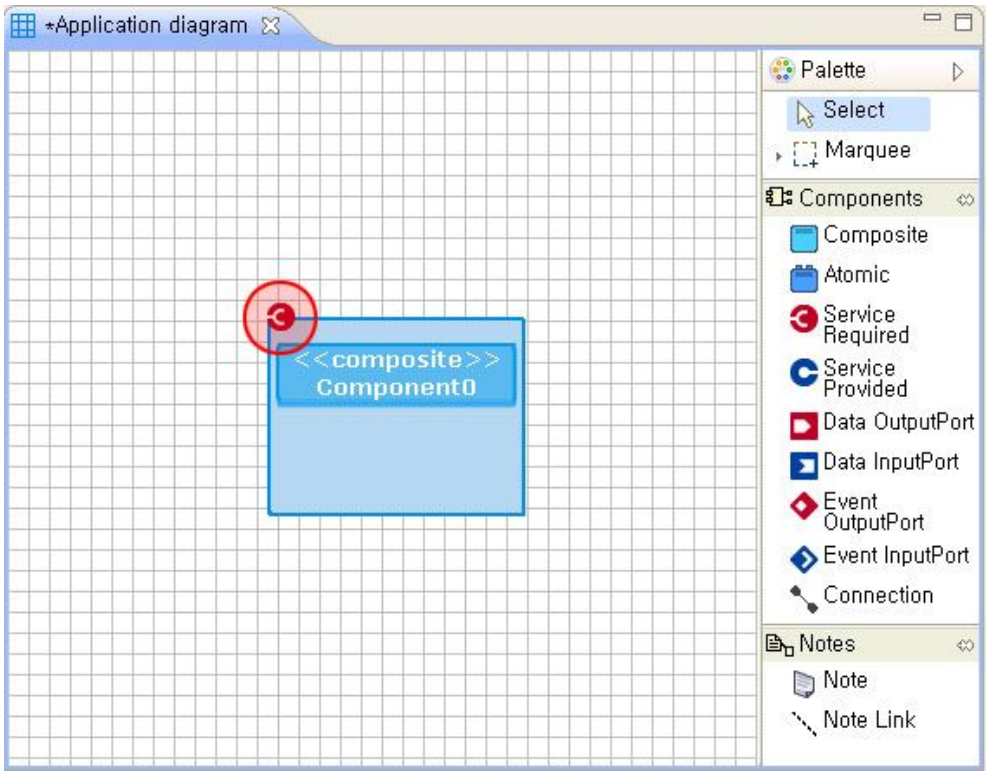

Figure 27. Create Composite's Ports 3

d. In a different way, a port can be added by drag-and-drop as adding an atomic component.

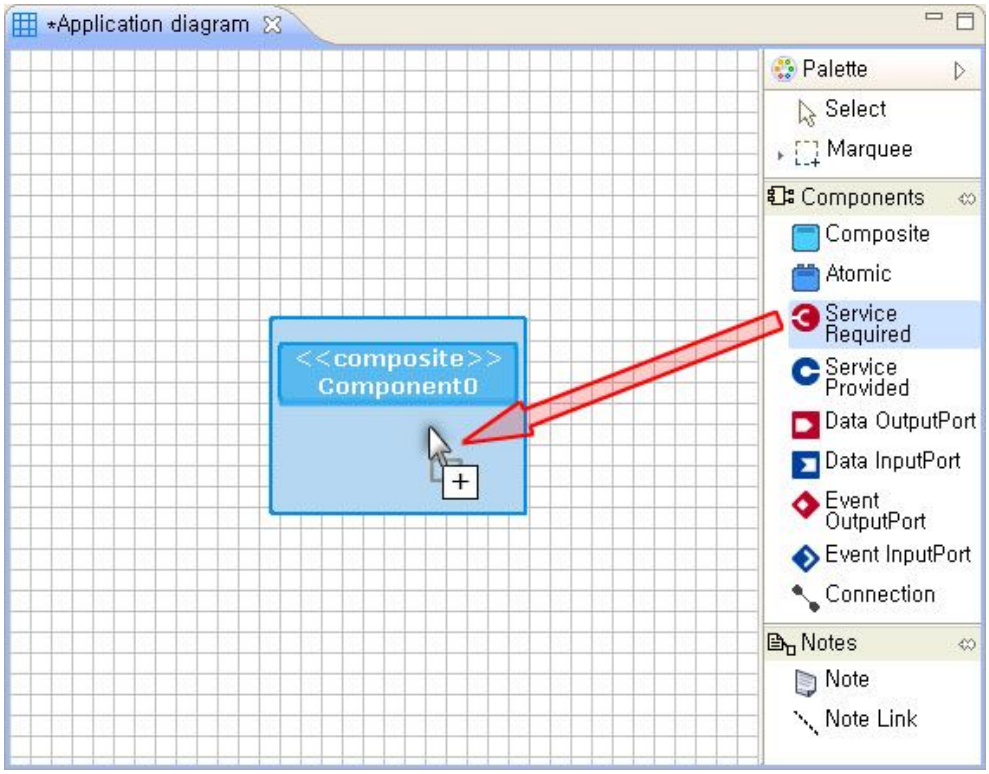

Figure 28. Create Composite's Ports 4

2.2.5. Create Atomic's Ports

a. Select a port to be added from the palette.

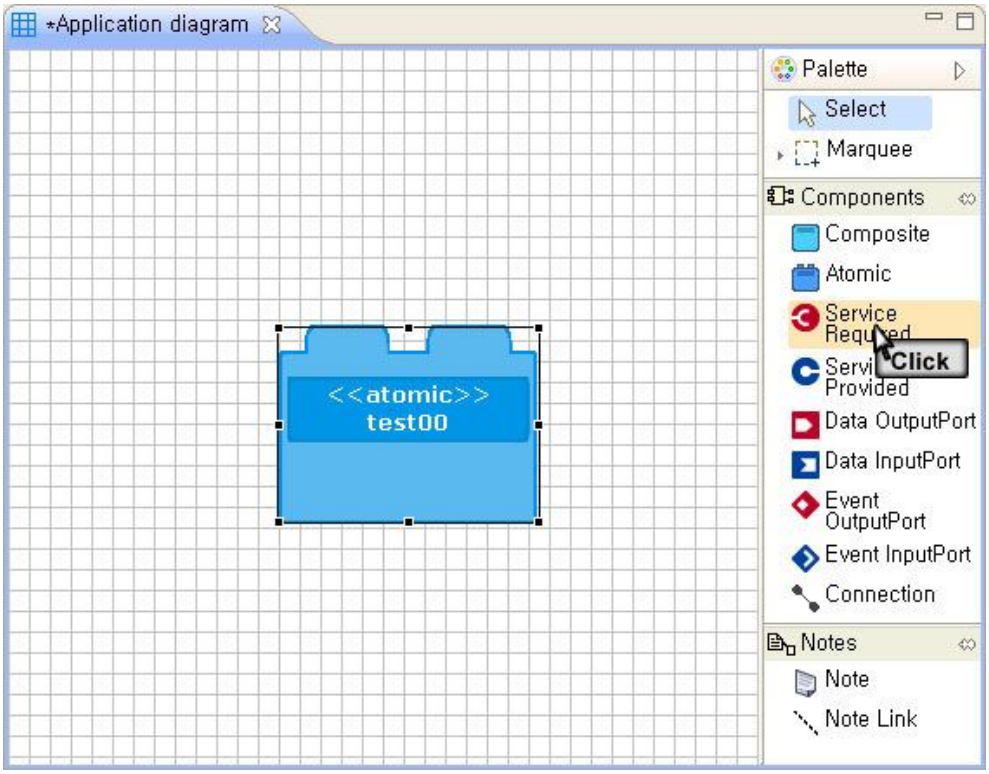

Figure 29. Create Atomic's Ports 1

b. Click the component, which you want to add a port, on the diagram edit window.

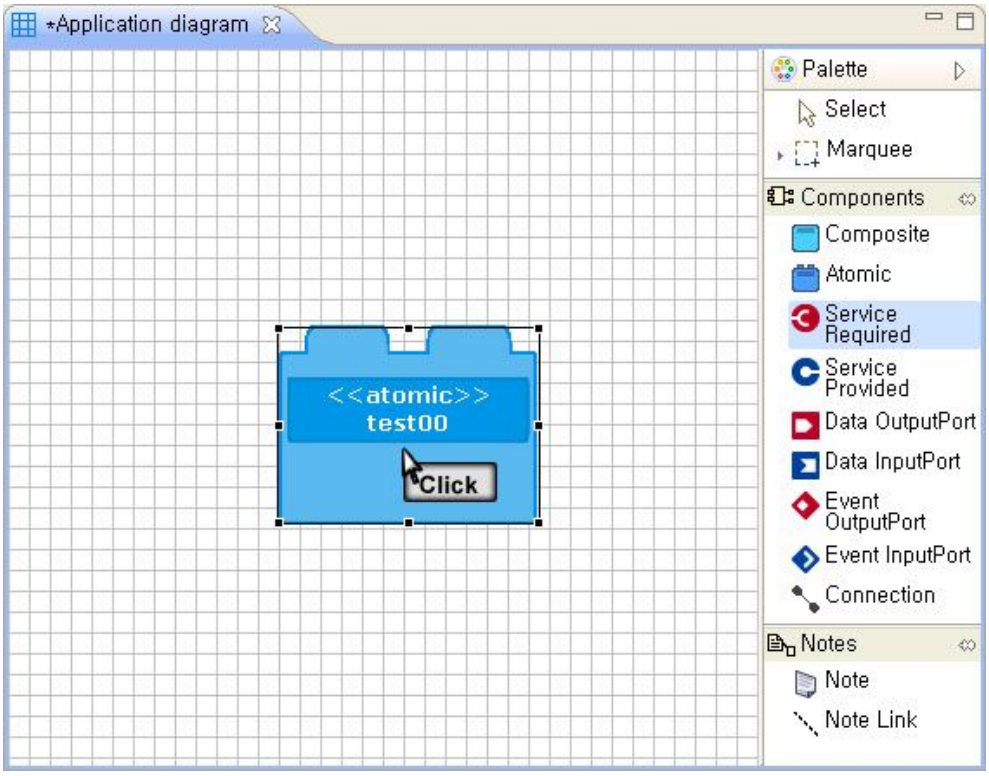

Figure 30. Create Atomic's Ports 2

c. For the Atomic, unlike the Composite, a dialog window is appeared to enter a name/properties etc. of the port.

A. Service Port

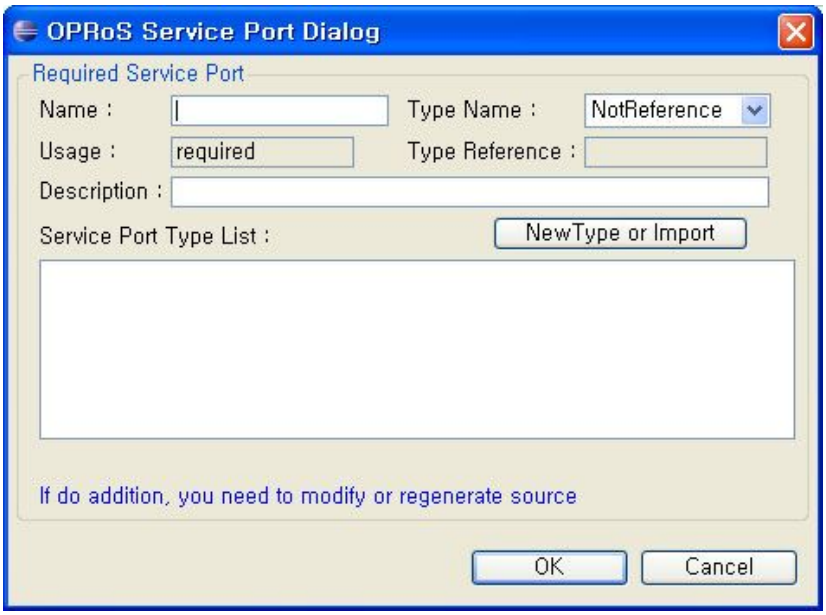

Figure 31. Service Port Dialog

- Name: Enter a name of the port.
- Type Name: Specify a type.

- Description: Enter description about the port.
- New Type or Import: Add a new type (refer to the Create Types.)
- Service Port Type List: Select a type created.

B. Data Port

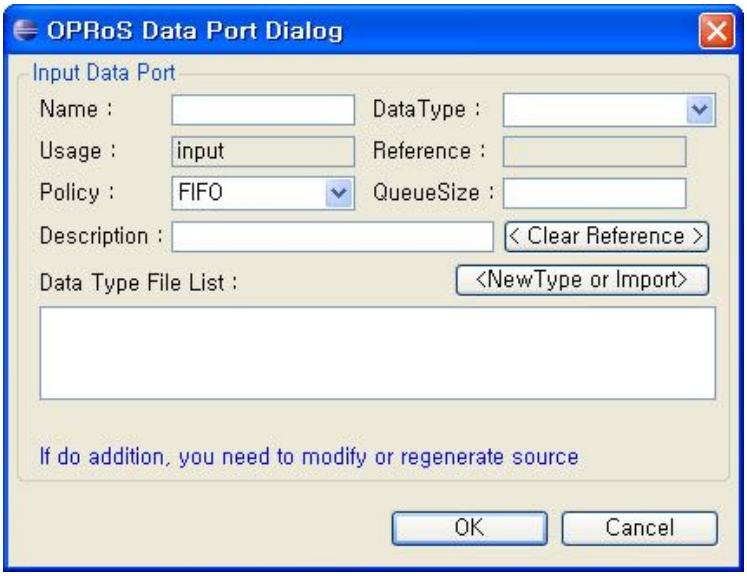

Figure 32. Data Port Dialog

- Name: Enter a name of the port.
- Data Type: Specify a type.
- Policy: Specify a policy.
- Queue Size: Specify a queue size.
- Description: Enter description about the port.
- Clear Reference: Initialize data types and references.
- New Type or Import: Add a new type (refer to the Create Types.)
- Data Type File List: Select a type created.

C. Event Port

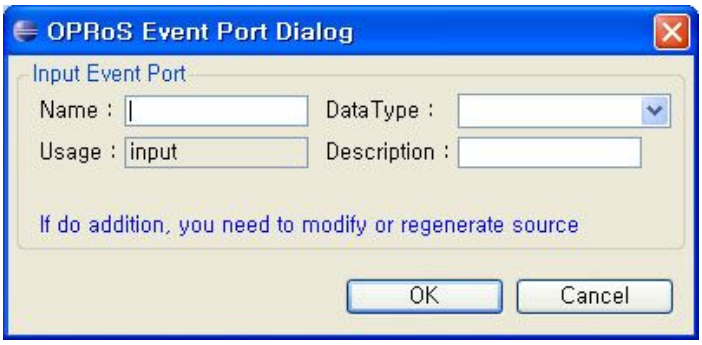

Figure 33. Event Port Dialog

- Name: Enter a name of the port.
- Data Type: Specify a type.
- Description: Enter description about the port.

d. When pressing the OK button to close the dialog, the corresponding port is added on the top left of the component.

