CMS Snapshot Analysis - Run Analyzer - Missing or Wrong links between two objects

Purpose (problem description)

This guide gives the approach to troubleshoot issues with missing or wrong / incorrect links between two objects. For more information on links refer to each page in official documentation for CAST AIP supported technologies:

- For CAST AIP 8.3: Covered Technologies
- For CAST AIP Extensions: CAST AIP Extensions documentation

Observed in CAST AIP

Release	Yes/No
8.3.x	✅

Observed on RDBMS

RDBMS	Yes/No
CSS4	✅
CSS3	✅
CSS2	✅

Step by Step scenario

1. Perform analysis.
2. Open Enlighten.
3. Drag two objects to the view and find that the link is incorrect / missing.

Action Plan

Proceed with the below steps:

1. Validate if link is supported by CAST AIP
    a. If you have confirmed that the link is not supported then this is an **EXPECTED** behavior.
    b. If link is supported or if it is a known issue that should be supported or if you cannot confirm expectation, proceed to next step
2. Validate that the caller and callee objects exist and that you are looking at the correct objects, by completing the first 4 steps in Enlighten - Links.
    a. If you cannot find the caller or the callee, then you should re-qualify the problem as a Missing Object issue.
    b. If you have identified them, then you should have an Enlighten Graphical view with the caller and the callee objects and you should have obtained the respective ObjectID.
3. Validate link in source code
    a. In the case of missing link, if the link is wrongly expected this is an **EXPECTED** behavior.
    b. In the case of missing link, if link is confirmed to be expected proceed to the next step.
    c. In the case of wrong link, if the link is expected this is an **EXPECTED** behavior.
    d. In the case of wrong link, if the link is confirmed to be invalid, then proceed to the next step.
4. Validate analysis configuration
5. If you have not solved the issue:
    a. Check if your case matches one of the known cases with solution, described in the pages under CMS Snapshot Analysis - Run Analyzer - Missing or Wrong Links per technology
    b. Check the prop type of links from ACC table. There is some issue where property types are not 0 but set to some values like 65537 which means the links are set as ignored. In this case, you will be able to see the links in the table but on Enlighten the links will be ignored and will not be present. So you need to reevaluate the links and change the links back to its normal property by using some rule file or maybe take a fresh local and run the analysis again to get good results.
    c. If your issue is a missing link, check the appropriate log file and remediate errors or warnings
    d. Evaluate KB Corruption and the cost of moving to a fresh Knowledge Base : If the Knowledge Base is corrupted and the missing /wrong link number is important then the best solution is to run the analysis again in a fresh installed KB.
    e. Remediation using tools
6. If you do not find the information you are looking for a solution for your problem, contact CAST Technical Support, with the below Relevant Input

Validate CAST AIP supported Links

Validate if links are supported by the CAST AIP analyzers and extensions

The goal of this paragraph is to find out if the link is **EXPECTED** or **NOT EXPECTED**, according the the supported objects and links created by the CAST AIP analyzers. Retrieve information about supported links for the relevant CAST AIP version and the various supported technologies in the official documentation.

1. For the core CAST AIP analyzers the relevant pages for each technology can be found in Covered Technologies. You can also find information on expected links for each technology in Objects and Links Guide
2. For CAST AIP extensions the relevant pages for each officially supported extension can be found in CAST AIP Extensions documentation wher e you can find info on expected objects and links
3. You should also search in Known Issues and Vulnerabilities for the affected version to detect if this is a known issue. Some of the technology or extension pages can also contain information for known limitations and possible workarounds
4. In case you have created several applications in a single management base and these applications have been analyzed in a single knowledge base, inter-application links are not expected. CAST analyzer only creates links for objects analyzed in the context of the same application.

Validate Source Code

Validate link in source code

Validate link expectations from the source code. This paragraph will help you find out whether the link is **EXPECTED** or **NOT EXPECTED**, based on the source code.

1. Open the source file containing the caller. and search for references to the object name of the callee.

    a. **If you find a reference to the callee name in the caller source file the link is EXPECTED, except in particular cases where more checks are needed:**
        i. **Compiled languages : imports and includes**

        If the technology of both caller and callee is one of the following : J2EE, C, C++, .Net then you should wonder : would this code compile? In other words, are all the necessary includes and imports present ? And are all the paths correctly defined ?

