Enabling the Future

Natural/Adabas

Migration Solutions

SYSTRANS AUDITOR

© **2017 FBD Associates Inc**. Copying or distribution of this documentation is not permitted without the prior written consent of FBD Associates Inc.

Table of Contents

Overview

Purpose

One of the outputs from parsing is a report of missing modules. The reporting of missing modules, the collection of these Natural modules and re-parsing is an iterative process, since the introduction of additional Natural modules can lead to new missing Natural modules beginning a fresh cycle of code collection and re-processing. Since parsing is a task which can take days or weeks to complete depending on the volume of code being parsed, it is important to identify as many missing Natural modules as possible prior to parsing.

Written in java and using Apache Derby (open source relational database), the Systrans Auditor is a quick way to:

a) identify missing Natural modules and indicate where they are called from
b) suggest steplibs that may serve to eliminate some of the missing Natural modules
c) identify certain modules which, though missing, can be ignored (for example, modules that are confirmed by the customer to be irrelevant, Construct support modules, or SAG USR* modules)
d) identify 3GL calls and the modules which call them
e) identify Construct support modules that exists in the code.

The use of the systrans auditor assists with the missing modules analysis and can help avoid weeks of work lost in re-parsing customer code.

Description

The SystransAuditor reads the raw systrans files provided by a customer, and splits them up by library into:

- a set of *.trn files (one for each library) as well as
- a library for COPYCODE members

In addition, a 'steplibs' file is created *if and only if none already exists*. This file is created in the first run, and on subsequent runs it will not be overwritten.

The resulting *.trn files and the COPYCODE members are analyzed and the Modules table populated with all the modules in the code base by library and type.

Then the same set of systrans files are scanned again, this time searching for NATURAL keywords such as FETCH, CALLNAT, INCLUDE, PERFORM, CALL and for the objects that are referenced by these keywords, for example:

FETCH – PROGRAM
CALLNAT – SUBPROGRAM
INCLUDE – COPYLIB MEMBER
PERFORM – INTERNAL OR EXTERNAL SUBROUTINE
CALL – 3GL external module name

Dynamic CALLs and FETCHes are ignored (e.g.: CALL #EXCALL or FETCH #PROGRAM-NAME).

The scan first looks for the object in the current library and the designated steplibs. If it is not found then it will be reported as missing.

If it is not found in the first search, then a further search is done to see if it exists anywhere in the code base.

If it is found in the second search, then the modules is still reported as missing, but the name of the library in which it was located is stored as well.

Finally, reports are run on the missing modules.

Construct Support Modules which *do* exist in the code are identified and listed in a separate report. *Since these are recognized by name only this report may not be complete.* As more code is processed the list of known CSM names will expand to improve the coverage of these modules but it may never be 100% accurate.

Getting started with the Systrans Auditor

This section provides the details you'll need to run the systrans auditor.

Note that "v**x**" represents the version number below.

1. Create a folder called SOURCE under the customer name, and move to it all the files that contain the systrans information you have for that customer.

2. Save the following email attachment in a folder (you can save them in SOURCE or a separate folder):

   - FBDA.SYSTRANs.Auditor.zip

   If you are working online:

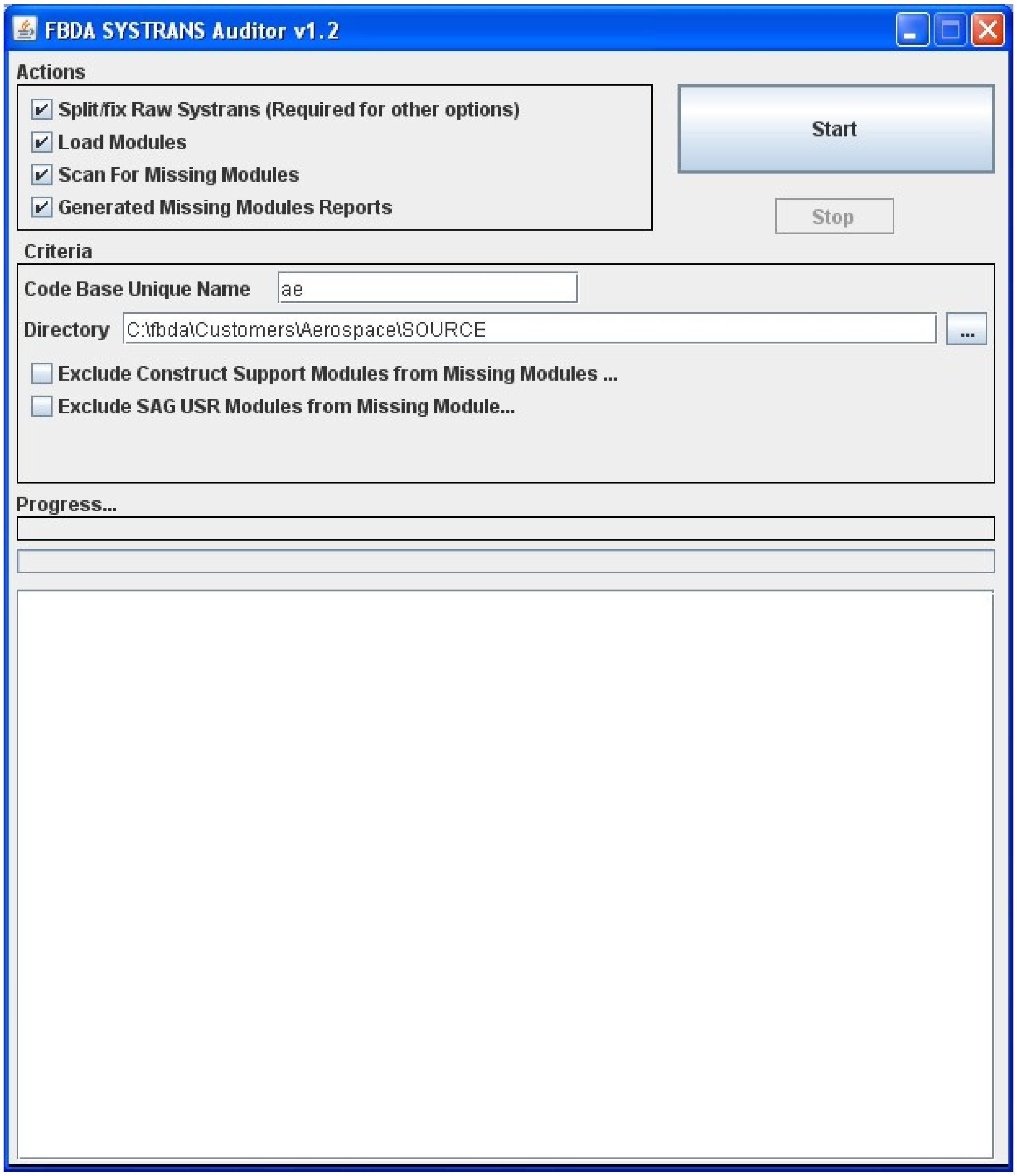
   - right-click on FBDA.SYSTRANs.Auditor.zip
   - select Save Target As
   - supply the name and location of the folder where it should be saved
   - click Save

3. Unzip the file FBDA.SYSTRANs.Auditor.zip. You should see the following:

   - FBDA.SYSTRANs.Auditor.v**x**.jar
   - a folder called "lib" that contains derby.jar

4. Double click on the file " FBDA.SYSTRANs.Auditor.v**x**.jar" to start the auditor.

The following window will appear. Enter the information as described beneath.


FBDA SYSTRANS Auditor v1.2

Actions
- [✓] Split/fix Raw Systrans (Required for other options)
- [✓] Load Modules
- [✓] Scan For Missing Modules
- [✓] Generated Missing Modules Reports

Start

Stop

Criteria

Code Base Unique Name: ae

Directory: C:\fbda\Customers\Aerospace\SOURCE

- [ ] Exclude Construct Support Modules from Missing Modules ...
- [ ] Exclude SAG USR Modules from Missing Module...

Progress...

Set the Options

Actions

- you must begin with **Split/fix Raw Systrans** as this populates the systrans files and creates a **steplibs file.**

  unless the txt (systrans) files have changed, there will be no need to rerun this step for a given customer.

- check **Load modules** to populate the Modules table.

  Again, unless the txt files are changed, or you have manually changed the systrans (.trn) files then there is no need to rerun this step.

- **Scan for Missing Modules** populates the Missing_modules and Affected_Modules tables

  - The **steplibs** file is generated with SYSTEM as a steplib for all libraries.