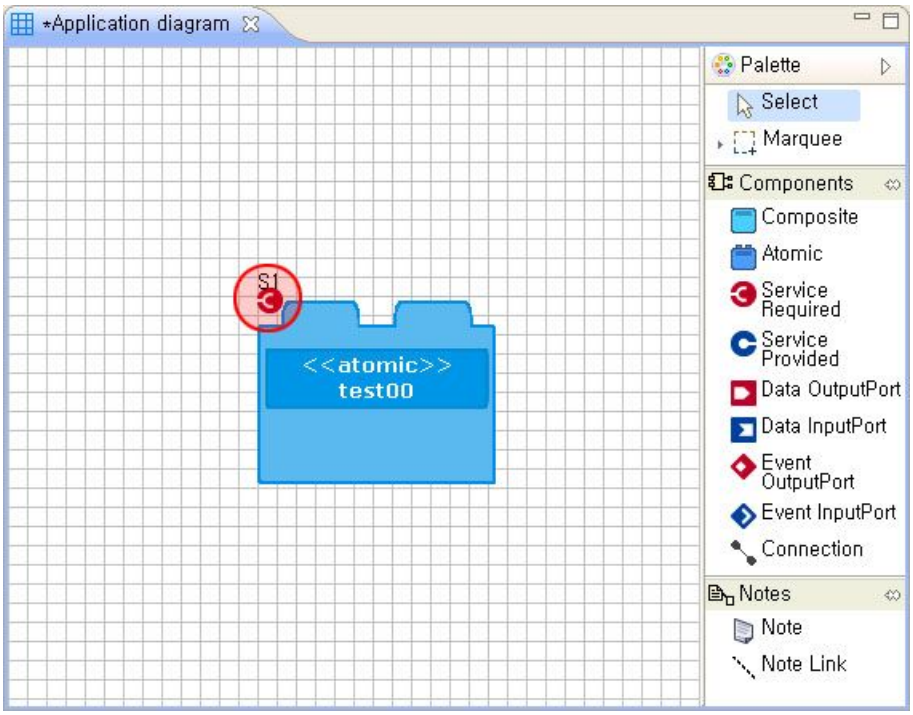

Figure 34. Create Atomic's Ports 3

e. A port can be added by drag-and-drop as adding the Composite's ports.

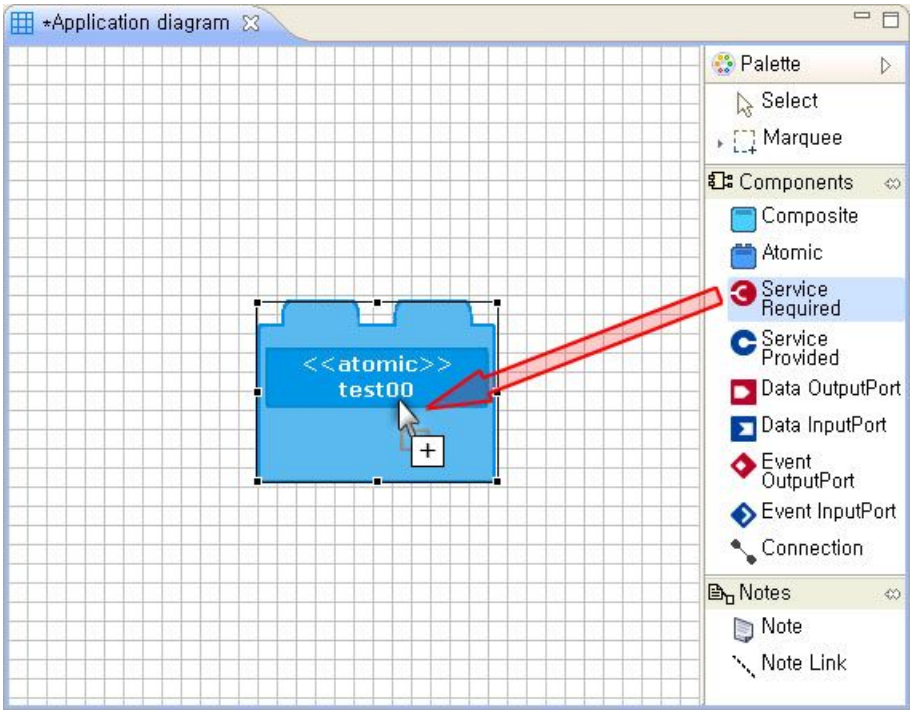

Figure 35. Create Atomic's Ports 4

2.2.6. Create Types

a. Press the New Type or Import button to open the Create window.

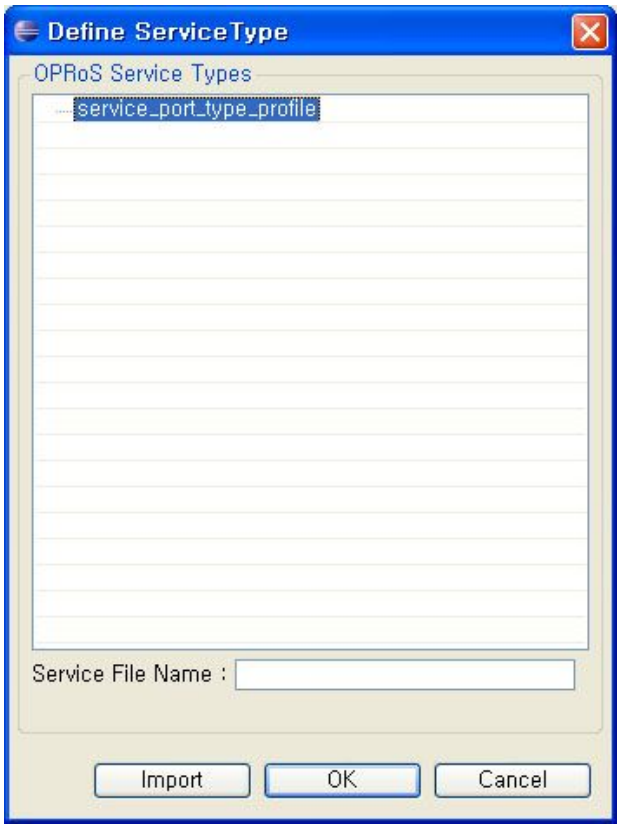

Figure 36. New Type or Import Dialog

- Service File Name: A name of the xml file to store the created type.
- Import: Fetches the type xml file from the outside.

b. Click the right mouse button on the Service_port_type_profile to open the popup menu.

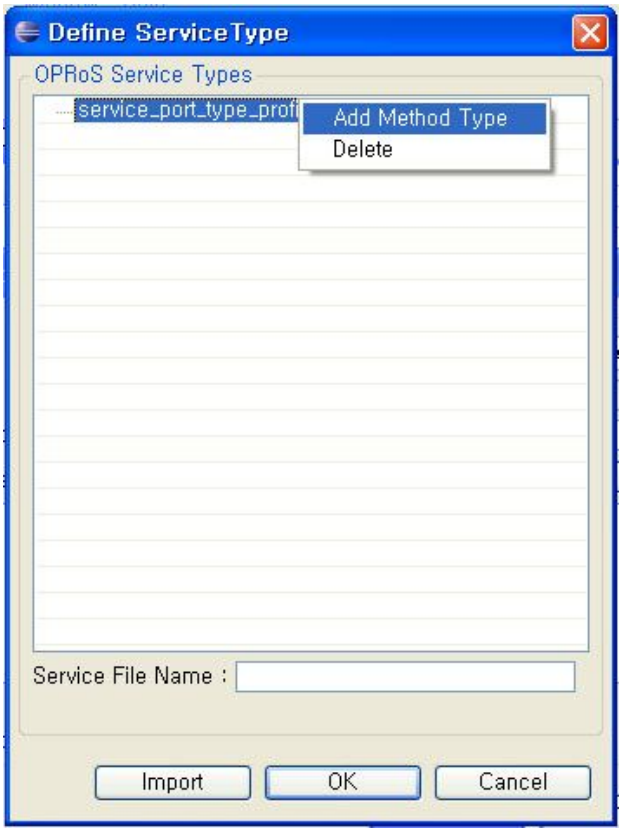

Figure 37. Add Method Type 1

c. Select the Add Method Type, and enter a type name.

Figure 38. Add Method Type 2

d. When pressing the tree expansion button on the left of the Service_port_type_profile, you can see the created type.

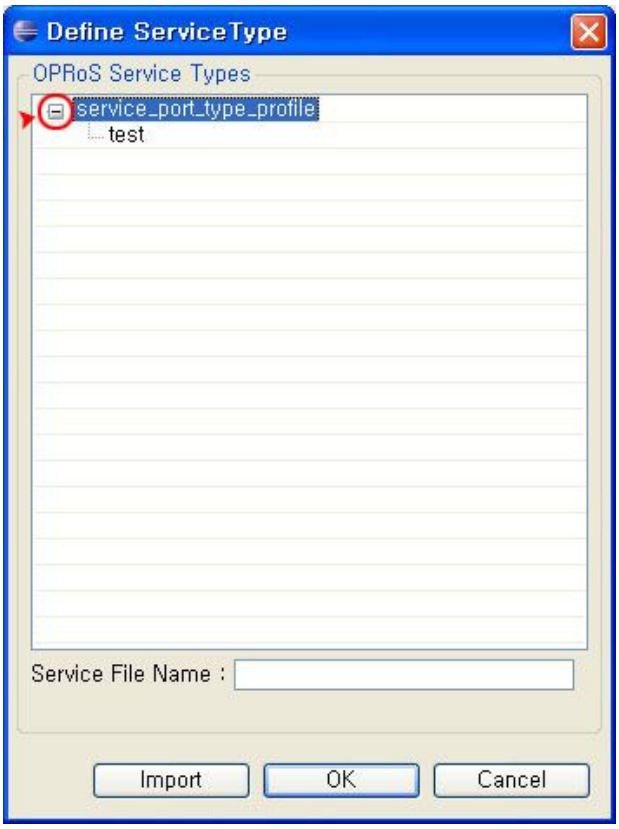

Figure 39. Add Method Type 3

e. After selecting the created type and opening the popup menu, select the Add Method to open a window to create methods.

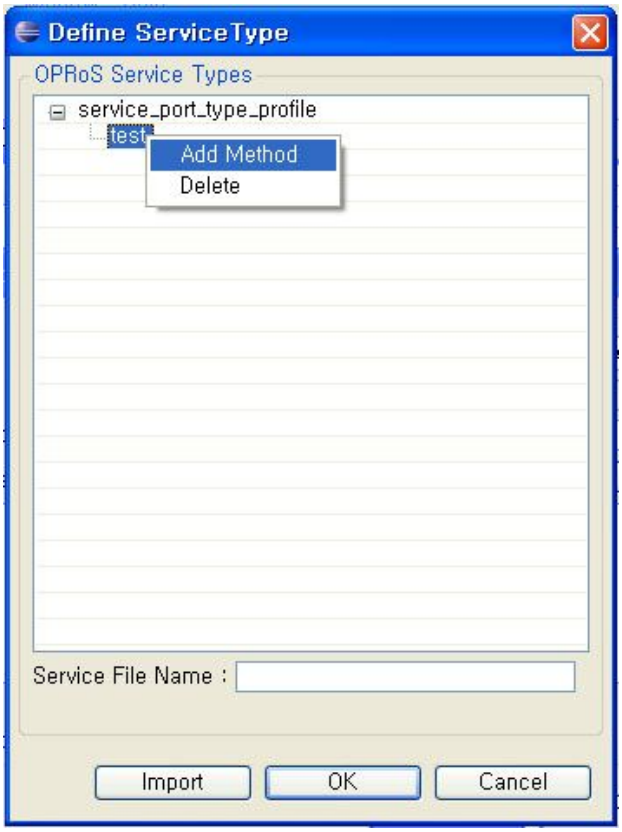

Figure 40. Add Method 1

f. Enter the Service Name, Return Data Type and Call Type, and create the method.

Figure 41. Add Method 2

- Service Name: Enter a name of the method to create.
- Return Data Type:
- Call Type: Enter description about the port.

g. After selecting the created method, select the Add Method Parameter to open the Create Parameter window.

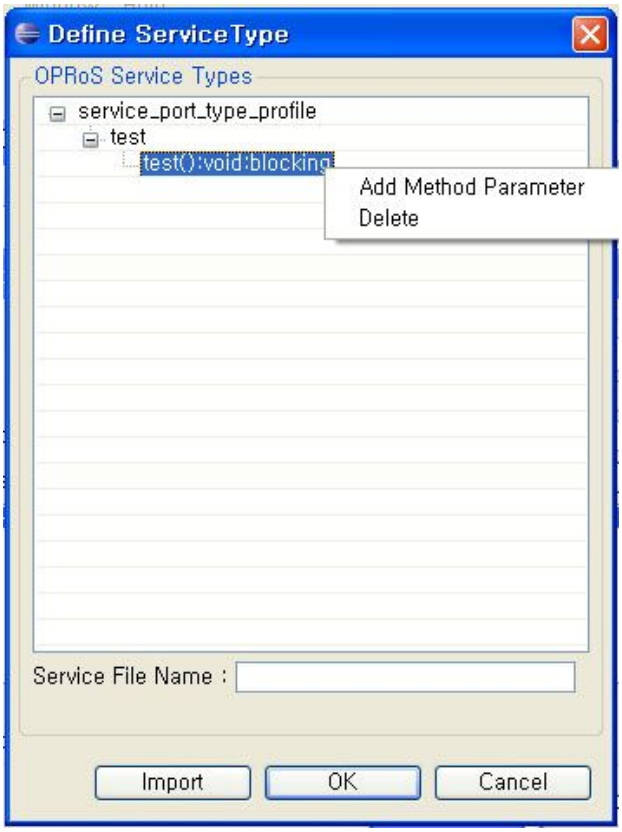

Figure 42. Add Method Parameter 1

h. Enter the Parameter Name and Data Type to create the parameter.

Figure 43. Add Method Parameter 2

i. When entering the created type name into the Service File Name and clicking the OK button, it is completed.

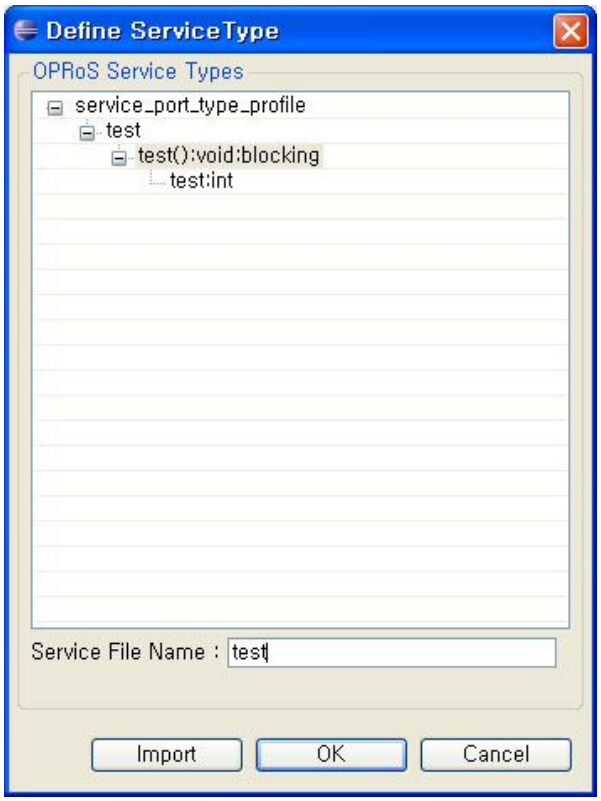

Figure 44. Completion of New Type or Import

j. When selecting the created type from the Service Port Type List and the type name from the Type Name, it can be used.

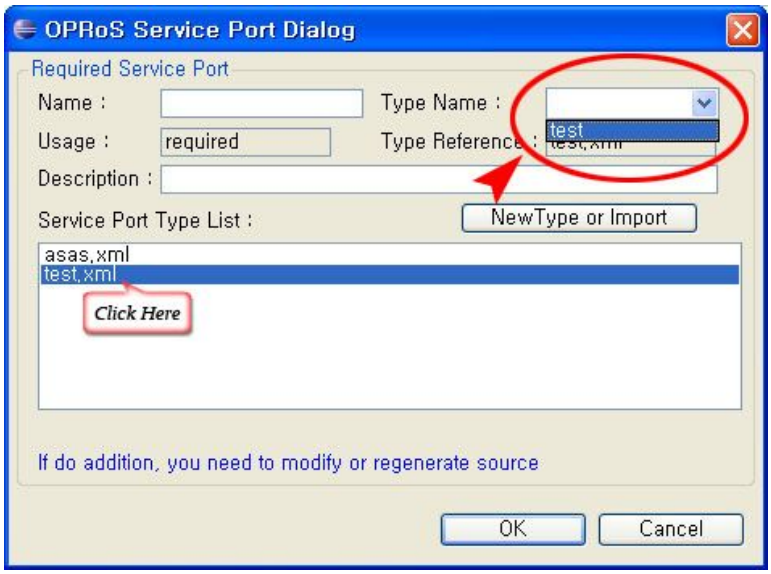

Figure 45. Type Select

2.2.7. Connect Connections

a. Select the Connection from the palette.