1. **J2EE**
    a. **: Links between Java objects**
    Check that the class containing the callee is imported into the caller source file. For example, if you expect a link between method getC() from class A and method makeC() from class B() :

    Example : Java import

    ```
    class A{
    private B myB;
    .....
    public C getC(){
        return myB.makeC();
    }
    ```

    Then you need to check that either classes B and C belong to the same package as class A, or you have the following lines at the beginning of file A.java:

    ```
    import com.mypackage.B;
    import com.myOtherPackage.C;
    ```

    If classes B and C do not belong to the same package as class A, or if the imports are not present, then the link is **NOT EXPECTED**

    b. **J2EE : Links from a JSP page**

    If the callee is another JSP file, then you must check that its path is correct in the caller source file. For example:

    Example : link from JSP to JSP

    ```
    <%@ include file="some/relative/path/other.jsp" %>
    <%@ include file="/some/absolute/path/yetAnother.jsp" %>
    ```

    c. Check that the relative path is correct, that is to say that, starting from the folder of the caller source file, there is a file called other.jsp located in <*folder_of_the_caller*>\some\relative\path
    d. Check that the absolute path is correct, considering as root the path defined as Application root path in Cast-MS configuration.
    Example : if Application root path is defined as *S:**Sources**App**webapp* then you should check that file *S:**Sources**App**webapp**some**absolute**path**yetAnother.jsp* exists
    e. If the callee is a Javascript method, please check that :

        i. The Javascript file containing the Javascript function is included
        ii. The Javascript file contains the Javascript function
        iii. The path of the Javascript file is correct

    f. If any of the above conditions are not true the link is **NOT EXPECTED**

2. **.NET :**

    a. **Links between C# methods**

    Inside a .Net job, we can analyze many different projects (many csproj files and/or ASP websites). Links are created between objects of 2 different projects only when one of these projects refers the other. If the missing links are between 2 objects then

        i. Check if the 2 objects belong to same project or not. Check if the cs files the objects belong to are refered in the same .csproj

            1. If the 2 objects belong to the same project then the link is **EXPECTED**
            2. If the 2 objects do not belong to the same project, check if one of the 2 projects refers to the other
                a. Check the references on each csproj file project
                Example: if the 2 projects are called ConfigFileLibrary and AdminFileLibrary
                If in AdminFileLibrary.csproj file you have reference to ConfigFileLibrary.csproj, as following one, this means that AdminFileLibrary refers the ConfigFileLibrary project. <ItemGroup>
                <**ProjectReference Include="..\testConfigFileLibrary\ConfigFileLibrary.csproj**">
                <Project> Project1</Project>
                <Name>testConfigFileLibrary</Name>
                </ProjectReference>
                </ItemGroup>
                b. Ensure that reference to the project contains relative path as this oneProjectReference Include="..\testConfigFileLibrary\ConfigFileLibrary.csproj {*}">
                The reference should not contain only the project name as <**ProjectReference Include="ConfigFileLibrary.csproj**">
                c. If one of the csproj contains a reference to other projects with a relative path, the link is **EXPECTED** else the link **NOT EXPECTED.**

3. **C++ :**

    a. **Links between objects**

    Check if the required header files are included in the caller source file. If the code of the caller calls a callee method, then the header file defining the callee should be included, as follows :

    C++ include of definition of the callee

    ```
    #include  "some/path/to/header/containing/callee_definition.h"
    ```

    You have to check that the path of the include file is correct (the file exists), else the link is **NOT EXPECTED**.
    There is also a C++ directive which informs the compiler that '.h' file must be included only one time in the C++ source code, it is : "**#pragma once**".
    For example, if C++ source file (a.h) contains this directive and this file is included by two other source files (b.cpp and c.cpp), then the C++ analyzer will take into account only the first include directive that it encounter (ex in b.cpp). So a link will be created between the two files (a.h and b.cpp), and all other #include directives will be ignored and no more include link will be created to the file a.h (c.cpp ---> a.h). In this case, the missing links are expected behavior, as the links are **NOT EXPECTED**

b. **If you don't find a reference to the callee name in the caller source file, then the link is NOT EXPECTED, except in the following particular cases**

i. **Caller belongs to an Object-Oriented technology (J2EE, .Net, C++) : dynamic link by inheritance**

In such languages, a link is expected if the reference is in an ancestor class or an implemented interface and if the call is in the child class.

