

COBOL Translation & Documentation Toolkit



Technical Overview

System Configuration for Conversion

Setup

Client Information | General | Directories | **COBOL Settings** | Target Language Settings | Form Design | (Java) Default

Cobol Platform: []

Column Settings

Comment Pos	7
Code Starts	8
Code Ends	72
Area B Starts	11
Tab Spaces	8

Keep Comments in Original Source
 Reduce array sizes
 UNIX to DOS conversion

Reset Defaults

Setup

Client Information | General | Directories | **COBOL Settings** | **Target Language Settings** | Form Design | (Java) Default

Target language: Java For Borland JBuilder IDE
 Nonpersistence base class: ACMBaseWrapper
 Java Package: com.softwaremining.client
 Persistence base class: ACMBasePersistence
 Access level: private
 Generate JSP

GUI Class Prefixes

Frames/Forms	ACM
TextEdit/Edit Box	
Label	
Panel	
List Box	
Button	

SQL Conversion type

SQLJ Non SQLJ

Output Include Directories/Packages

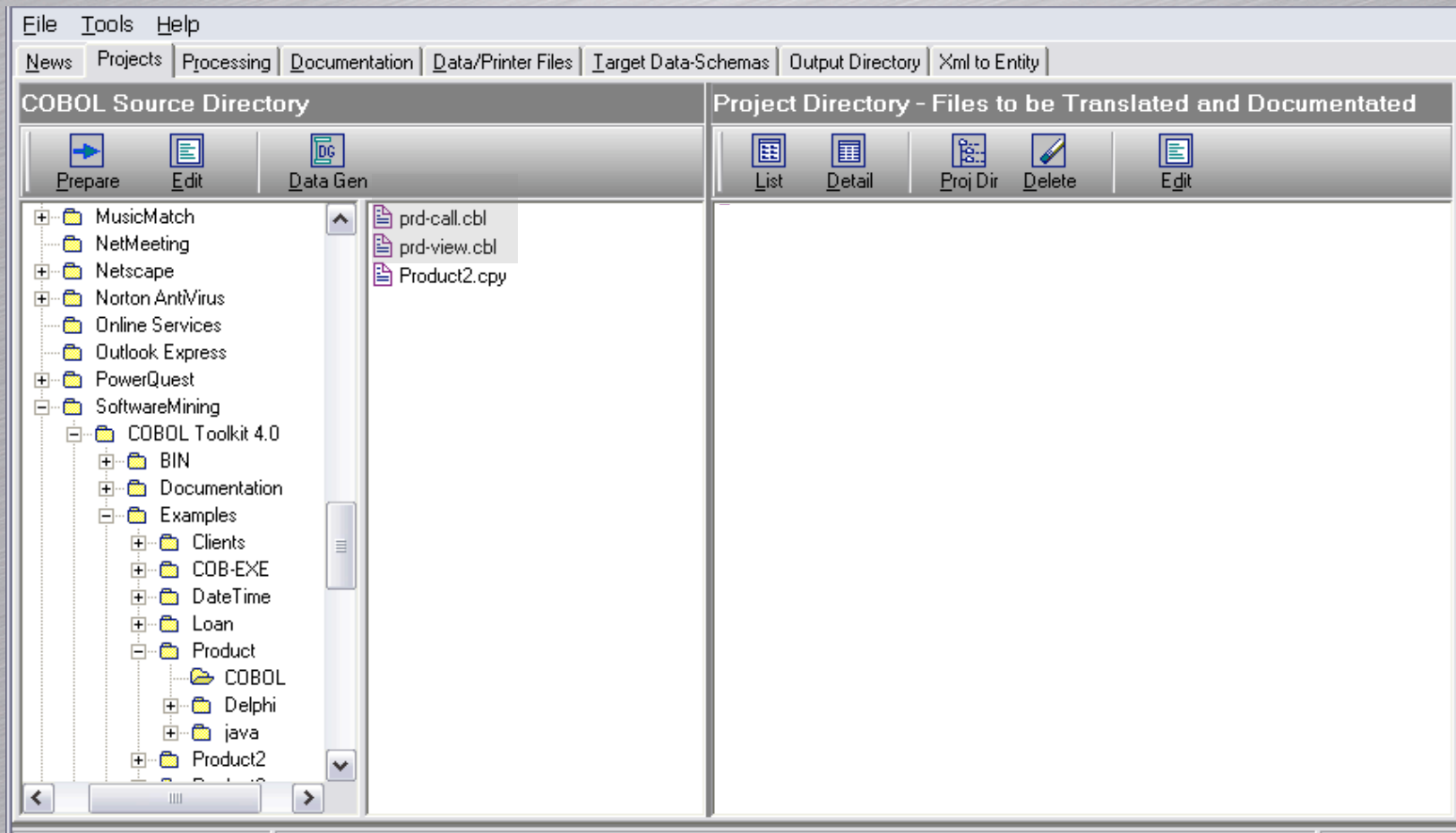
FD Wrapper Directory	FDInc
WS Wrapper Directory	WSInc