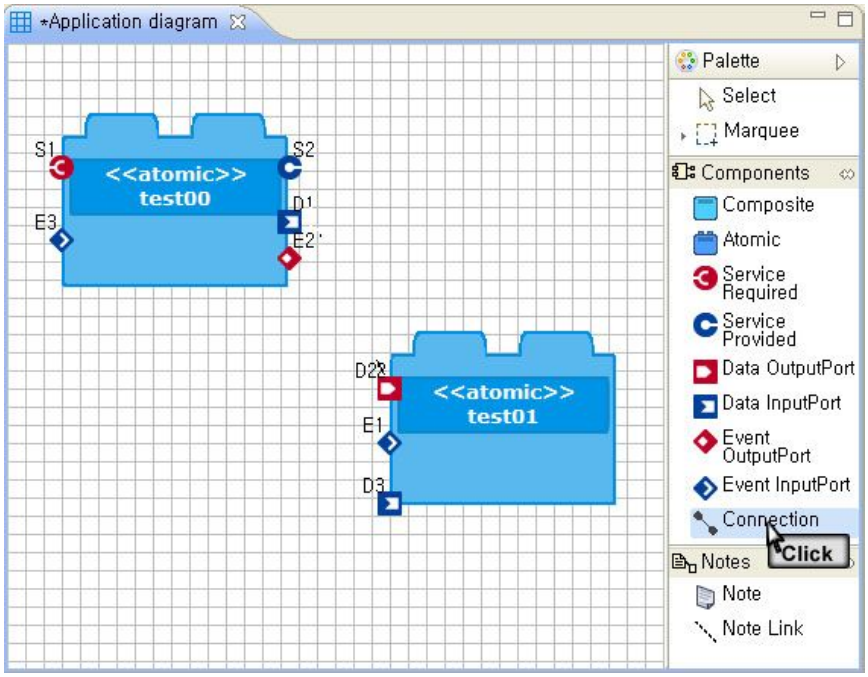

Figure 46. Connect Connections 1

b. After selecting a port of the source component on the diagram edit window, move the mouse to select a port of the target component.

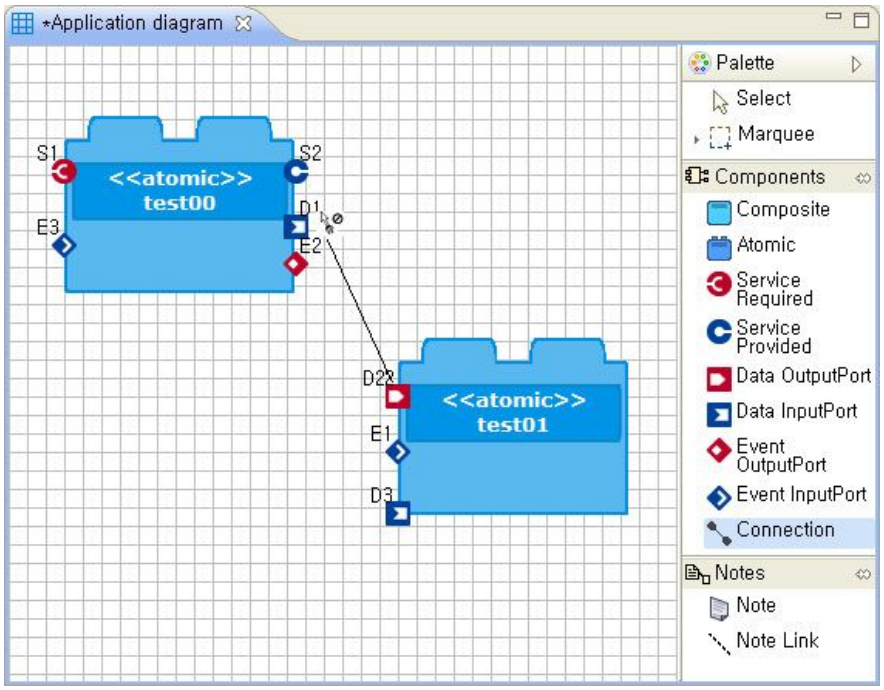

Figure 47. Connect Connections 2

c. Two ports are connected with a line.

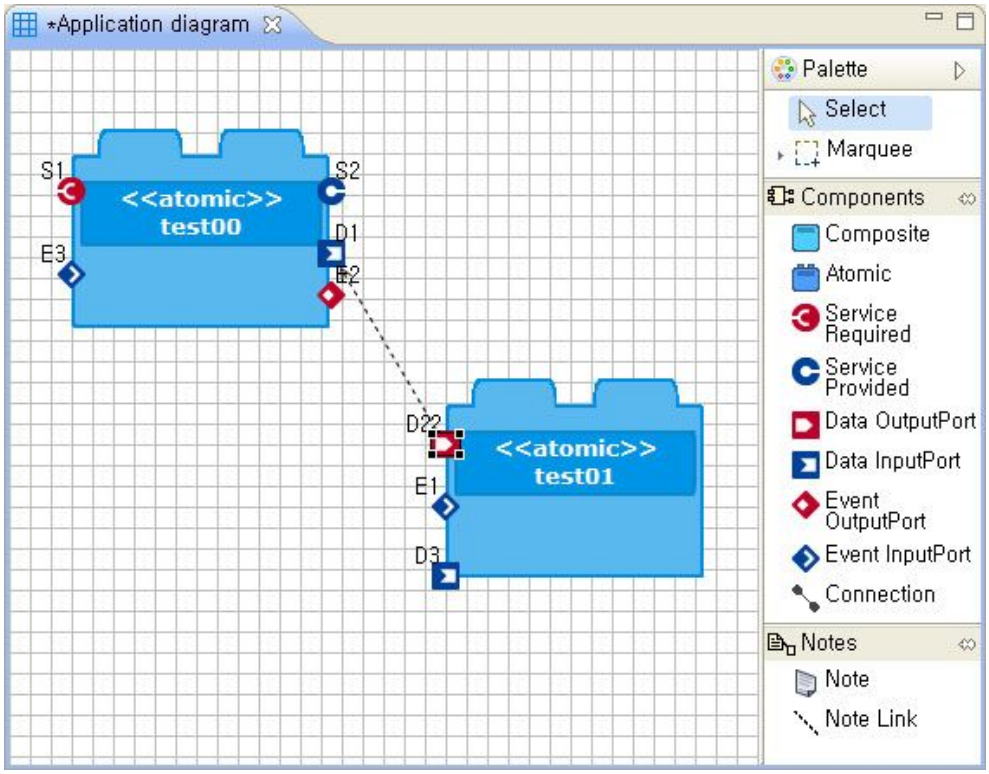

Figure 48. Connect Connections 3

※ Lines must be attached with the direction of connections as below.

Source	→	Target
Service Required		Service Provided
Data OutputPort		Data InputPort
Event OutputPort		Event InputPort

2.2.8. Connect Note Lines

a. Select the Note from the palette to create it on the diagram edit window.

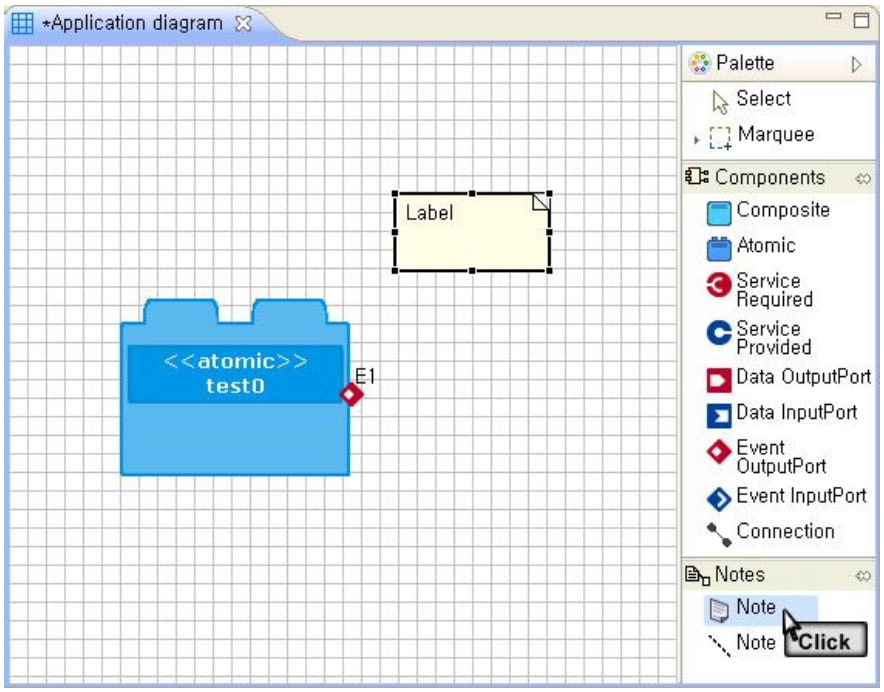

Figure 49. Connect Note Lines 1

b. After selecting the NoteLine, click the Note and a component (or a port) to connect them (the NoteLine is not distinguished between sources and targets unlike the Connection.)

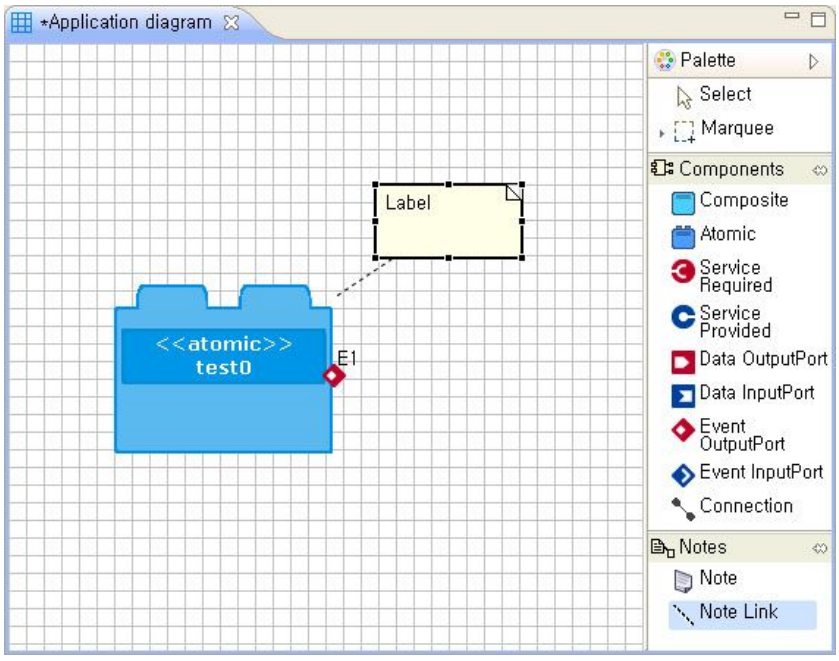

Figure 50. Connect Note Lines 2

2.2.9. Move Components

a. When dragging a component after selecting it by the mouse, it is moved.

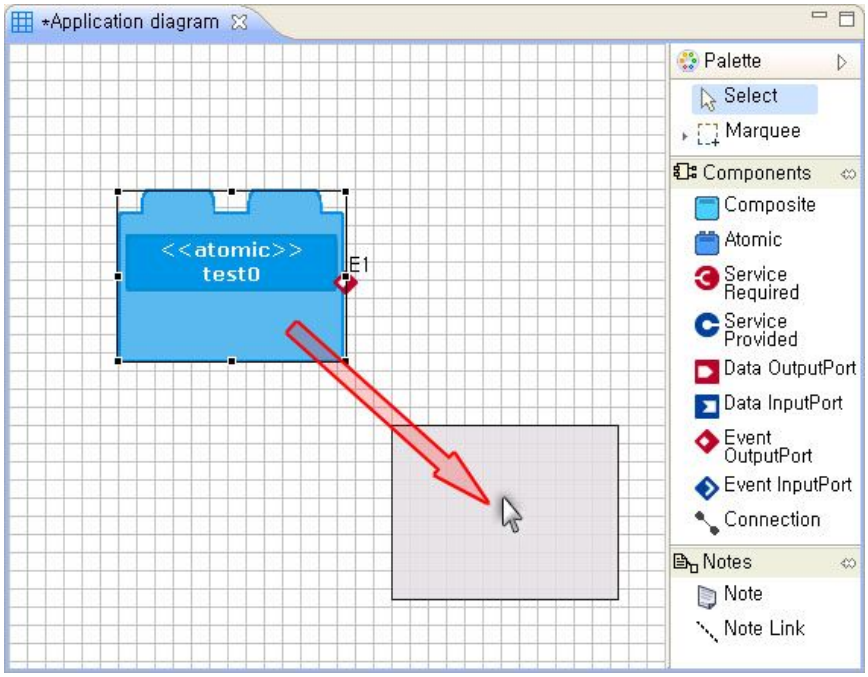

Figure 51. Move Components 1

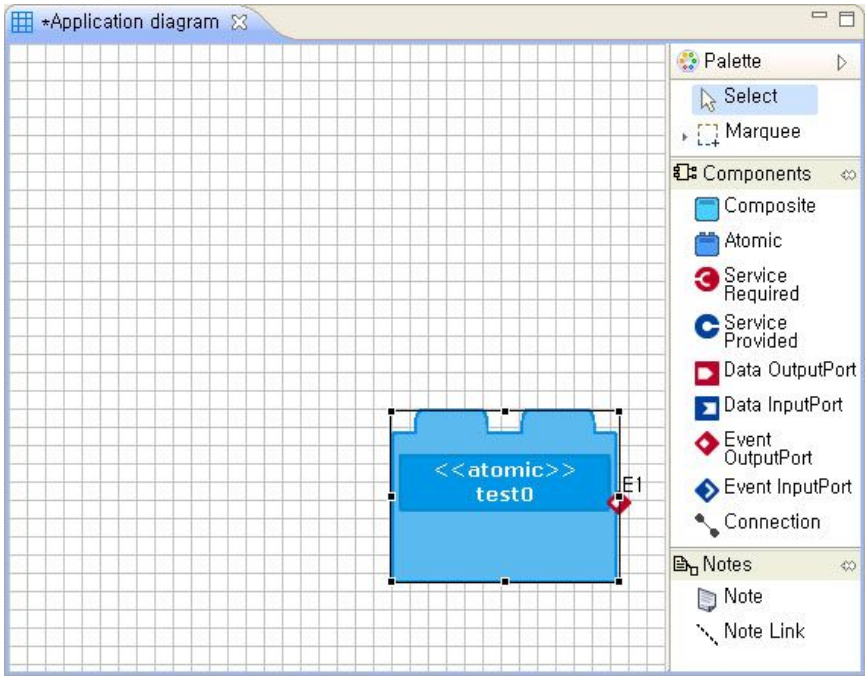

Figure 52. Move Components 2

b. If using the 'Alt + arrow keys' after selecting a component, it can be moved minutely.

2.2.10. Change Component Sizes

a. If dragging the handle bound of a component while clicking it, the component size can be changed.