Example in Java:

Interface ICaller

```
public interface ICaller{
    public static String _TABLE_NAME = "MyTable";
}
```

class Caller

```
public class Caller implements ICaller{
        public int execQuery(){
                String query= "select * from " + _TABLE_NAME +";";
                connection.executeQuery(query);
                ....
        }
}
```

Here, a link is expected between method execQuery and table TABLE_NAME because class Caller inherits interface ICaller, even if the reference is in the interface.

So, even if you do not see a reference to the callee in the caller source file, you have to :

1. Find out the list of ancestors of the caller class: Please refer to How to find out the list of ancestors of a class
2. Look for references to the callee in all the source files for each ancestor of your list :
   In Java, the source files have the same name as the class or interface they contain: you will find interface IGrandParentInterface in file IGrandParentInterface.java and class Parent in file Parent.java
   If you find a reference to the callee in the source files, the link is **EXPECTED**. If not, the link is **NOT EXPECTED**

ii. **For J2EE:** If the caller is a **Java method** or a **Java constructor**, and the callee is a **Java Class**, it may be referred to implicitly by **Dependency Injection** as in the example **Spring** or **CDI** frameworks. Check whether these frameworks are used in the source code by following How to check whether a framework is used by an application
   1. Find out all the interfaces implemented by the callee by following How to find out the list of ancestors of a class and consider only the interfaces
   2. Look for references to any of the interface list in the caller source file
      a. If you find a reference to any of the interface lists in the caller source file then the link is **EXPECTED.** In this case, also check Spring configuration or CDI configuration in Validate Configuration
      b. If you don't find a reference, then the link is **NOT EXPECTED**

Check configuration

Validate Configuration

This step depends a lot on the technologies involved in your missing/wrong link. Here we give a few general guidelines before we dive into technology-specific details

1. **Overlapping analysis units**
   When an object is analyzed twice because it belongs to several analysis units, then it is considered as a duplicate and may not get saved into KB at the end of the analysis. This can lead to missing/wrong links. An object can belong to several AU:

   a. If the project path of the AU is overlapping with the project path of a different AU, check if there are any overlapping analysis units in relation to the AU project path, by running queries in SQL Query - CAST Management Base - How to check for overlapping analysis units by technology
   b. If the source code path from AU 'Source settings' path is overlapping with the 'Source settings' path of a different AU, check if there are any overlapping source paths, by running queries in SQL Queries - CAST Management Base - How to check for overlapping source folders by technology

2. **Missing or unwanted dependencies**
    a. Check that the dependencies at each of the following levels are correctly configured in this order:
        i. dependencies between analysis units
        ii. dependencies between technologies
    b. Check the following:
        i. No self-referring dependency at the technology level should exist like for example a J2EE to J2EE dependency, since wrong links can be generated.
        ii. Starting from version 8.0.0, with the objective of avoiding wrong links, dependencies between analysis units are mandatory if you expect links between them. In case of the missing link, check that the analysis unit which contains the caller has a dependency towards the analysis unit which contains the callee.
            1. If yes, then the dependencies configuration is correct
            2. If not, then this could indicate an issue on the DMT level since in most cases dependencies are automatically discovered during source code delivery. In this case, first, check for any DMT alerts on Delivery Manager Tool - Package - Validation - Alerts

3. **Escalated links**
   In order to create escalated links which are indirect links between two objects and can be seen in CAST Enlighten, you should check the escalate links option in CMS - Common Technology options

4. **Technology specific configuration options**

    a. **J2EE : Framework options (tab Analysis)**
        i. Refer to CMS Snapshot Analysis - Run Analyzer - Information - How to find out if a framework is used in an application
        ii. If the application uses such supported frameworks as Spring, Struts, or Hibernate, the corresponding options must be activated and set to the right version

        ? Unknown Attachment

    b. **.NET : Framework extensions**
        i. Refer to CMS Snapshot Analysis - Run Analyzer - Information - How to find out if a framework is used in an application
        ii. If the application uses such supported frameworks as WCF or Entity Framework, then the corresponding Extensions should be properly installed.