Target Database: Oracle
 Table Owner: []
 Additional Java Imports: []

Reset Defaults

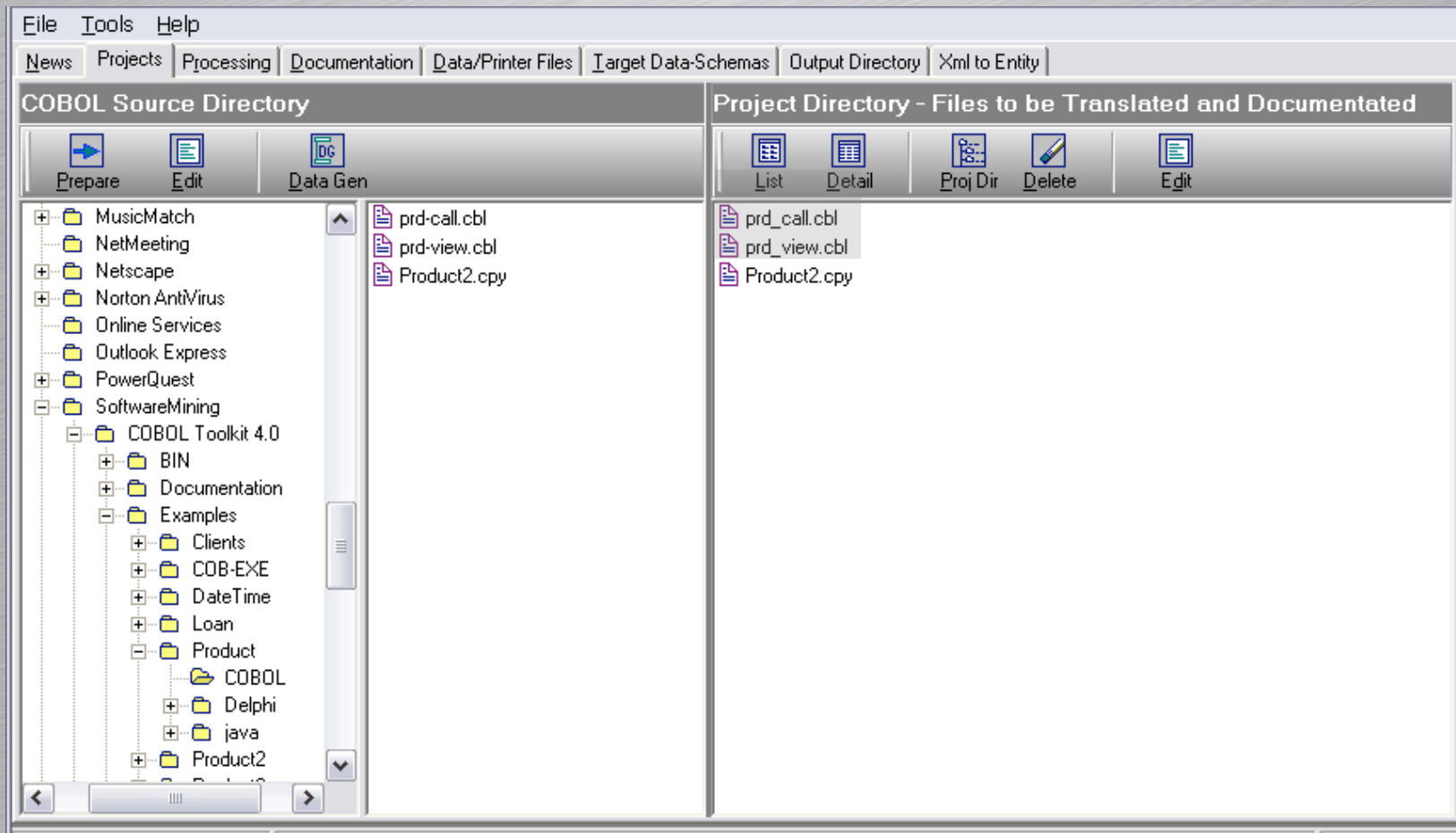
Accurate conversions require specification of key parameters
 Most important are COBOL source code characteristics. Other parameters include target language settings, etc.