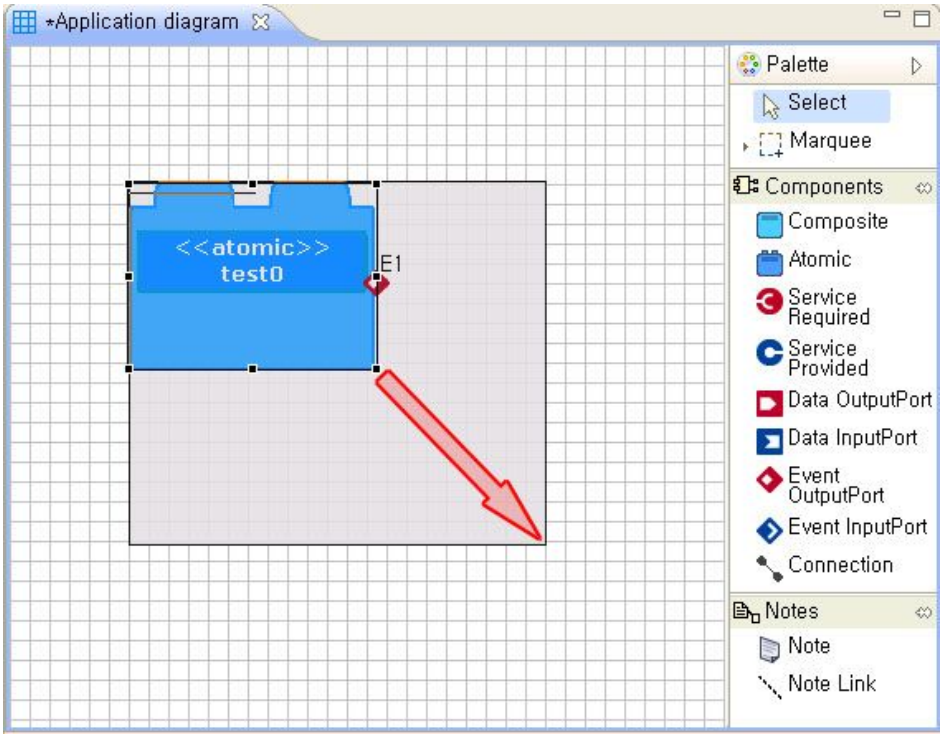

Figure 53. Change Component Sizes 1

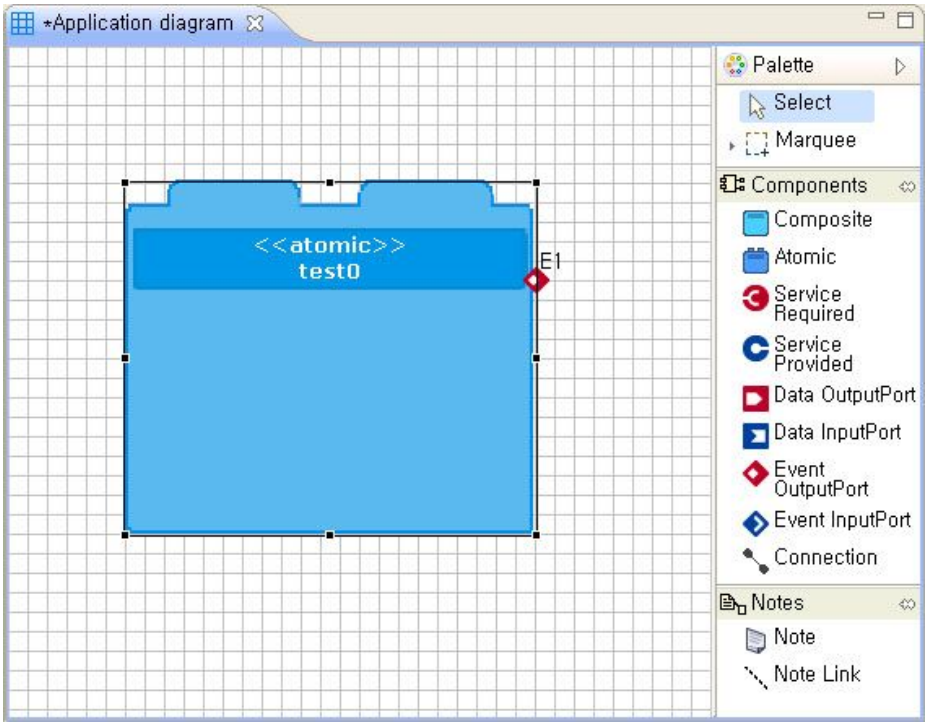

Figure 54. Change Component Sizes 2

2.2.11. Change Component Names

a. If double-clicking the name area of a component on the diagram edit window, a text edit window is created, and the name can be modified.

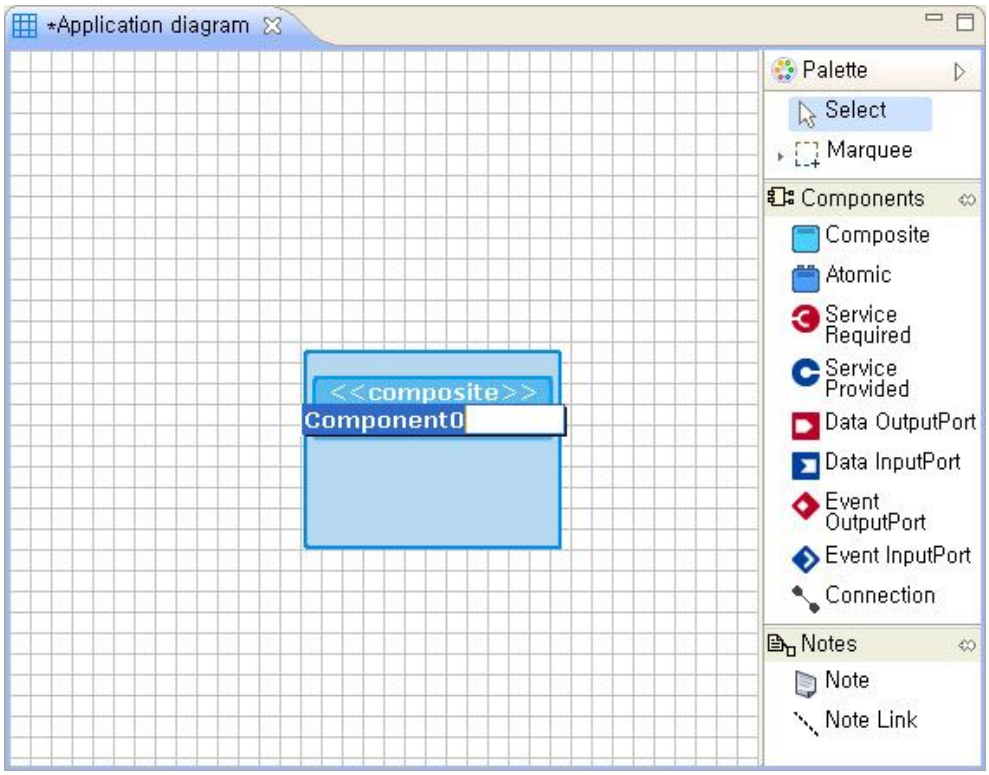

Figure 55. Change Component Names 1

b. After selecting a component, the name can be modified through the name property on the properties window.

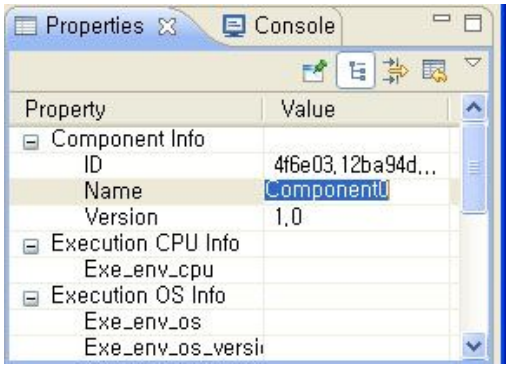

Figure 56. Change Component Names 2

2.2.12. Move Ports

a. When dragging a port, which you want to move, after selecting it, the port will be moved along the component's border.

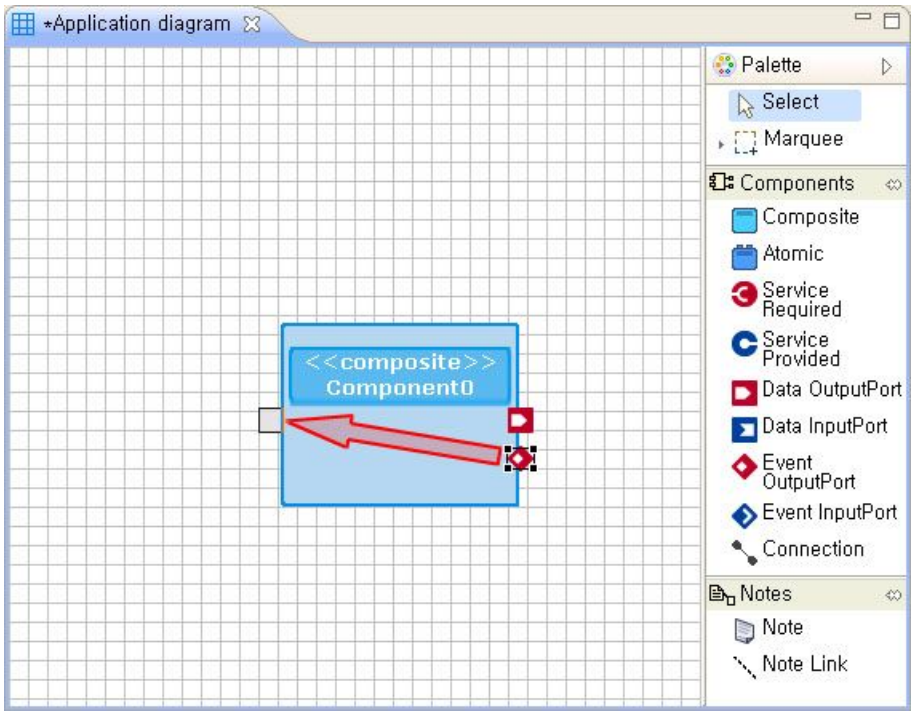

Figure 57. Move Ports 1

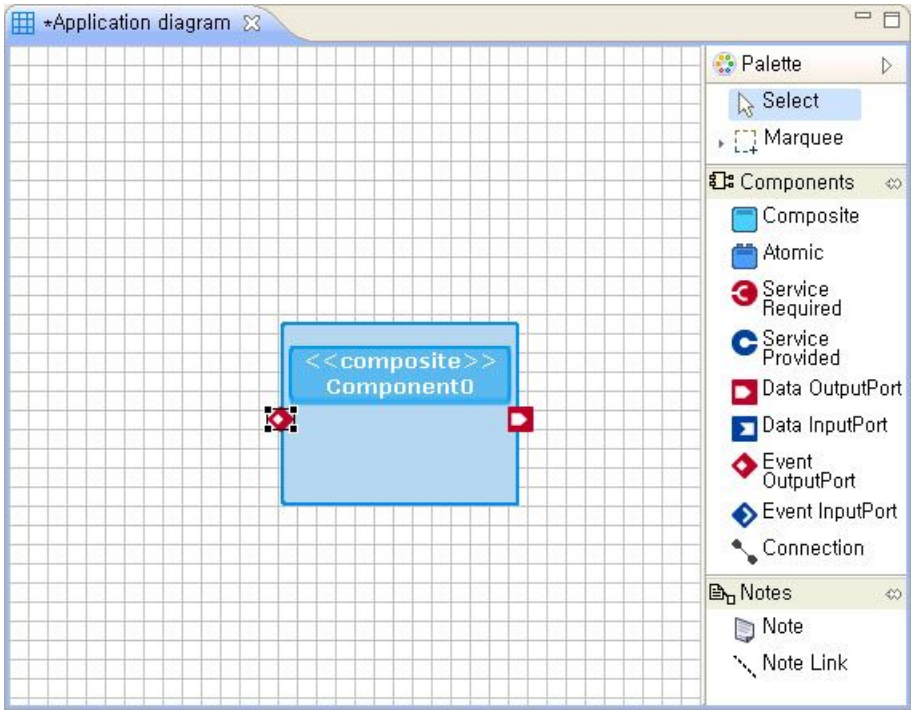

Figure 58. Move Ports 2

2.2.13. Component Popup Menu

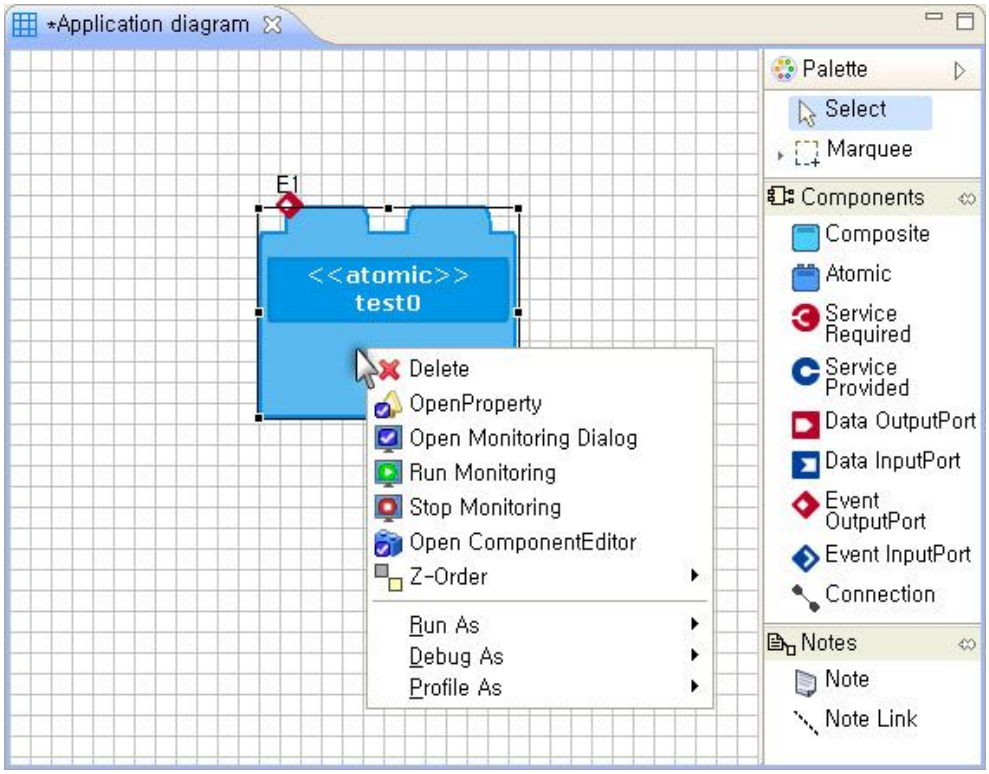

Figure 59. Component Popup Menu

- Delete: Deletes a component selected on the diagram edit window.

■ Open Property: Sets properties and monitoring variables of a component.

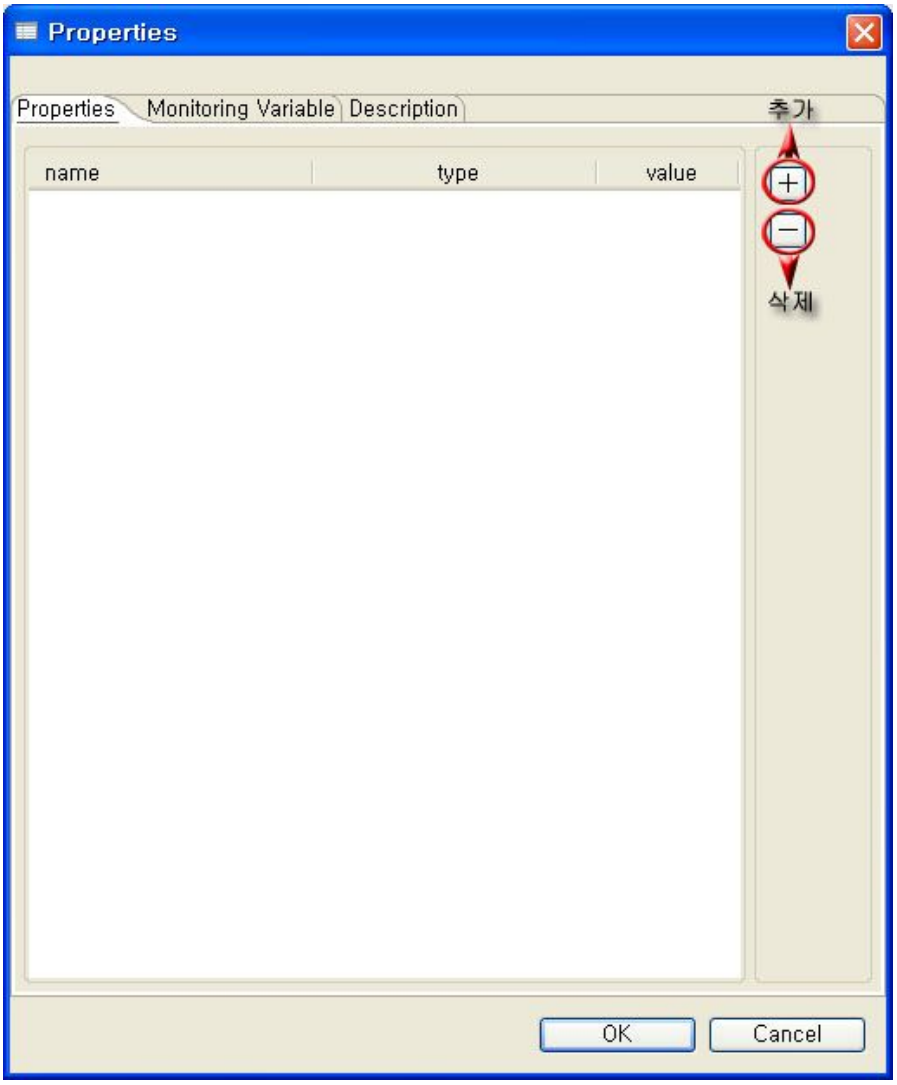

Figure 60. Open Property

◆ Properties: Adds/Deletes variables of a component.