5. **Check if the wrong link is created by Reference Finder or KB Update Tool**
    a. Run the following query on your local database, using the ObjectID of the caller and called objects:

Wrong link due to external tool

```
SELECT kPro.keynam AS PROJECT_NAME,
       t.typnam    AS PROJECT_TYPE
FROM   KEYS kPro
       JOIN TYP t
       ON     t.idtyp = kPro.objtyp
       JOIN ACC a
       ON     a.idpro = kPro.idkey
WHERE  a.idclr        =<CALLER_ID>
AND    a.idCle        =<CALLEE_ID>;
```

    b. The Project Name and Project Type value contain information on whether the link was created by Reference Finder(in Dependencies or as a search string) or KB Update Tool. For example, possible values that the above query can return are:

       "src_2b8fc99b";"JSP_APP" Link created by JSP analyser
       "_src_2b8fc99b_";"JV_PROJECT" Link created by Java analyser
       "AppViewer Analysis";"APPVIEWER_ANA" Link created by application like Enlighten
       "JavascriptUA_51130_HTML5_com.castsoftware.html5";"CAST_PluginProject" Link created by HTML5 plugin
       "JSP_Controller_SearchString_51551_src_2b8fc99b_ReferenceFinder";"Cast_ReferenceFinder_ProjectForLinks" Link created by Reference Finder
       "SEARCH_STRING_TO_CONTROLLERMETHOD_51614";"universalProject" Link created by Reference Finder search string
       "CONTROLLER_CLASS_TO_METHODS_51628";"universalProject" Link created by KB Update Tool
    c. If the link is created by a tool, then it is expected behavior. You should refine your search criteria /target as described in CMS - Dependencies - using Reference Patterns or in the Update CAST Knowledge Base Tool page

6. **Check if wrong links are Dynamic Links**
   Dynamic links can be reviewed manually or automatically
   For automatic review: There is an extension automaticlinksvalidator that has to be installed in your bases for automatic validation of link, if you still have an issue after installing it regarding the wrong links then you can report to CAST Technical support.
   For manual review: You need to open DLM and select the links which have to be manually reviewed and review it.

Check the approriate log file

Check the appropriate log file and remediate errors or warnings

1. The first step is to acknowledge a link as **EXTERNAL** or **INTERNAL**

    a. If the caller and the callee belong to different technologies (for example, the caller is a Java method and callee is an Oracle Stored Procedure), then the link is **EXTERNAL**
    b. If the caller and callee belong to the same technology, then run the following query on the local(KB) database:

```
SELECT kClr.idkey as CALLER_ID, kCle.idkey as CALLEE_ID, a.prop as INTERNAL_EXTERNAL
FROM ACC a
JOIN KEYS kClr
 ON kClr.idkey = a.idclr
JOIN keys kCle
 ON kCle.idkey = a.idcle
WHERE kClr.idkey='<CALLER_OBJECT_ID>'
AND kCle.idkey='<CALLEE_OBJECT_ID>';
```

        i. If the above query returned a result, then look at the value in column **INTERNAL_EXTERNAL : 0 means internal, 1 means external**
        ii. If the above query did not return a result, then you have to consider whether the objects belong to the same execution units. Run the following query:

```
SELECT apClr.idjob
FROM ANAPRO apClr
JOIN OBJPRO opClr ON opClr.idpro = apClr.idpro AND opClr.prop = 0
JOIN ANAPRO apCle ON apCle.idjob=apClr.idjob
JOIN OBJPRO opCle ON opCle.idpro = apCle.idpro AND opCle.prop = 0
WHERE opClr.idobj=<CALLER_ID>
AND opCle.idobj=<CALLEE_ID>;
```

        If this query returns at least one row, then the expected link is **INTERNAL**, else it is **EXTERNAL**

2. Locate the analysis log file or if the technology is an extension you may also have to locate the "Running extensions at application level" log

    a. If the link is **INTERNAL**, then you should look for warnings about the called object in the analysis log file. Any warning message containing the called object name is a good candidate to be the root cause of the absence of the link.
    You should then refer to CMS Snapshot Analysis - Run Analyzer - Warnings
    b. If the link is **EXTERNAL**, then you should look for warnings or errors at the External Link Resolution step, at the end of analysis CMS Snapshot Analysis - Run Analysis for Technology - Error Free Logs - CAST AIP 8
    Any warning message containing the called object name is a good candidate to be the root cause of the absence of the link.