  - If required, further Steplib information can be entered in the **steplibs** file for each library in the format:

    Prefix.steplibs=SYSTEM,LIB1,LIB2, etc.

  - Unless you have added more steplib information there is no need to rerun this either

- **Generate Missing modules Reports**

  - Again, if you have modified that steplibs file and rerun the preceding step then these should be rerun also

Criteria

- Enter the code base unique name (Choose a name; e.g.: Aerospace Engineering could be AE)

- Supply the SOURCE directory where the raw systrans are located

Exclusion Options

- Exclude Construct Support Modules from Missing Modules…

  This option will ignore construct support modules when reporting

- Exclude SAG USR Modules from Missing Module…

  This option will ignore SAG USER modules when reporting

Click on Start button

When the options have been set, click the start button to start processing.

Steplibs file

The **steplibs** file is named ***prefix*.steplibs**, and contains a line for each library in the systrans.

E.g.: AE.steplibs

The generated file looks something like this:

```
ETSANI.steplibs=SYSTEM
ETSAPC.steplibs=SYSTEM
KRONOS.steplibs=SYSTEM
STSL.steplibs=SYSTEM
```

If you wish to add steplib information, append the library name with commas between library names. Like this for example:

```
ETSANI.steplibs=SYSTEM,ETSAPC,KRONOS
ETSAPC.steplibs=SYSTEM
KRONOS.steplibs=SYSTEM
STSL.steplibs=SYSTEM,STSLINFO
```

Excluding selected modules

If there are modules that you do not expect to find or whose existence is not critical (generally it is the customer who makes this call), you can exclude these from the processing so they will no longer appear on the Missing modules Reports.

To do this, create an exclusion file and save it in the same directory where the systrans were saved (SOURCE), and rerun "Scan for Missing Modules" and "Generate Missing Modules Reports". (There is no need to rerun the first 2 steps for this purpose).

- In the same directory as the systrans (SOURCE), create a text file called ***prefix*.Ignore** where *prefix* = Code Base Unique Name (as above)

  E.g.: AE.Ignore

- Enter the names of the Library.module(s) you want to ignore, 1 module per line, in this format:

  ```
  LIB1.AELDA01=N
  LIB1.AEPROG01=N
  LIB2.XXX0001=N
  ```

- If there are some modules that can be ignored *no matter what library they appear in*, then enter them without the library name. E.g.:

  ```
  COMCODE1=N
  COMCODE2=N
  COMCODE3=N
  ```

  The resulting file contains either the first or second type or a combination of both.

  ```
  LIB1.AELDA01=N
  LIB1.AEPROG01=N
  LIB2.XXX0001=N
  COMCODE1=N
  COMCODE2=N
  COMCODE3=N
  ```

  It does not matter in what order they are entered.

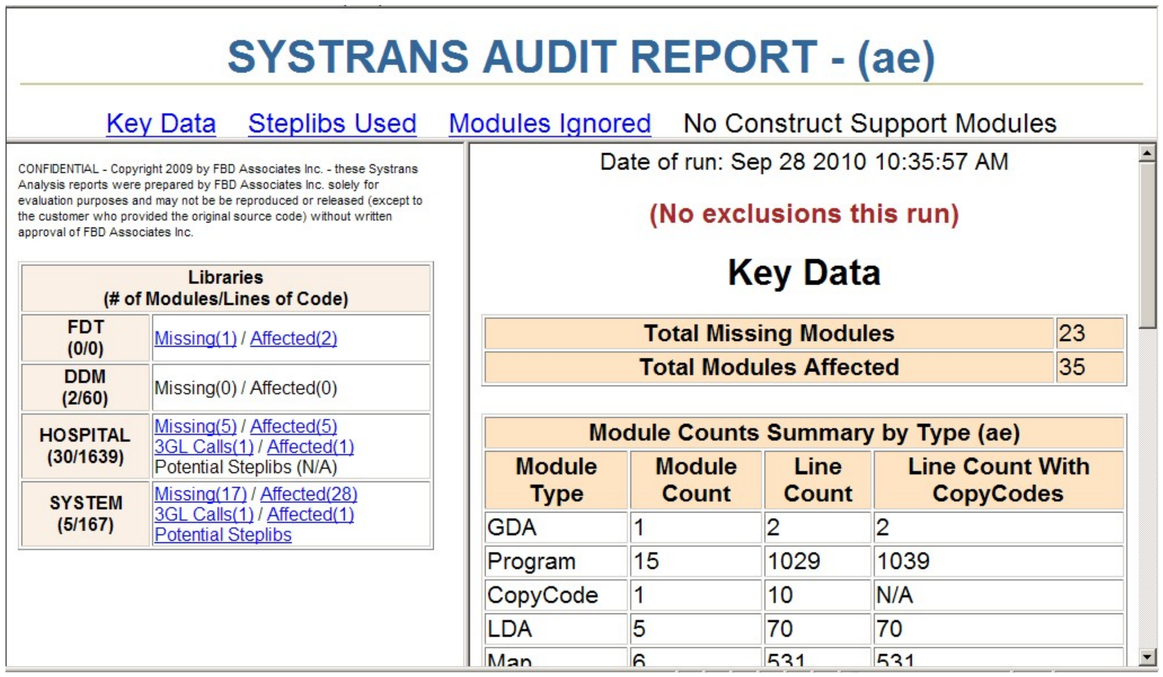
- Now rerun the last 2 steps of the systrans audit (i.e.: “Scan for missing modules” and “Generate Missing Modules Reports”)