◆ Monitoring Variable: Adds/Deletes the monitoring variables.

- Open Monitoring Dialog: Able to check changes of the monitoring variable's values.

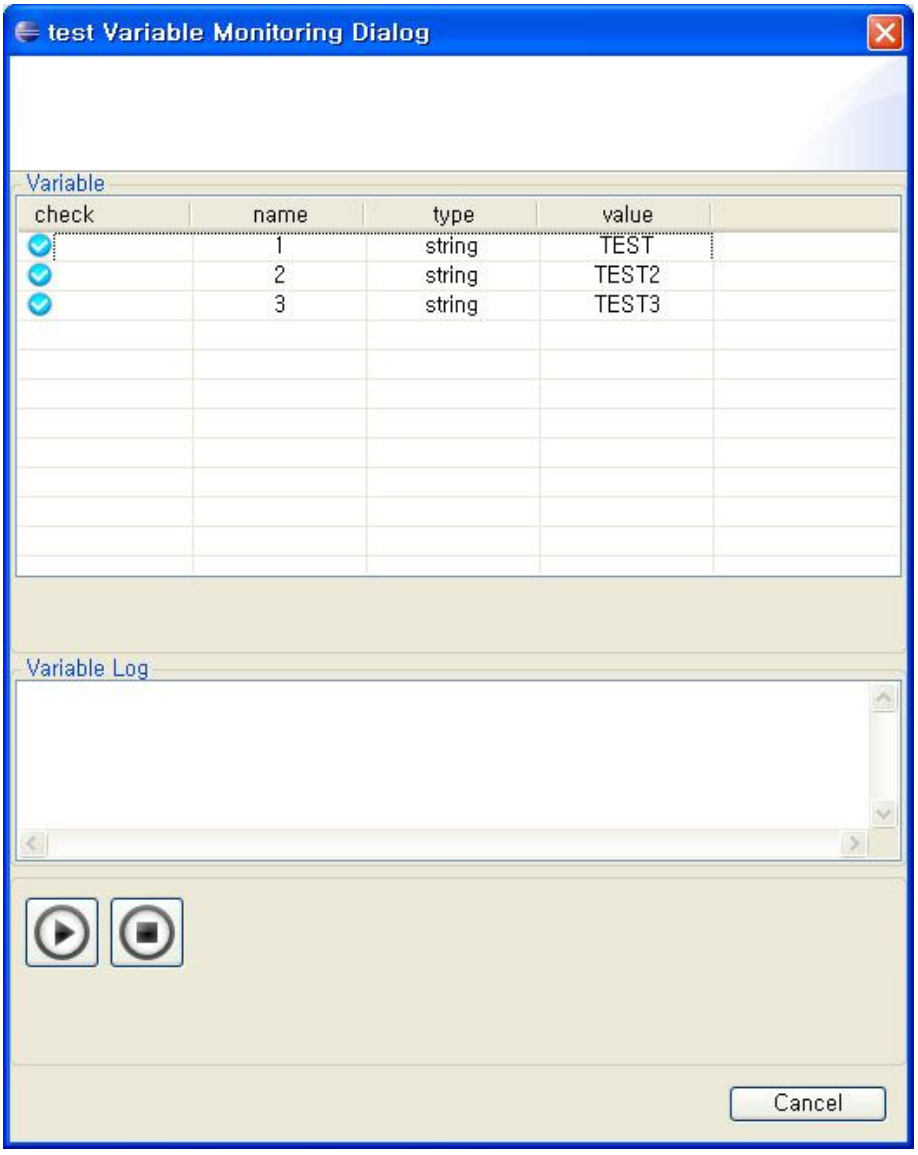

Figure 61. Open Monitoring Dialog

- Run Monitoring: Monitors the selected component.
- Stop Monitoring: Stops to monitor the selected component.
- Open ComponentEdtior: Popup menu applying only to atomic components, which calls a component editor to the corresponding component.
- Z-Order: Moves the selected component back and forth on the z-axis.
  - Bring Forward: Moves the selected component forward on the z-axis.
  - Send Back: Moves the selected component backward on the z-axis.

2.3. Examples

2.3.1. An Example of Developing Composite Components

a. Create a composite component below the package.

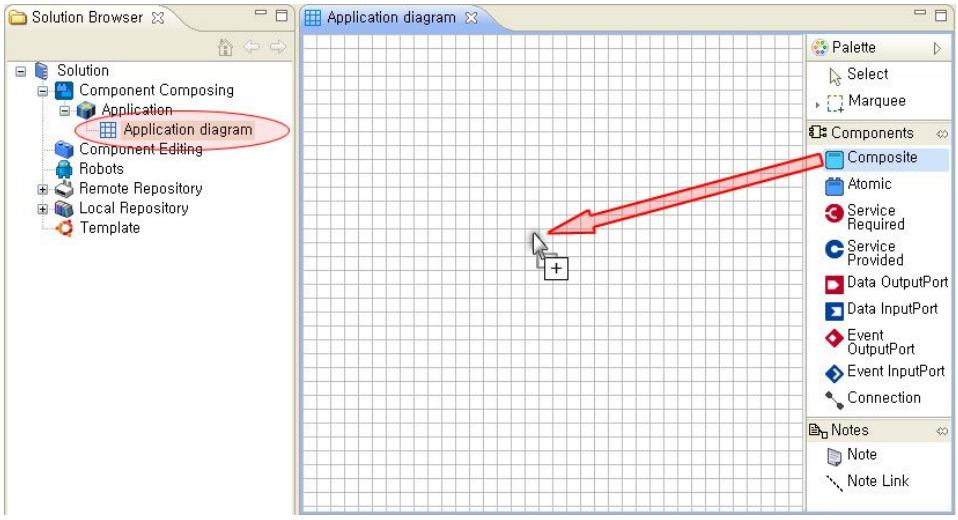

Figure 62. Example of Developing Composite Components 1

b. Open the internal diagram for editing the component (double-click the component's internal diagram in the solution browser, or use the Open Diagram from the popup menu of the component on the diagram edit window)

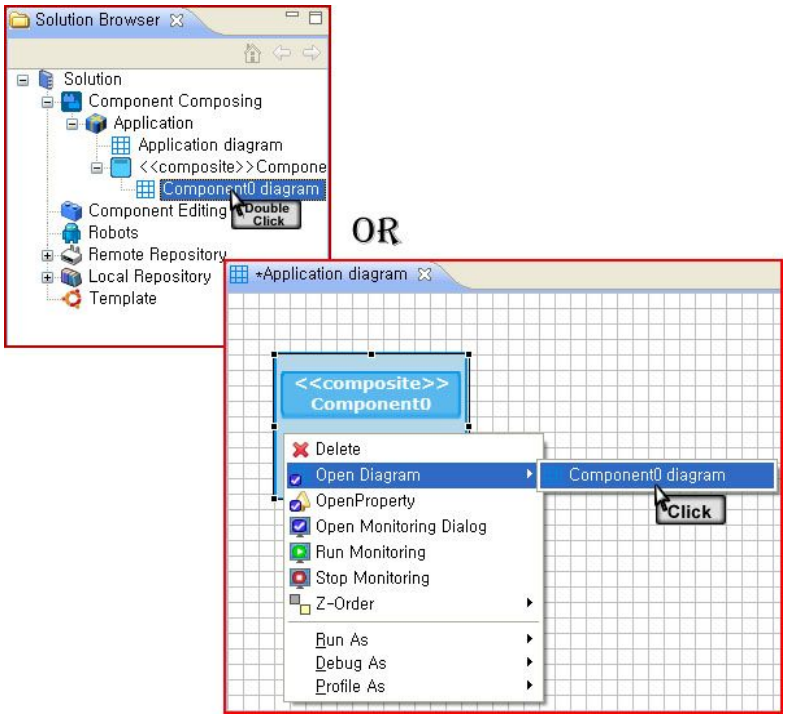

Figure 63. Example of Developing Composite Components 2

c. The internal diagram of the default component created is opened.

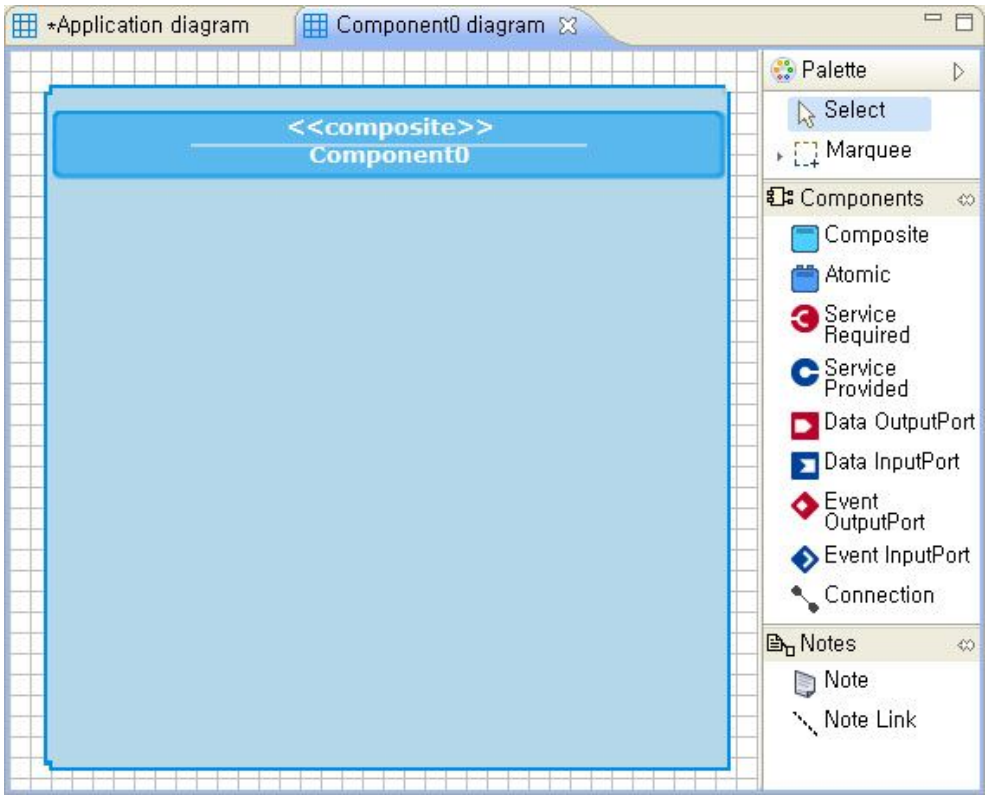

Figure 64. Example of Developing Composite Components 3

d. Drag-and-drop components in the Library or Repository to add them into the diagram edit window.

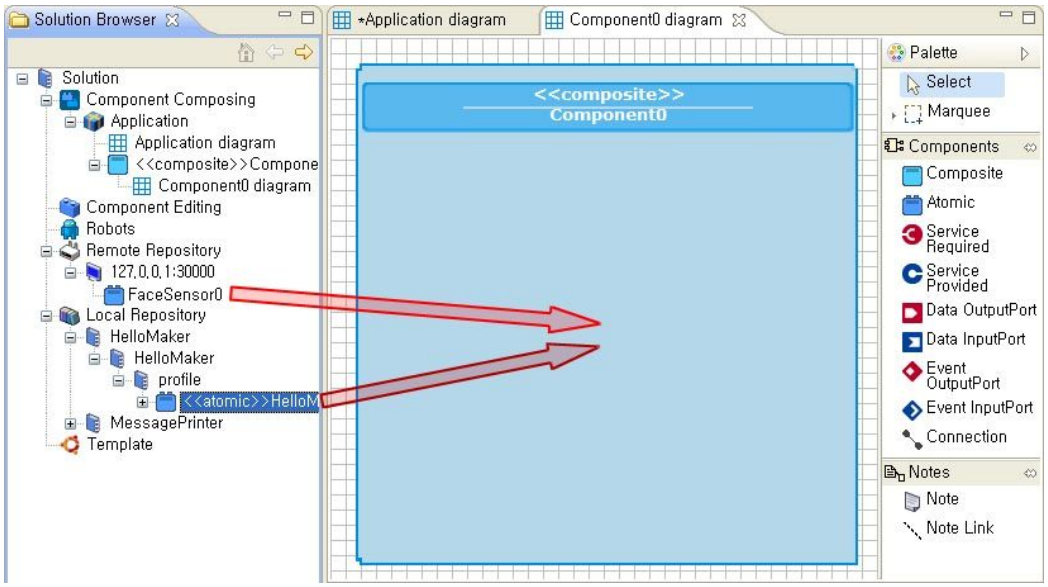

Figure 65. Example of Developing Composite Components 4

e. Connect the related ports each other using the connection. If necessary, add ports to the composite component, and connect them with the internal component.

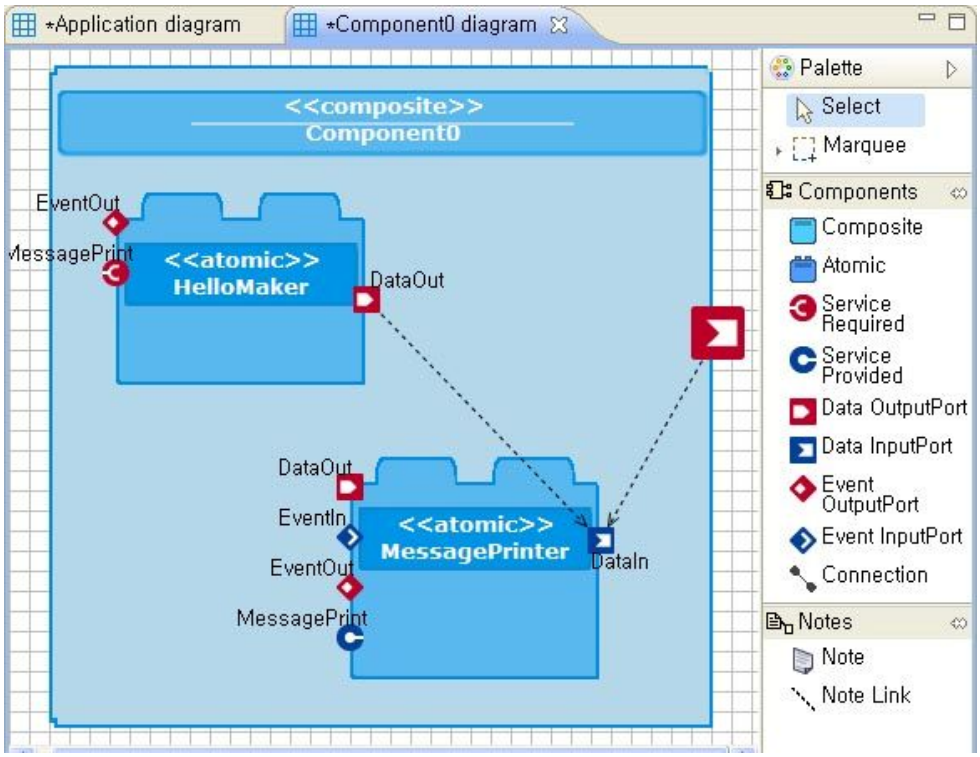

Figure 66. Example of Developing Composite Components 5

f. When adding a port to the composite component in the internal diagram, the port is added to be represented also on the reduced component as below.