Setting up the Projects



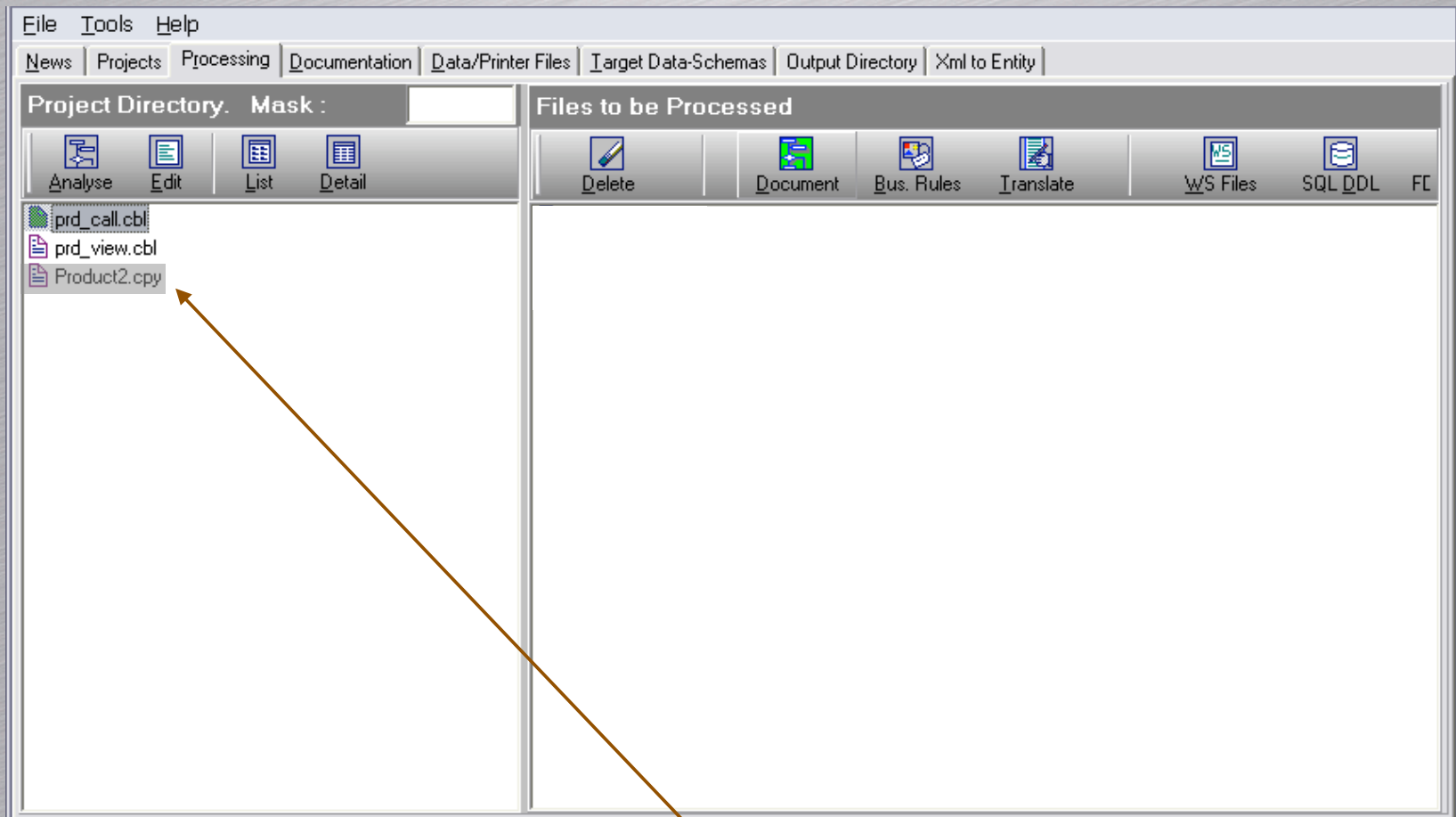
Select COBOL source code and copy-files

Setting up the Projects



