

# Tutorial Demo

## Quick-Start Guide

In this Tutorial Demo, you will use the **Spoken Language User Interface (SLUI) Toolkit** to create an application in C++. The goal of your application is to help a user buy a product from BCL Computers. What makes the application unique is that the input request comes in the form of a **Natural Language (NL)** query-- in other words, the user will be able to enter a sentence in plain English, and your application will execute a function based on what the user wants! For more detailed information about the SLUI Toolkit, please refer to the documentation.

## Begin SLUI Toolkit

The SLUI Toolkit must be run directly from the Visual C++ environment because it is still in development.

1. Open Microsoft Visual C++
2. Load the SLUI Toolkit Workspace
3. Execute the Program **Ctrl+F5**

### See Figure 1: Main Program User Interface

The left side of the main window is called the **Project Tree Pane**

The right side of the main window is called the **Table Display Pane**

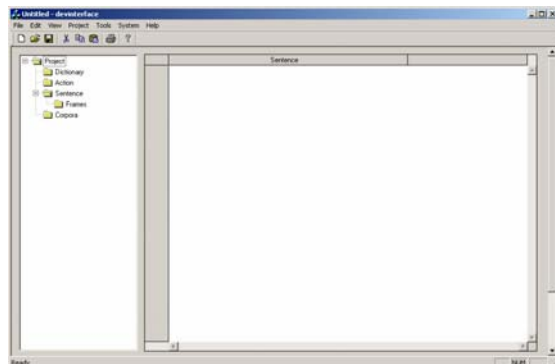


Figure 1

## Input Setup Information

One BCL Product you want the user to be able to buy is "Magellan." The first step in creating your application is to provide the SLUI Toolkit with sample sentences that would map to this request.

### Insert a Single Sentence

1. Click [**Project → Add into Project → Sample Sentences**]
2. Enter the sentence, "I want to buy Magellan."
3. Click [OK]

### See Figure 2: Add Sample Sentence Window

The sentence is processed by the Toolkit, and a frame is inserted into the **Project Tree Pane**. The frame contains semantic information about the sentence, and thus it is called a **Semantic Frame (SF)**.

### See Figure 3: Semantic Frame

	Action	Sentence Type	Predicate	Subject (Arg1)	Object1 (Arg2)
1		REQUEST	buy		Magellan

Figure 3

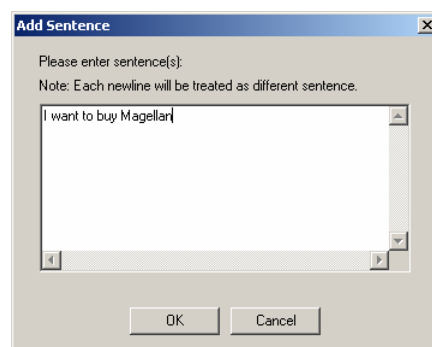


Figure 2

**Advanced Topic:**

You may create a SLUI-enabled application without completing this step. However, it is important to consider this step in the development process.

## Insert Corpus File

A **corpus file** is simply a collection of sample sentences contained in a text file. It is much more convenient to insert an entire corpus file than individual sentences. Before inserting the corpus file, make sure that a new line separates each sample sentence in the corpus file.

1. Click [**Project** → **Add into Project** → **Sample Sentences**]
2. Select the corpus file.
3. Click [OK]

## Expand Semantic Frame Table (SFT)

It is likely that the user will not enter the exact sentence, “I want to buy Magellan” if they want to make a purchase. The SLUI Toolkit will create a table of Semantic Frames called a **Semantic Frame Table (SFT)**. The SFT will include all of the sentence variations the user might enter, along with their associated actions. You must guide and monitor the three main processes of expanding the SFT: creating Variable Sentence Parameters (VSP), expanding synonyms and setting the actions.

**Advanced Topic:**

You may create a SLUI-enabled application without completing this step. However, it is important to consider this step in the development process.