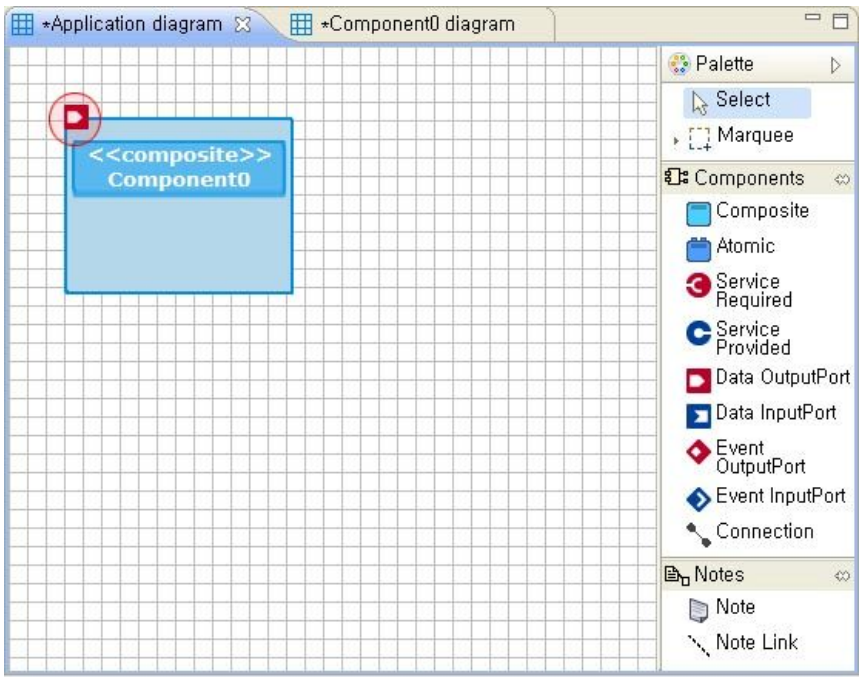

Figure 67. Example of Developing Composite Components 6

g. Execute the component's 'Popup Menu>Export Component' to save the component as a XML file.

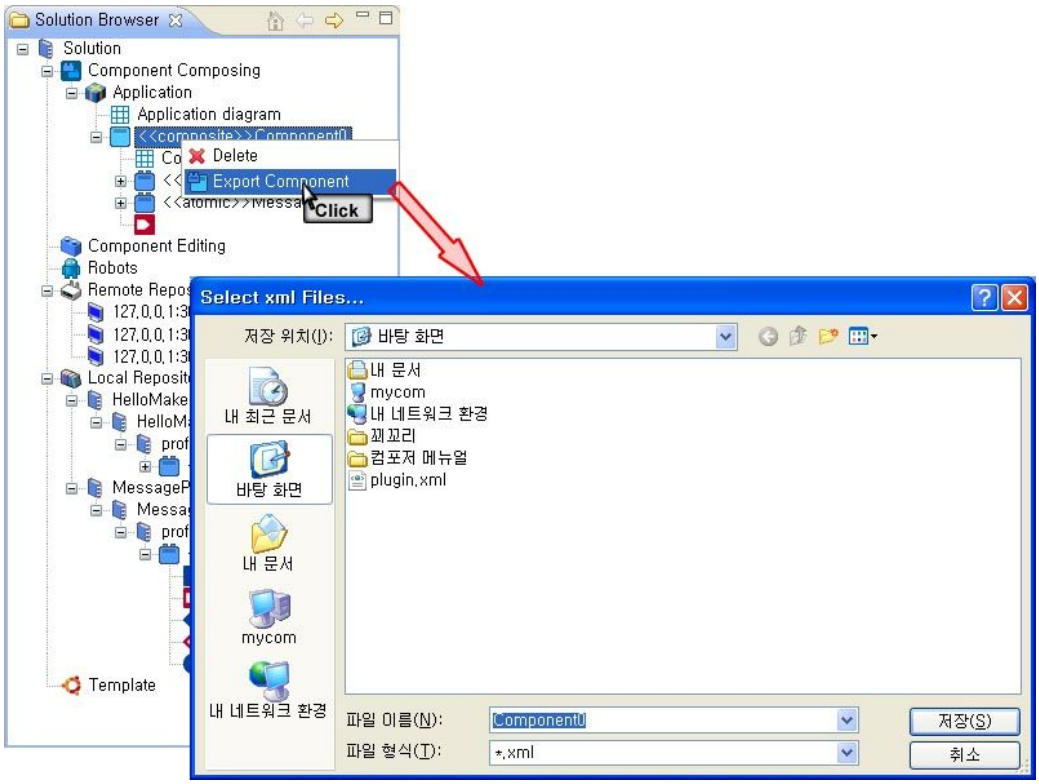

Figure 68. Example of Developing Composite Components 7

h. In the solution browser's library, the released component can be added to the component repository through the 'Popup Menu>Import Component' (※ Select a folder storing the released component files separately because all the lower folders are searched.)

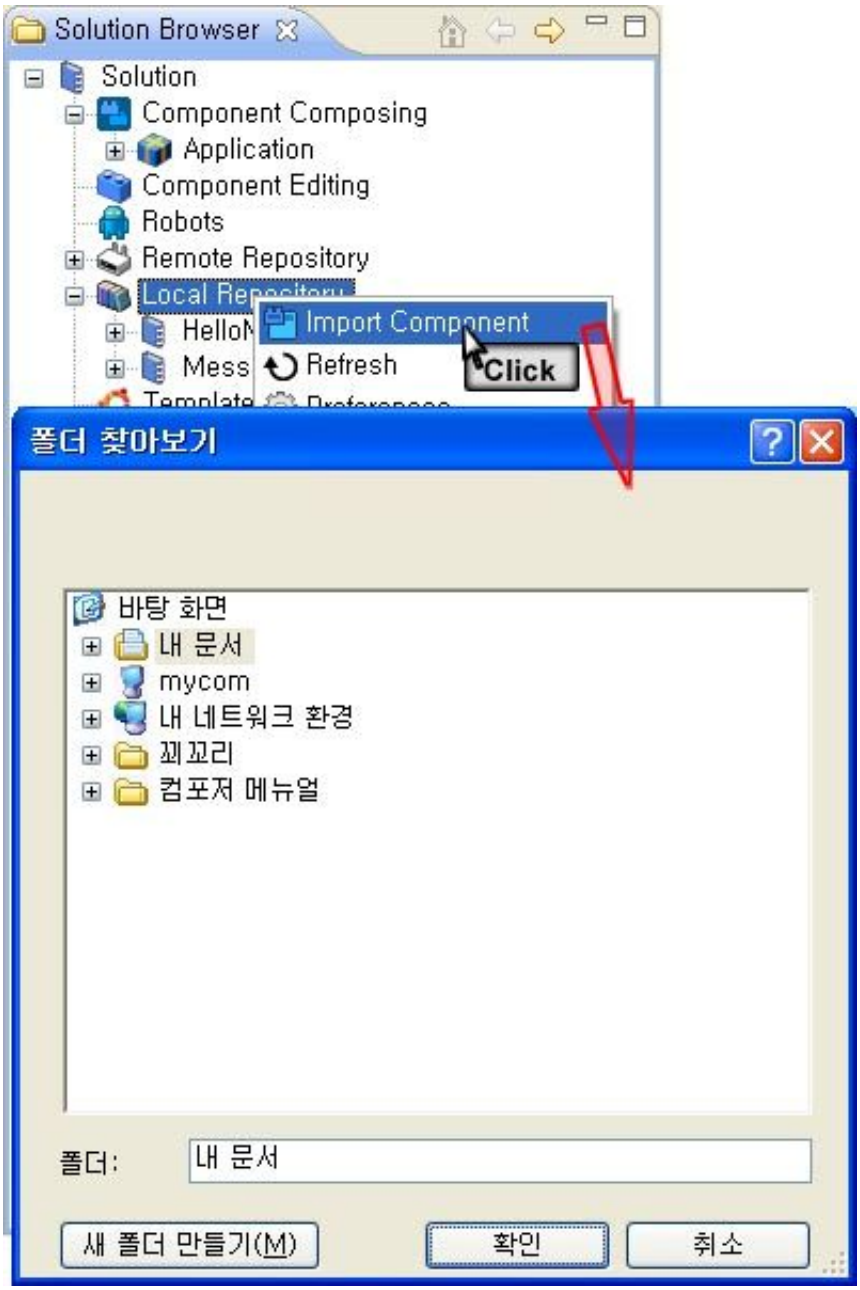

Figure 69. Example of Developing Composite Components 8

i. It can be seen that the component is added to the component repository.

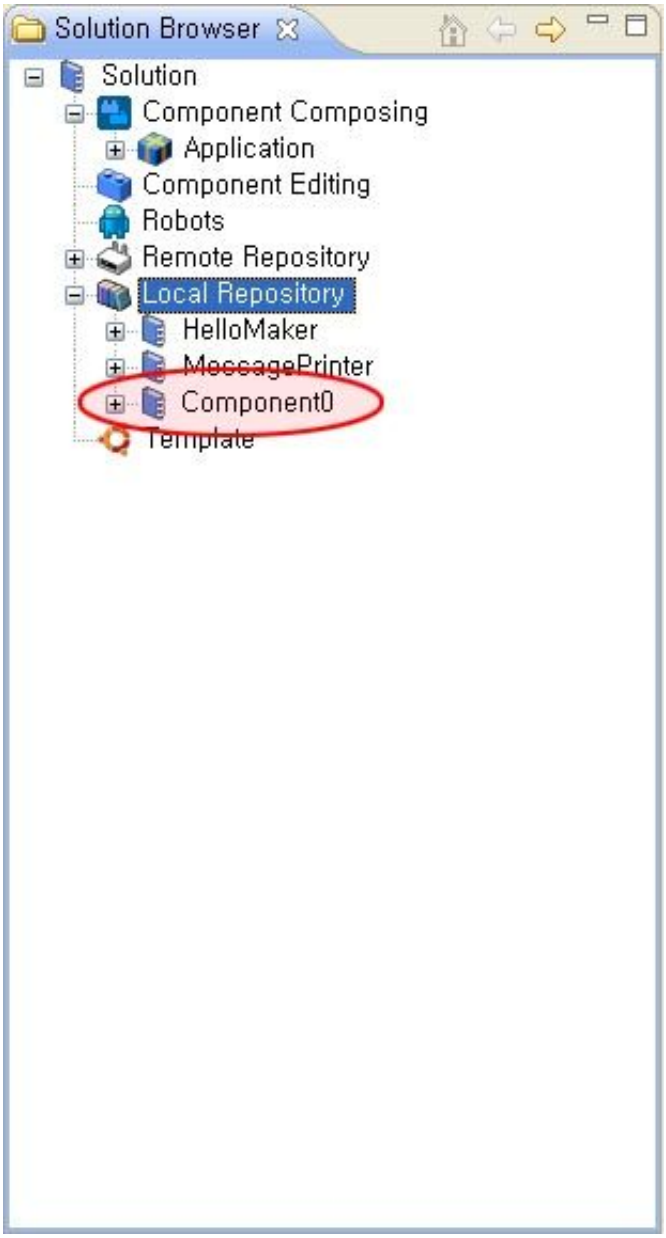

Figure 70. Example of Developing Composite Components 9

2.3.2. An Example of Developing Applications

a. Double-click the Application diagram in the Application package to open it.

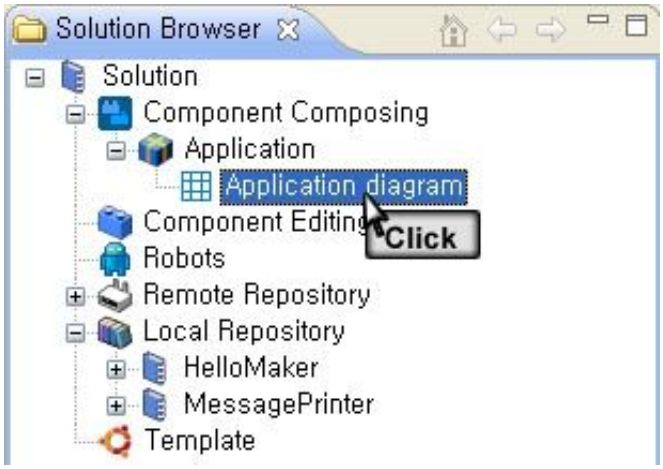

Figure 71. Example of Developing Applications 1

b. Add components to the diagram edit window. The atomic components in the library or repository can be added. Or composite components made in the application package can be added directly or by importing them to the library.

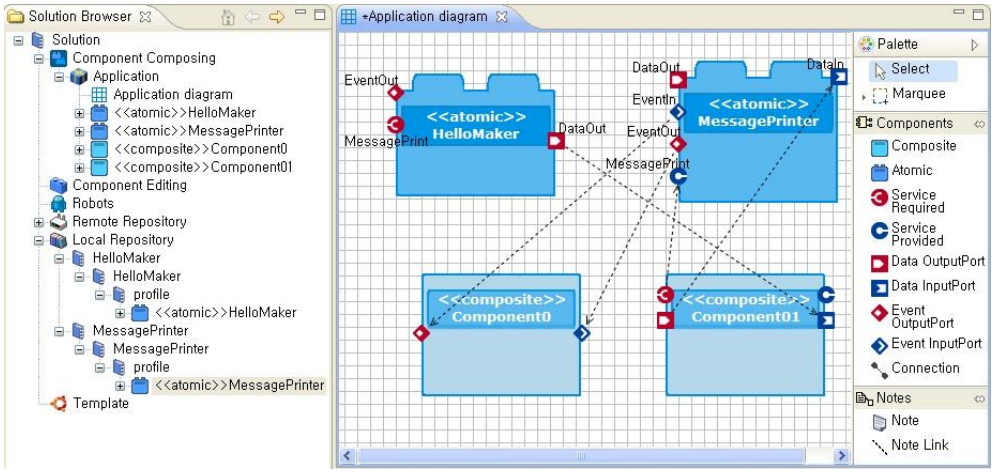

Figure 72. Example of Developing Applications 2

c. Applications can be released by executing the ‘Popup Menu>Export Application’ of the application package. Specify a location to be released and press the OK.

Figure 73. Example of Developing Applications 3

2.3.3. An Example of Interworking with Component Editors

c. After selecting the Atomic Component on the palette, drag it to the diagram or click a desired location.