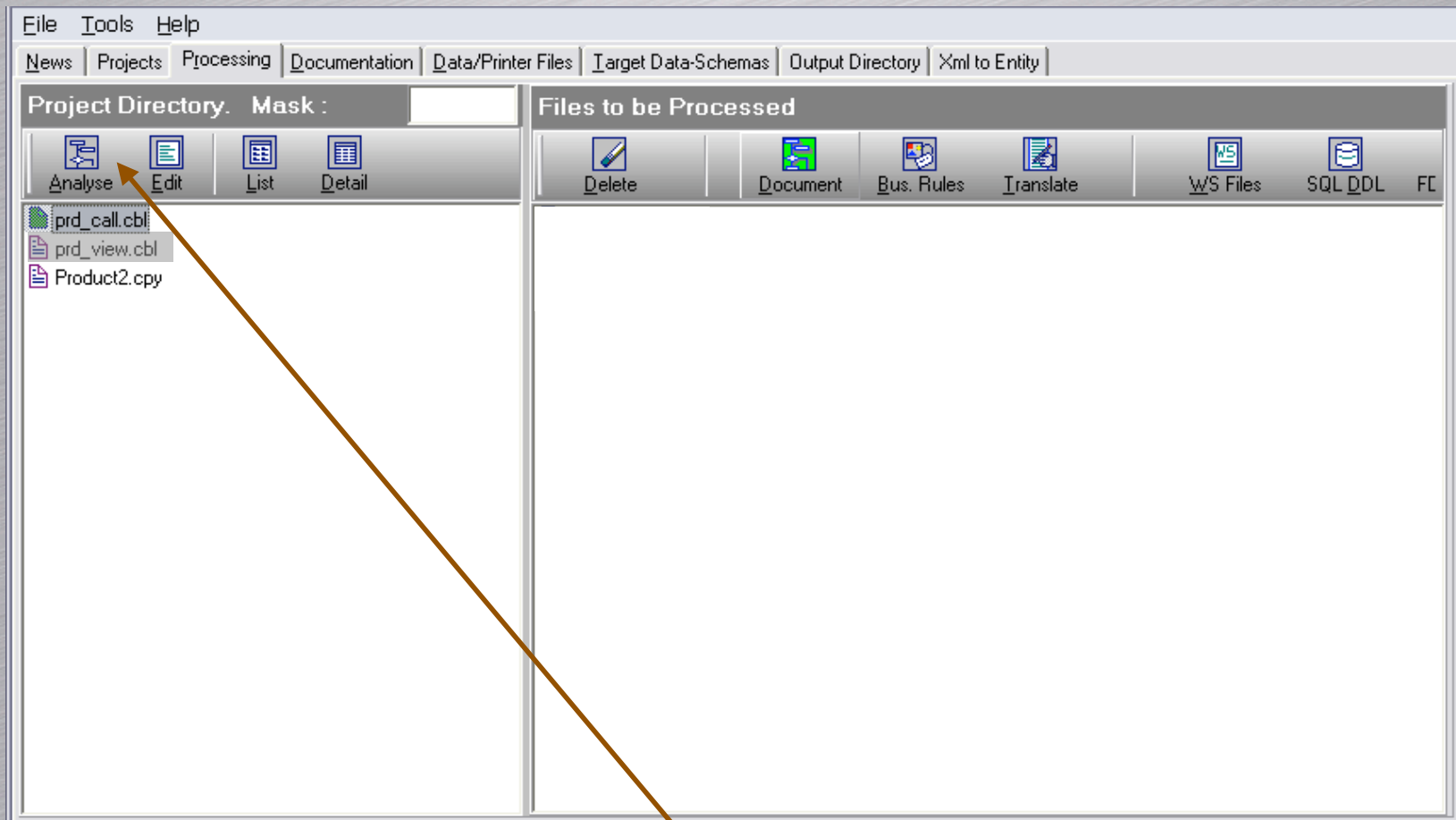
Move to the working repository

Processing Files



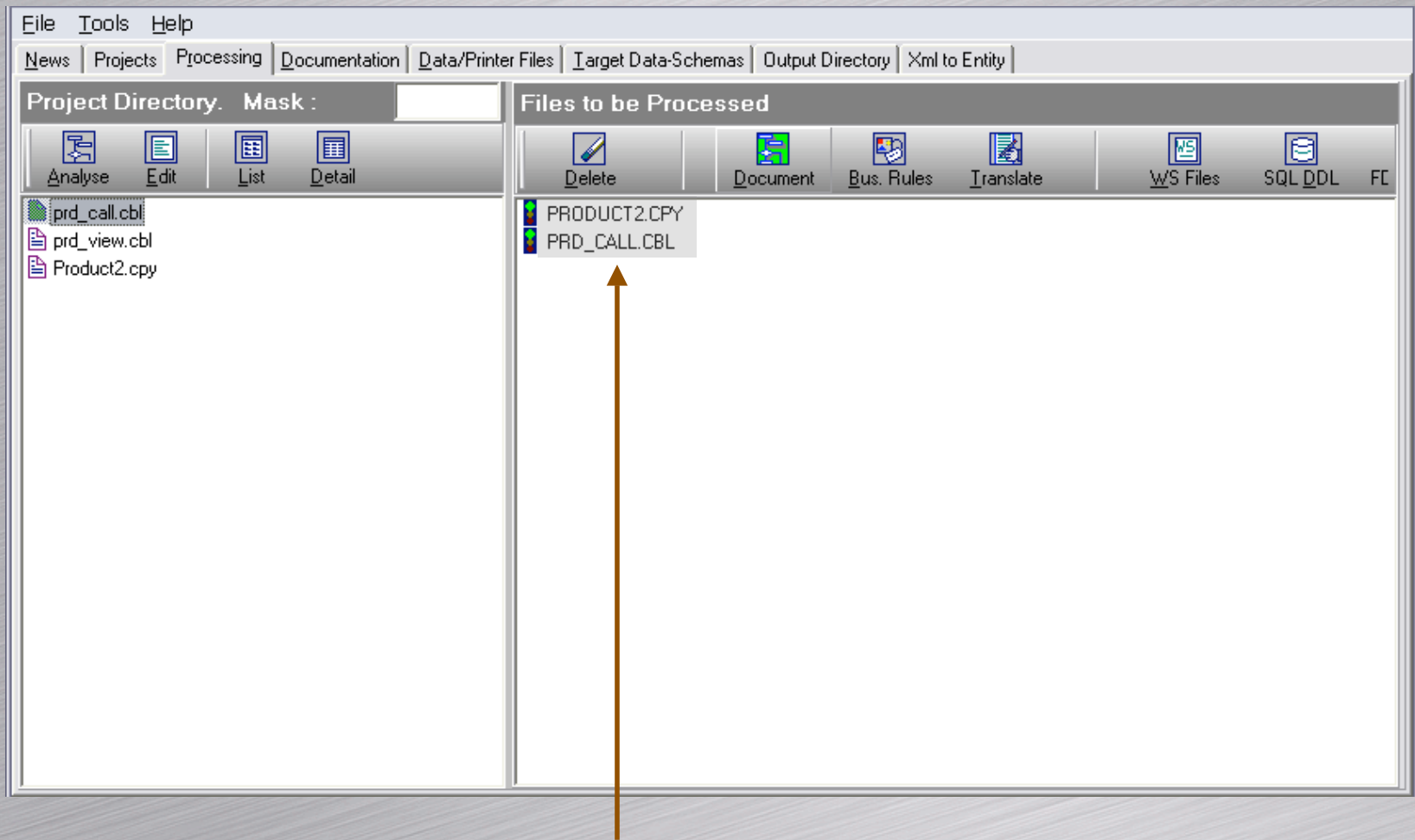
Select file(s) to be processed