## Variable Sentence Parameter (VSP)

A **Variable Sentence Parameter (VSP)** is a placeholder for a word in a sentence that might have several values. It is not necessary to specify a VSP in the development of the system, but it decreases the number of sample sentences that you must provide the SLUI Toolkit. Instead of using a specific product name such as “Magellan,” you can use a generalized product name. This general product name can be used to represent other products such as “Freebird” and “Drake.”

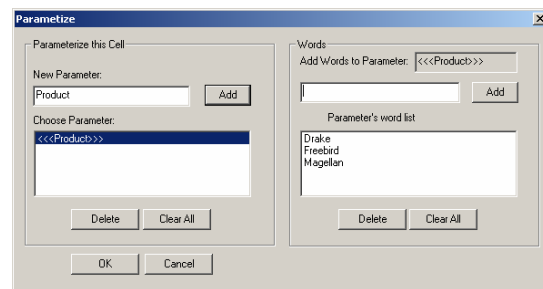


Figure 4

Refer to the documentation for information on how to extend VSP values across all of the sample sentences.

### See Figure 4: Variable Sentence Parameter Window

1. Select and Right-Click “Magellan” in the [Object 1] column of the **Table Display Pane**
2. Select [Parameterize this Cell] from the context menu
3. Enter “Product” in the **New Parameter text box** and click **Add**
4. Enter “Magellan” in the **Add Words to Parameter <<<Product>>> Text Box**
5. click [Add]
6. Enter “Freebird” in the **Add Words to Parameter <<<Product>>> Text Box**
7. click [Add]
8. Enter “Drake” in the **Add Words to Parameter <<<Product>>> Text Box** and click [Add]
9. Click [OK]

[<<<Product>>>] will replace “Magellan” in the [Object 1] column of the **Table Display Pane**

## Set Action

When the user enters a sentence that matches a **Semantic Frame**, an action will be executed in your application. The action can be either a function call or a URL redirection. In this case, when a user enters the sentence, “I want to buy Magellan,” the application will execute the function “Buy.”

1. Select and Right-Click the **Semantic Frame** in the **Table Display Pane**
2. Select [Set Action] from the context menu
3. Select “void BuyNow()” from the List Of Actions in the **Set Action Window**
4. Click [OK]

See Figure 5: Set Action Window

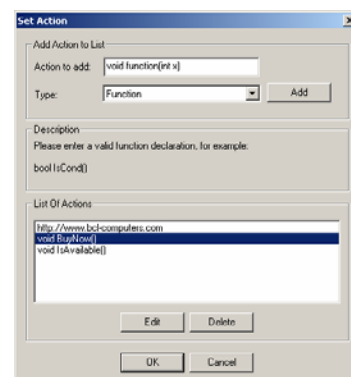


Figure 5

### Advanced Topic:

You may create a SLUI-enabled application without completing this step. However, it is important to consider this step in the development process.

## Expand Synonym

This step is not necessary in the development of the system, but it is an integral part of the SLUI Toolkit. The user will probably not use the exact words that you used in your sample sentence. For example, a user may use the words “order” or “purchase” in place of “buy.” To resolve this situation, the SLUI Toolkit inserts synonyms of the words in your sentence into the SFT. This will help guarantee that the correct action will execute when a user makes a request.

1. Select and Right-Click the **Semantic Frame** in the **Table Display Pane**
2. Select [Expand Synonym] from the context menu

See Figure 6: Semantic Frame with Expanded Synonyms

	Action	Sentence Type	Predicate	Subject (Arg1)	Object1 (Arg2)
1	void BuyNow()	REQUEST	buy		<<<Product>>>
2	void BuyNow()	REQUEST	order		<<<Product>>>
3	void BuyNow()	REQUEST	purchase		<<<Product>>>

Figure 6

**Advanced Topic:**

You may create a SLUI-enabled application without completing this step. However, it is important to consider this step in the development process.

## Debug Semantic Frame Table (SFT)

This step is not necessary in the development of the system, but it is important to make sure the **Semantic Frames** have been generated correctly. Use this step to make sure that the correct action will be executed when the user enters variations of the original sample sentence.