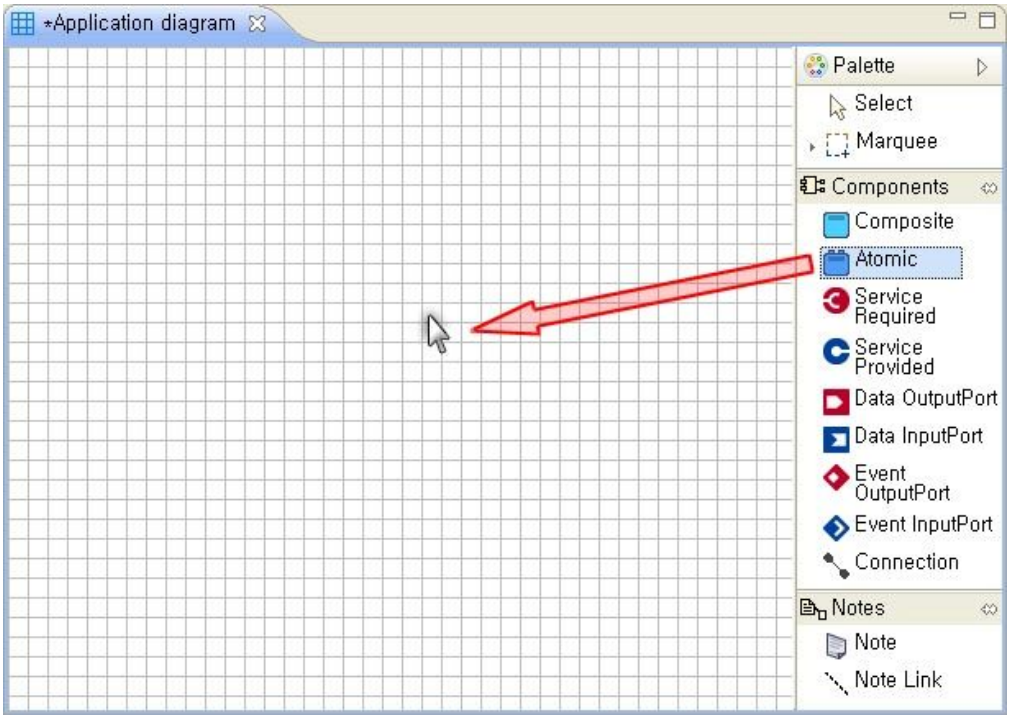

Figure 74. Example of Interworking with Component Editors 1

d. When clicking the OK after selecting a name of the component to be created and a development language (MinGW C++/ MS VC C++,) the component is created.

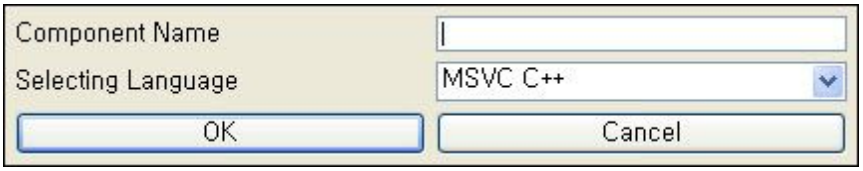

Figure 75. Example of Interworking with Component Editors 2

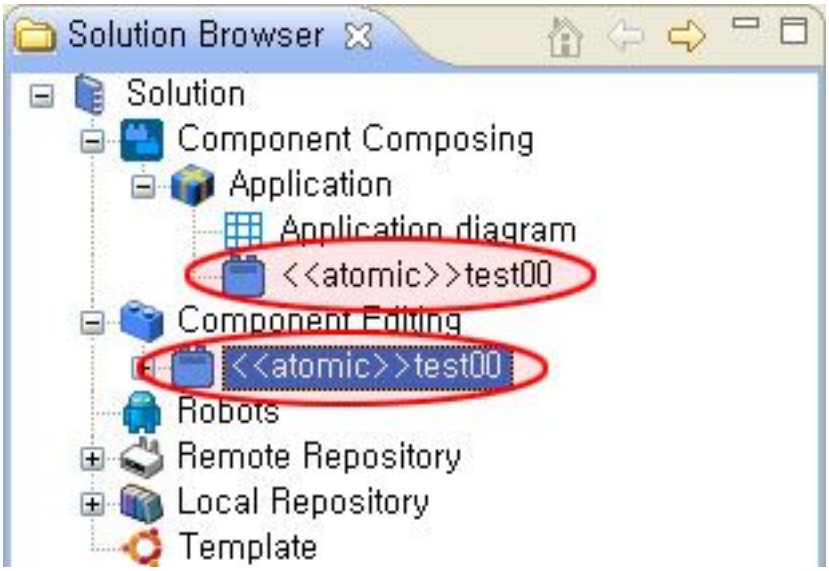

Figure 76. Example of Interworking with Component Editors 3

e. After adding ports required for the component, click the 'Open Component Edit' on the Atomic Component popup menu.

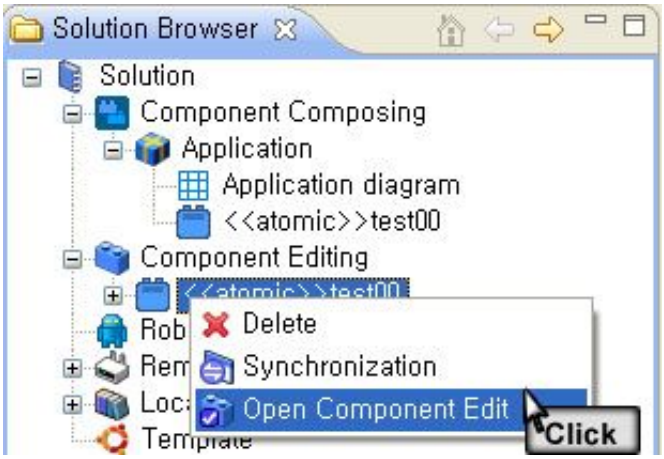

Figure 77. Example of Interworking with Component Editors 4

f. You can see that it is changed into the component editor as a component editor project for the corresponding component is created (refer to the component editor manual for how to use the editor.)

2.3.4. An Example of Deploy

g. Execute the Robots' 'Popup Menu>Add Robot' to add a new robot.

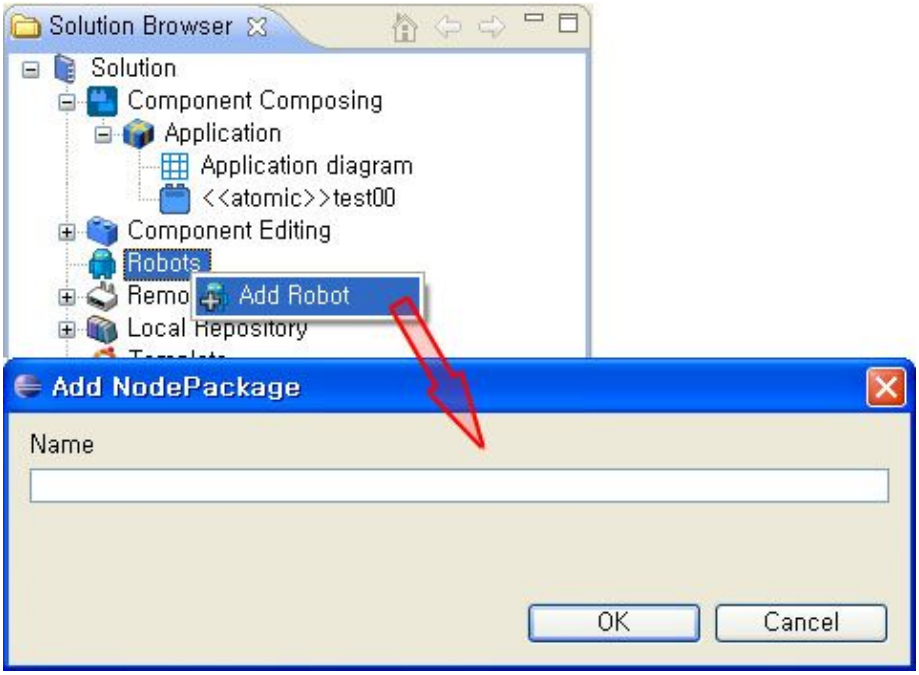

Figure 78. Example of Deploy 1

h. Execute the 'Popup Menu>Add Node' of the added robot to add a new node.

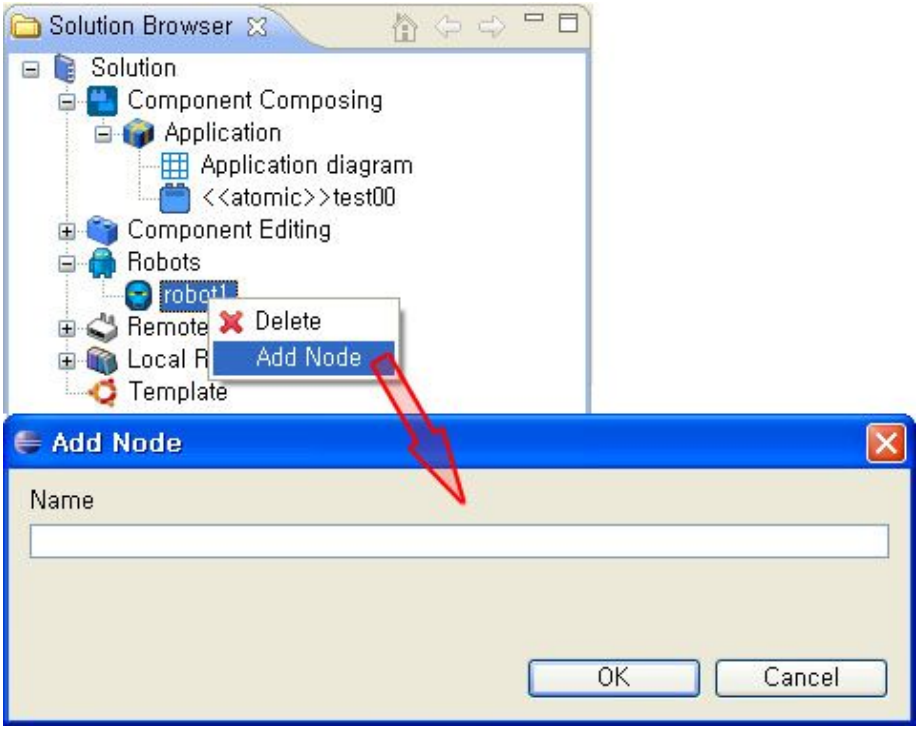

Figure 79. Example of Deploy 2

i. Enter IP and Port information into the node.

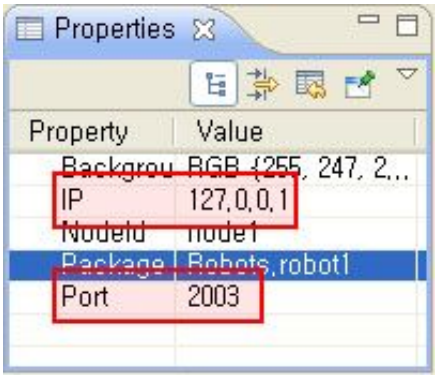

Figure 80. Example of Deploy 3

j. Drag-and-drop the node to add it into the component.

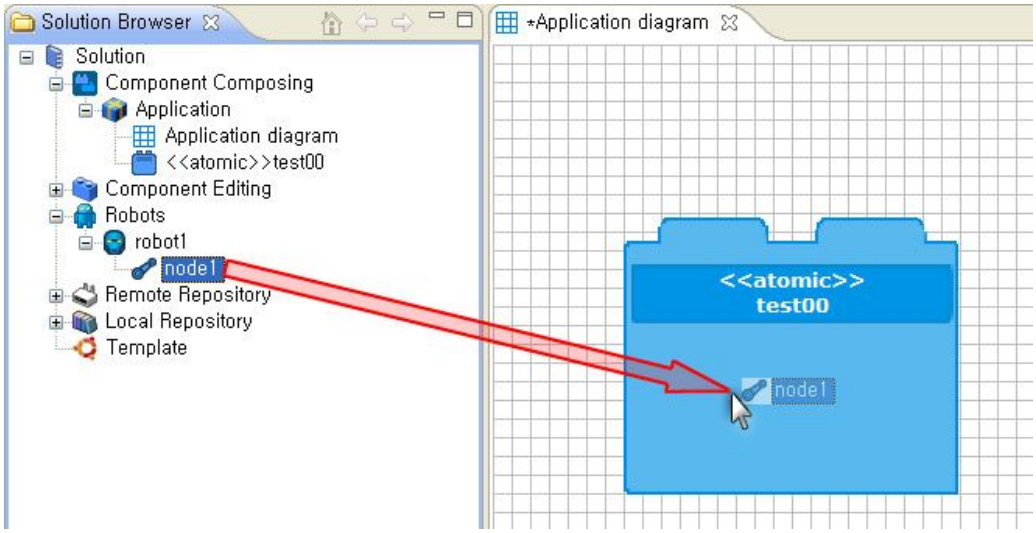

Figure 81. Example of Deploy 4

k. Execute the Application Package's 'Popup Menu>Deploy' to deploy the component into the node. You can execute one node or all the nodes at a time.

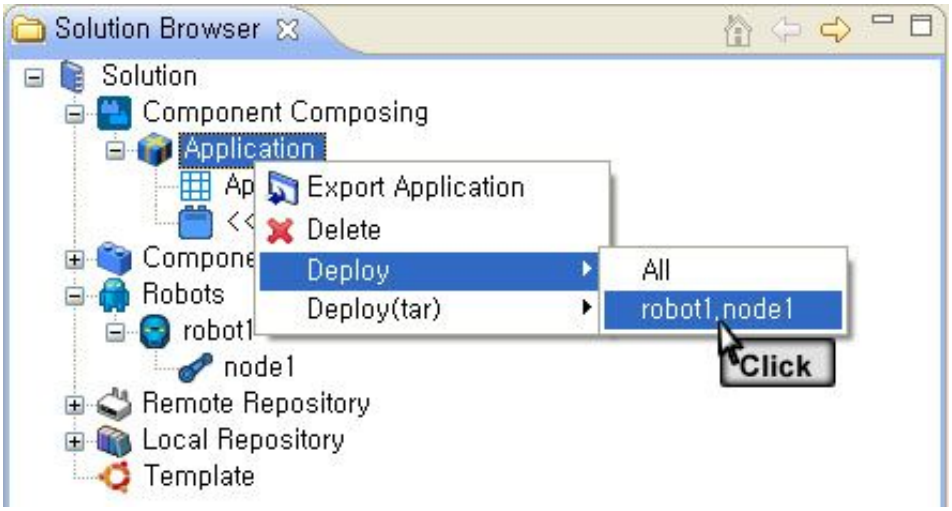

Figure 82. Example of Deploy 5

l. The released result can be checked on the Console window.

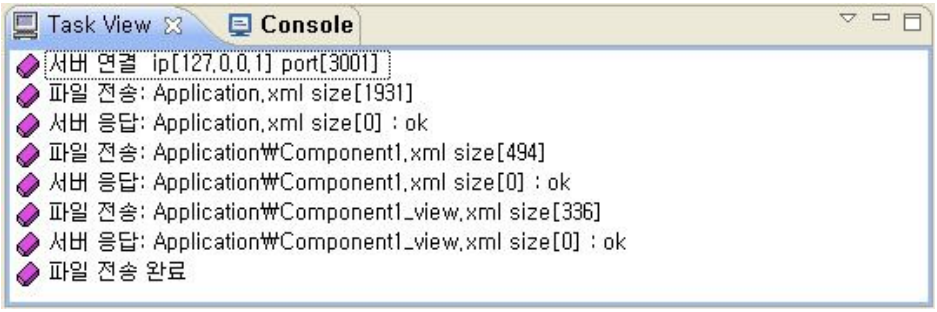

Figure 83. Example of Deploy 6

m. Execute the node's 'Popup Menu>Get Application List' to fetch the application list from the repository of the computer connected with the node, and indicate it.