Processing Files



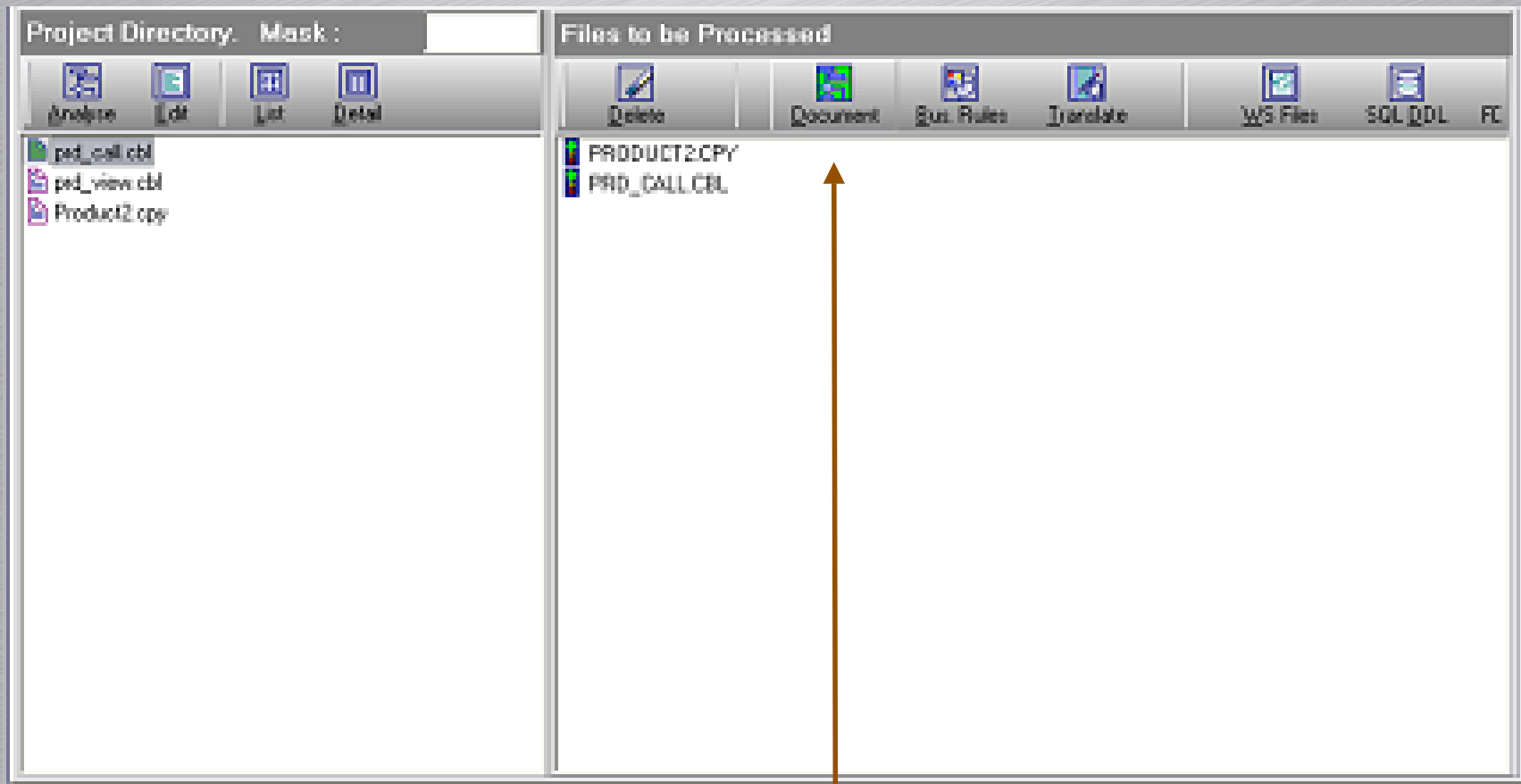
Click on the 'Analyze' button
The system will detect any required copy-files