Viewing the reports

When done, the results will be stored in the same directory as the one you specified for the systrans file, under a new folder called "Reports".

Open the Reports directory and double-click on **index.html** to view the reports.

You will see the welcome screen, whose format is show below:



SYSTRANS AUDIT REPORT - (ae)

Key Data Steplibs Used Modules Ignored No Construct Support Modules

CONFIDENTIAL - Copyright 2009 by FBD Associates Inc. - these Systrans Analysis reports were prepared by FBD Associates Inc. solely for evaluation purposes and may not be be reproduced or released (except to the customer who provided the original source code) without written approval of FBD Associates Inc.

Libraries (# of Modules/Lines of Code)	
FDT (0/0)	Missing(1) / Affected(2)
DDM (2/60)	Missing(0) / Affected(0)
HOSPITAL (30/1639)	Missing(5) / Affected(5) 3GL Calls(1) / Affected(1) Potential Steplibs (N/A)
SYSTEM (5/167)	Missing(17) / Affected(28) 3GL Calls(1) / Affected(1) Potential Steplibs

Date of run: Sep 28 2010 10:35:57 AM

(No exclusions this run)

Key Data

Total Missing Modules	23
Total Modules Affected	35

Module Counts Summary by Type (ae)

Module Type	Module Count	Line Count	Line Count With CopyCodes
GDA	1	2	2
Program	15	1029	1039
CopyCode	1	10	N/A
LDA	5	70	70
Map	6	531	531

Top panel

There are 4 links in the top panel:

- **Key Data** (returns to this screen)
- **Steplibs Used** (link to steplib information that was input this run)
- **Modules Ignored** (link to list of modules ignored this run)
- **Construct Support Modules** (link to list of CSM *if any are identified*)

Left panel

On the left panel is a summary of all the libraries processed. ***Each of the following is a link to the details:***

- **Missing(n)** / **Affected(n)**
  Missing Modules total (modules invoked but not found in the current library or any steplib supplied)/Affected Modules total (modules that invoke those missing modules)

  Details shown for each missing module:

  Module Name
  Module Type
  CSM - "Y" if it is a missing Construct Support Module
  Found in Library(ies) – if module exists in a different Library
  Modules Affected – modules that call or use the missing module

- **3GL Calls(n)** / **Affected(n)**
  3GL calls total / total of modules that invoke those 3GL calls

- **Potential Steplibs**
  Potential steplibs (libraries where "Missing modules" were found)

Right panel

Initially, the right panel shows "**Key Data**", which are the options used and a summary of the application (lines counts by module type):

- Date and time of run
- Exclusion options used (Construct and SAG USR* modules)
- Total Missing and affected modules across the application
- Total module and line counts for each individual library

When any of the links is selected from the left pane the requested information will replace the Key Data, which can be redisplayed at any time by clicking on the "**Key Data**" link.

For assistance

If problems are encountered or assistance is required please contact Susan Boyd at susanb@fbda.ca.