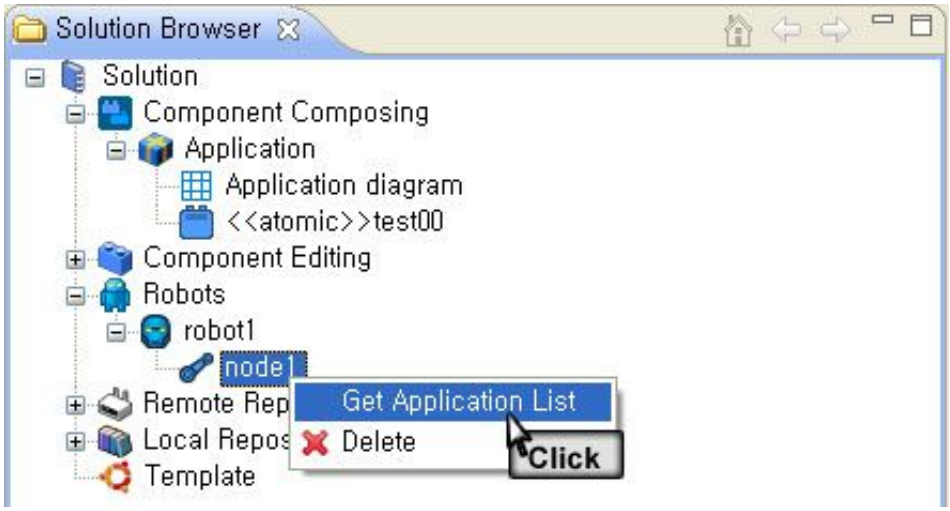

Figure 84. Example of Deploy 7

n. If succeeded in fetching the application list, an application package is created below the node, and the server's response information is printed on the console window. If failed, a message of 'Connection is Failed' is appeared.

Figure 85. Example of Deploy 8

o. Application List Menu

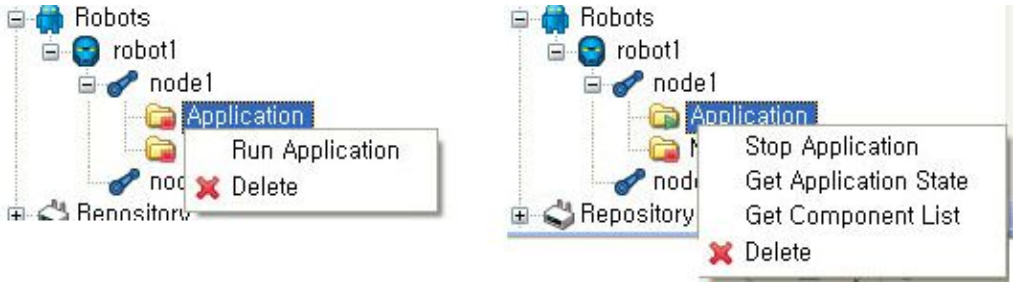

Figure 86. Example of Deploy 9

- Run Application: Executes the stopped application.
- Stop Application: Stops an application in execution.
- Get Application State: Prints state of the application in execution on the console window.
- Get Component List: Fetches a list of components included in the application to show it.

2.3.5. An Example of Monitoring

p. Execute the Robots' 'Popup Menu>Add Robot' to add a new robot.

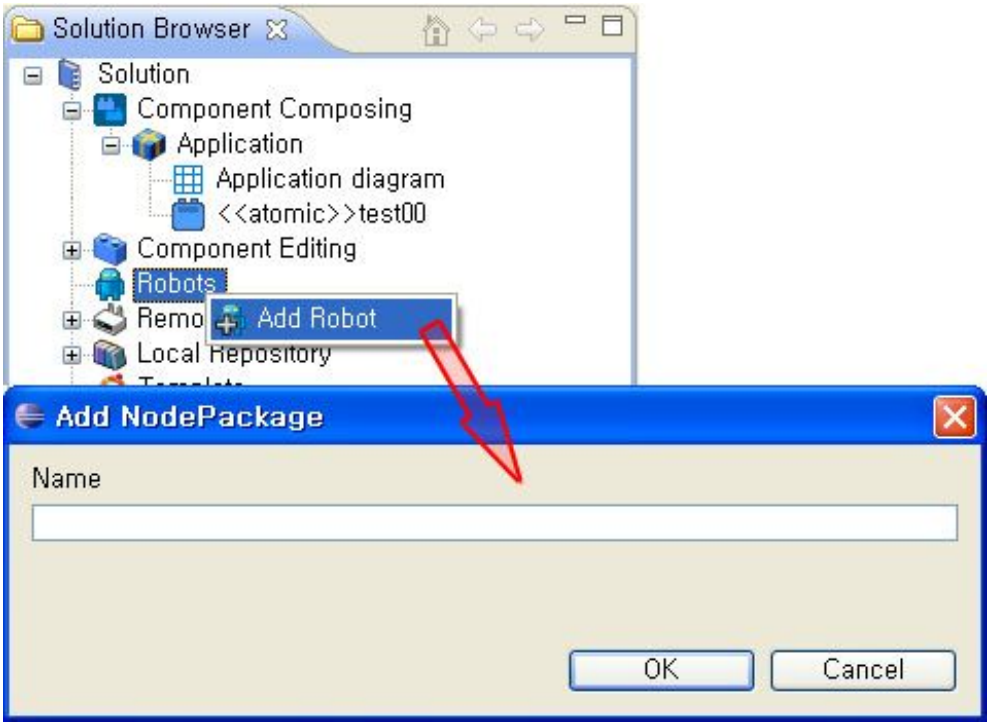

Figure 87. Example of Monitoring 1

q. Execute the added robot's 'Popup Menu>Add Node' to add a new node.

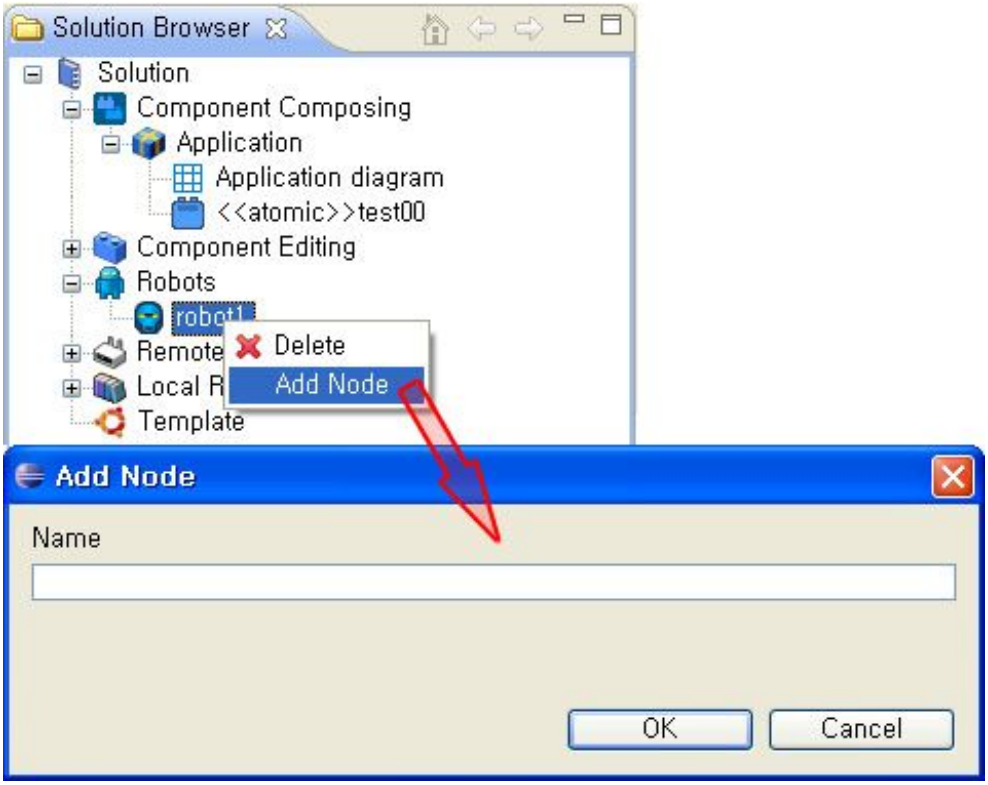

Figure 88. Example of Monitoring 2

r. Enter IP and Port information into the node.

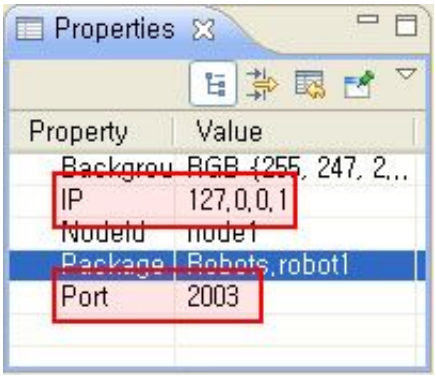

Figure 89. Example of Monitoring 3

s. Drag-and-drop the node to add it into the component.

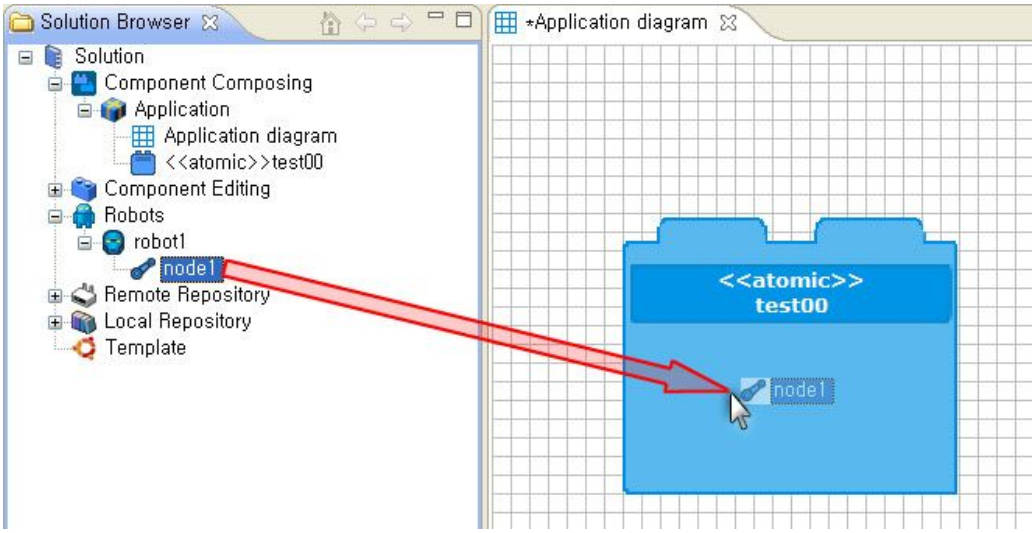

Figure 90. Example of Monitoring 4

t. If clicking the Run Monitoring on the popup window of the component to be monitored, the corresponding component can be monitored.

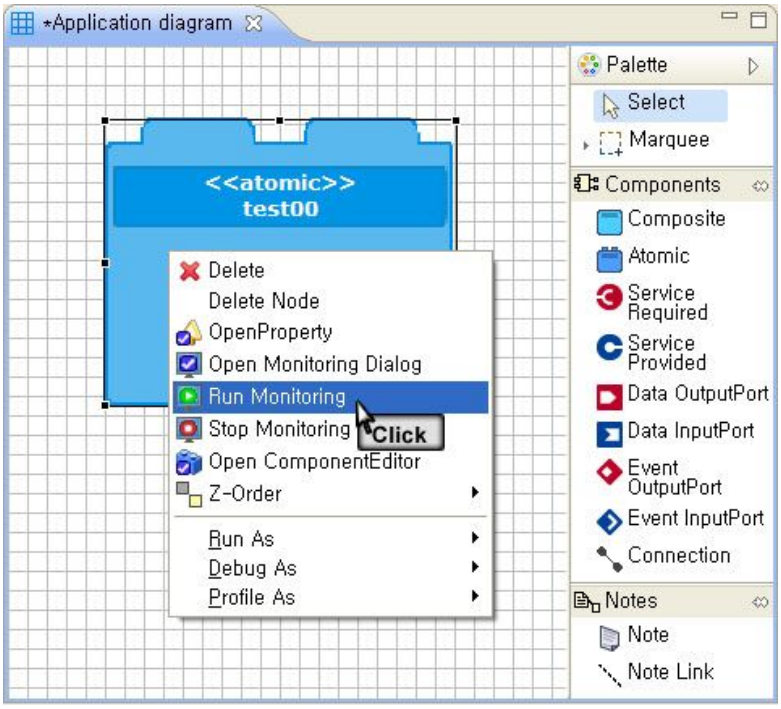

Figure 91. Example of Monitoring 5

u. It is executed as below.

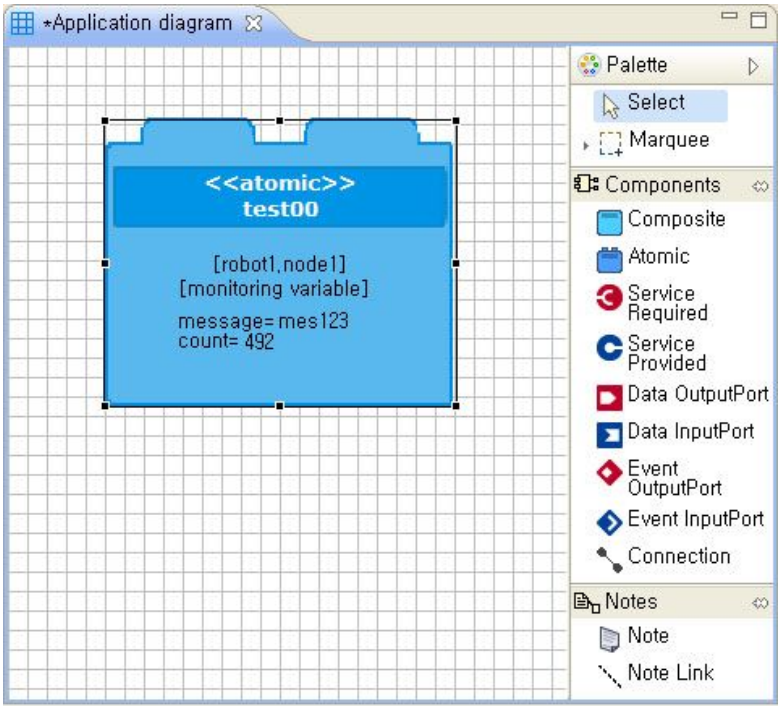

Figure 92. Example of Monitoring 6

v. If pressing the 'Popup Menu>>Stop Monitoring,' the monitoring can be stopped.

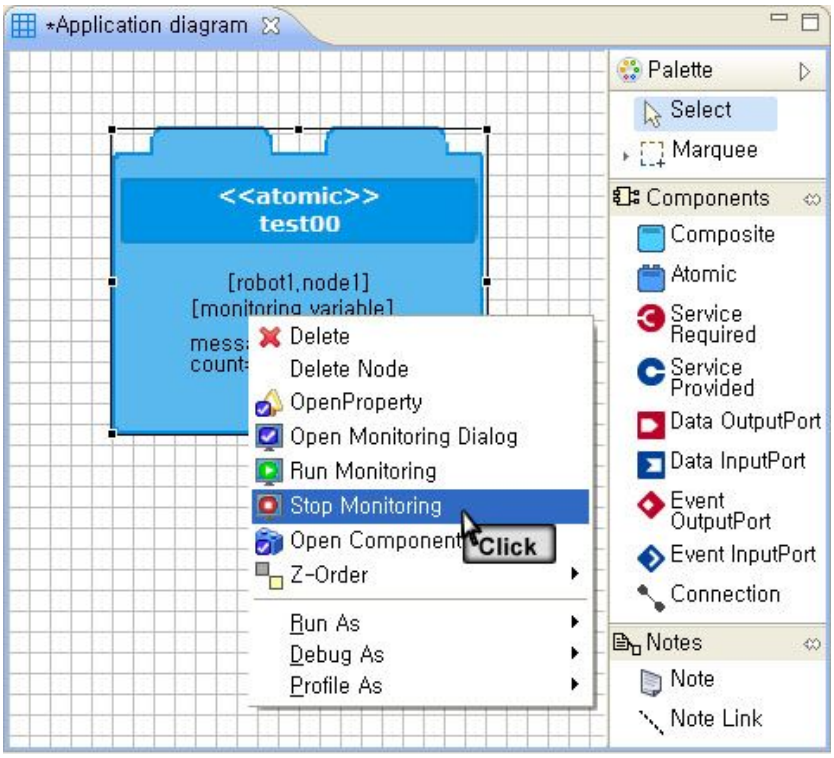

Figure 93. Example of Monitoring 7