Processing Files



Move all files to work area

Generating Documentation and Breaking Down Large Programs



Click on 'Document' button to generate the documentation.

Generating Documentation and Breaking Down Large Programs

The screenshot displays the Xactis software interface with the following components:

- Current Program Set:** A tree view showing the program structure for PRD_CALL.CBL, including sections like SELECT-ASSIGN, File Division, Working Storage Section, Linkage Section, Screen Section, Report Section, Communication Section, and Programs. The Programs section is expanded to show a DEFAULT program with sub-programs: MAINX, GET_TOTAL_VALUE, READ_NEXT_PRODUCT, and INVALID_OPTION.
- Field analysis and document flowchart:** A table listing field analysis results:

Slot / Record	Program Name	Contains: Redefines:
WS_DESCS	PRD_CALL...	
WS_ENTRY	PRD_CALL...	
WS_MISC	PRD_CALL...	
- SoftwareBinding - COBOL Documentation Toolkit:** A panel showing program details:
 - Program ID : PRD_CALL
 - Physical Name : PRD_CALL.CBL
- Flowchart:** A flowchart showing the execution flow:

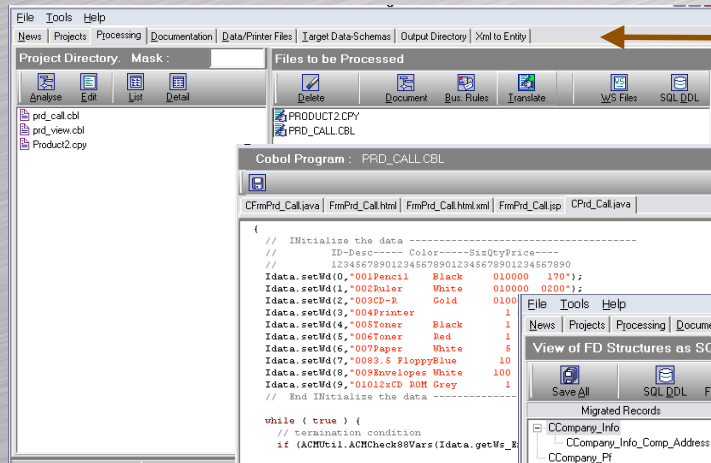

```

      graph TD
      A([MAINX]) --> B[GET_TOTAL_VALUE]
      B --> C[READ_NEXT_PRODUCT]
      C --> A
      
```

You can create new programs and move sections/labels here.

Generated documentation can be inspected, edited, saved, or exported to HTML/SVG

Heuristic Based Code Generation and a Translation of FD's to SQL Schema



'Translate' Button starts the translation process.

System by default will comment out dead codes (methods) and report on unused records structures.