1. Click **[Project → Testing as a User]**
2. Enter the Sentence "I want to purchase Drake"
3. Click **[OK]**

**See Figure 7: Testing as a User Window**

The Toolkit searches the **Semantic Frames** for a similar representation of the sentence and displays the action.

4. Click **[OK]** to dismiss the Dialog Box
5. Click **[Close]** to dismiss the **Testing as a User Window**

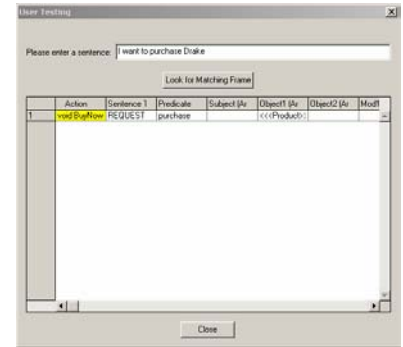


Figure 7

## Deploy Program

Finally, the SLUI Toolkit will generate code that will be used to compile your application.

1. Click **[Project → Generate Final System]**
2. Click **[Next]** to accept the default Database settings
3. Click **[Finish]** to accept the default File settings

## End SLUI Toolkit

Because the SLUI Toolkit is still in development, you must save the project and close the C++ environment.

1. Click **[File → Save Project]**
2. Type "Project\_1" in the save dialog box and click **[OK]**
3. Click **[File → Exit]**
4. Close Microsoft Visual C++

# Begin SLUI Enabled Application Development

This is the next phase in the creation of your application. All of the skeleton code has been generated by the SLUI Toolkit, and now you must interface with it. After you compile the application, we will use the same test sentence we used in "Testing as a User" because we know that the output will map to the "Buy" function.

1. Open Microsoft Visual C++
2. Load the SLUI Stub project **C:\Project\src\Driver\driver.dsp**

**Advanced Topic:**

You may create a SLUI-enabled application without completing this step. However, it is important to consider this step in the development process.

## Modify Generated Code

Here is where your application will respond to the user's NL input. You can modify the "Buy" function any way you wish, but for this example we will use a simple dialog box.

1. Insert these lines of code in the **BuyNow** function located in **Stub.cpp**:

```
CString outputText;
CStringArray arrParamList;
pResolvedFrame ->CreateParameterValueList(&arrParamList);
outputText.Format("void BuyNow() is called \n VSP Name: %s. \n VSP Value: %s. \n",
    pResolvedFrame ->GetParameterName(arrParamList.GetAt(0)),
    pResolvedFrame ->GetParameterValue(arrParamList.GetAt(0)));

gDriverView->SetOutputText(outputText);
```

## Execute your SLUI-enabled Application

Now you are ready to run your application!

1. Execute the Program **Ctrl+F5**
2. Click **[Process → Input]**
3. Enter "I want to buy Magellan" in the **Input Sentence Text Box**

**Advanced Topic:**

If you completed all of the Expand SFT steps:  
Enter "I want to purchase Drake" in the **Input Sentence Text Box**

4. Click [OK]

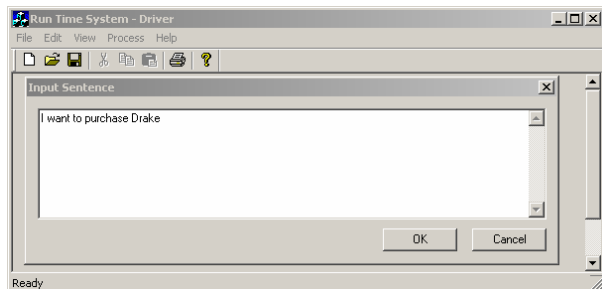


Figure 8

### See Figure 8: SLUI-enabled Application

The application will display "void BuyNow() is called" in the output text box. This indicates that the application has correctly understood the user's Natural Language sentence.

**Advanced Topic:**

If you completed all of the Expand SFT steps and modified the code, the application will display:  
Void BuyNow() is called  
VSP Name: Product  
VSP Value: Drake