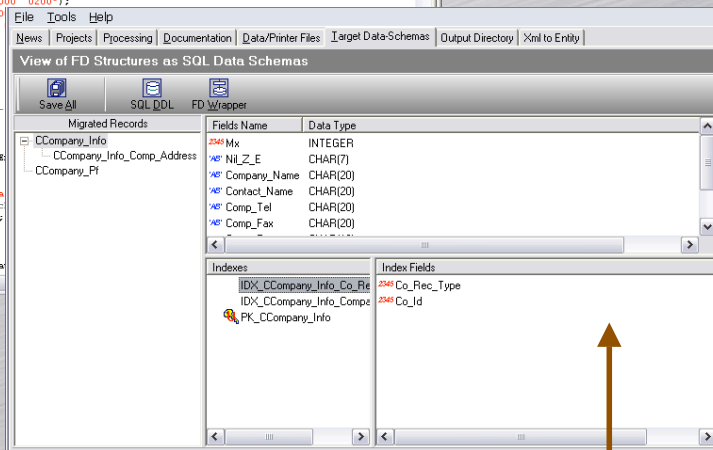
```

Cobol Program : PRD_CALL.CBL
No of Chargeable Lines : 100
CFrmPrd_Call.java | FrmPrd_Call.html | FrmPrd_Call.xml | FrmPrd_Call.jsp | CPrd_Call.java

(
// Initialize the data -----
// ID-Desc----- Color-----SizeQtyPrice-----
// 12345678901234567890123456789012345678901234567890
// 12345678901234567890123456789012345678901234567890
Idata.setWd(10,"001Pencil Black 010000 170*);
Idata.setWd(11,"002Puler White 010000 0200*);
Idata.setWd(12,"003CD-R Gold 0100 0100
Idata.setWd(13,"004Printer Black 1
Idata.setWd(14,"005Toner Black 1
Idata.setWd(15,"006Toner Red 1
Idata.setWd(16,"007Paper White 5
Idata.setWd(17,"0083.5 FloppyBlue 10
Idata.setWd(18,"009Envelope White 100
Idata.setWd(19,"01012xCD ROM Grey 1
// End Initialize the data -----

while ( true ) {
// termination condition
if ( ACHUtl1.ACHCheck88Vars(Idata.getWd(10)) == 0 ) {
// calculate the order value
invoke("Get_Total_Value", "CO10_Get_Tota
// display the order value + get user c
invoke("Display_Choices", "Disp_Sec02");
// get user choice
Wd_Entry_Update();
// check the user entry
if ( true == ACHUtl1.ACHCheck88Vars(Idata

```



COBOL File Descriptors (FD) will be translated to ANSI SQL Schemas.

Choices are Session Bean, Container Manage Bean or simply JDBC wrappers.

Business Rule Extraction

From 'Processing' page, click on 'Bus Rule' button.

Various views of the translated code (Java) can be presented.

The screenshot shows the Xactis software interface with several panels:

- Project Directory:** Shows files like prd_call.cbl, prd_view.cbl, and Product2.cpy.
- Files to be Processed:** Shows files like PRODUCT2.CPY and PRD_CALL.CBL. The 'Bus Rules' button is highlighted with an arrow.
- Program Structure:** Shows a flowchart with nodes like MAINX, GET_TOTAL_VALUE, and READ_NEXT_PRODUCT.
- Generated Code:** Shows Java code snippets for data initialization and a while loop.
- Data Items:** Shows a table of field names and their types.
- Data Filter:** Shows a table for filtering data items.
- Statement Filter:** Shows a list of statement types with checkboxes for filtering.

Business Rule Extraction II

Program Structure

Sect/Lab Filter Gen. Code Save Clone

Methods

- PRD_CALL
 - GET_TOTAL_VALUE
 - C000_GET_TOTAL
 - Block
 - READ_NEXT_PRODUCT
 - C010_READ_MOVE
 - Math
 - C020_READ_CALL

Statements

Data Items

Field Name
PRODUCT_INFO
WS_DESCS
WS_ENTRY
WS_MISC

Data Filter

Field Name
PRODUCT_INFO

Statement Filter

<input checked="" type="checkbox"/>	Condition statements
<input checked="" type="checkbox"/>	Exit statements
<input checked="" type="checkbox"/>	Persistence statements
<input checked="" type="checkbox"/>	Embedded SQL statements
<input checked="" type="checkbox"/>	Screen statements
<input checked="" type="checkbox"/>	Perform statements
<input checked="" type="checkbox"/>	Call statements

```

public void C000_Get_Tot
{
    // initialize loop va
    Idata.setI(new Integer(1));

    while ( true ) {
        // termination condition
        if (Idata.getI().intValue() == 11) break;

        Product2.setProduct_Info(Idata.getPaddedWs_Temp());

        // increment loop variable
        Idata.setI(new Integer(Idata.getI().intValue() + 1));
    }
}
    
```

STEP 3
Click on 'Filter' followed by 'Gen Code' buttons to see the subset of code matching the filter criteria.

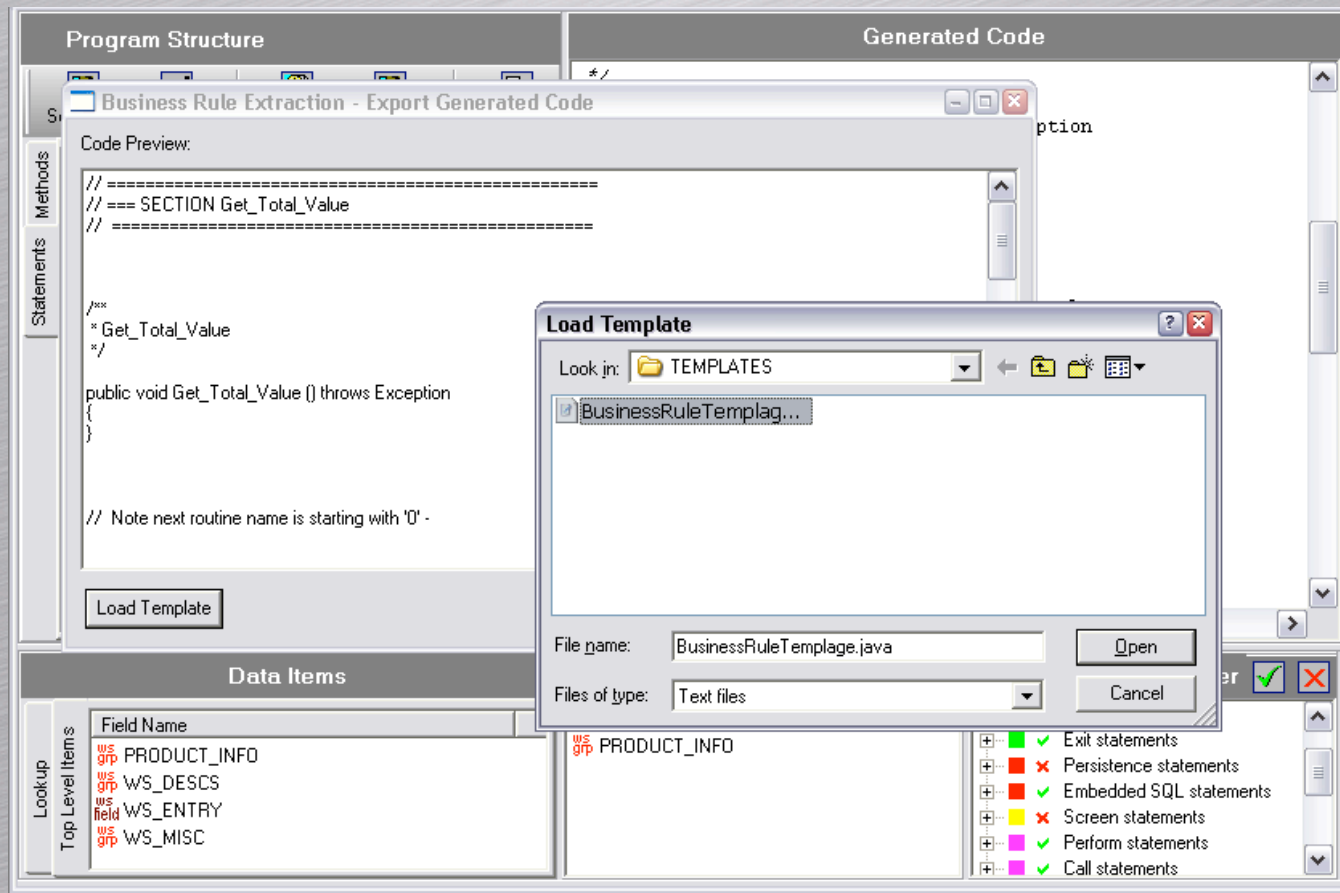
Additional manual tuning can be achieved by right hand clicking on each statement.

STEP 1
Select one or more variables of interest and move to data-filter.

STEP 2
Select one or more statement types of interest.

Business Rule Extraction III

Saving the code to 'Templates'



Architecture specific Templates can be designed by end users, to code “ported” into the customized frameworks – perfect for making SAIDware™ applications.