Consider KB Corruption

Evaluate KB Corruption and the cost of moving to a fresh Knowledge Base

Having false links or missing links does not mean that your Knowledge Base is corrupted. It can come from configuration or product limitation and can easily be re-mediated.

One more concerning issue is inconsistencies in the Knowledge Base. This may be due to migration or some other reason, which can cause the missing or wrong link.

You can check the Knowledge Base by running the below queries:

```
SELECT * FROM ACCESSBIT
SELECT * FROM LinkTyp

select *
from Typ t
join TypCat tc
on tc.IdTyp = t.IdTyp
where tc.IdTyp = <object_type>
and tc.IdCatParent = <object_parent_type>;

select *
from Prop p
join PropCat pc
on p.idProp = pc.IdProp
where pc.IdCat = <object_category>
and pc.IdProp = <object_property>;
```

If any of the above queries return no rows then follow the steps below. If it returns rows then go to the next step to check the KB corruption.

1. Run **Load Metamodel** (**Open ServMan -> Right-click on the KB -> Load metamodel from disk**) from Server Manager (SRV - Load MetaModel)
2. Component reinstall.

To Look for corruptions in the Knowledge base

1. Run the following queries
    a. Run the following query Count incoherent link in the FusAcc table
    b. Run the following query Count incoherent link in the AccRaw table
    c. Run the following query Count Objects with links to Keys Objects in the Acc table but the links are missing in the Keys table
2. If any one of the above queries does not return **0** as a result, then the knowledge base is corrupted.

Remediation via Reference Finder

Remediation using tools

In case the remediation via Text Replacement is not possible, other tools are available.

1. In case of missing links
    a. You may use a Reference Finder with dependencies. This solution is appropriate when the objects already exist and the object name of the calle, corresponds to a regular expression that can be found in the caller source code
    b. You may use a KB Update Tool with table CI_LINKS to create the missing links. This solution is appropriate when the objects already exist and the links can be identified from some property (for instance object type) that can be found in the local database tables.
    c. You may use a Reference Pattern Search String combined with a KB Update Tool . This solution is appropriate when there are missing intermediate objects that need to be created in order to create links between the caller and the callee
2. In case of wrong links :
    a. You may use a KB Update Tool with table CI_NO_LINKS to delete the wrong links.

Impact of missing default links

The diagnostics that measure interactions among objects are impacted. The list of diagnostic depends on the type of the link (call, import, use, inheritance etc.) and the technology. Below is some example of impacted diagnostics:

1. Missing call link
    a. PB: Avoid having unreferenced code
    b. PL-SQL: Avoid unreferenced Functions / Procedures
    c. C: Avoid unreferenced Files
    d. C: Avoid unreferenced Data Members
    e. C: Avoid unreferenced Function
    f. Java: Avoid using 'System.gc'
    g. .NET: Avoid unreferenced Classes
    h. COBOL: Avoid cyclic calls with PERFORM statementsMissisng call and inheritance links
    i. .NET: Avoid cyclical calls and inheritances between namespace
    j. Java: Avoid cyclical calls and inheritances between packages
2. Missing Import link

    a. Java: Avoid Unused ImportsMulti technology
    b. Avoid Artifacts with High Fan-In
    c. Automated IFPUG Function Points Estimation

Impact of missing belongs to links

The diagnostics that measures behaviors of artifacts according to their parents' attributes are impacted. For instance the diagnostics

1. .NET: Declare as Static all methods not using instance fields
2. .NET: Provide a private default constructor for utility classes
3. Java: Avoid declaring Non Final Class Variables with Public or Package access type
4. Java: Avoid to use this within constructor in multi-thread environment
5. Struts: Avoid instance and static field in an Action Class
6. Servlet: Avoid instance and static field in a Servlet Class

Impact of wrong default links

.NET : If the wrong link is a call from a C# method or VB method to a SQL table, then it leads to false violation : "Data access must be based on Stored Procedure Calls"

Web Technologies : If a local variable has the same name as a function or a method (in JavaScript, Jscript, vbscript), the analyser will create a false link from the function (or method) containing the local variable to the function (or method) having the same name as the variable. For more details refer this page

This is a known limitation, so you may use a KB Update Tool to suppress the false link and create the real one.

Relevant Input

1. Description of the expectation regarding why the link is expected/wrong according to you.
2. A screenshot in Enlighten showing the full object names(can be found in object properties F11), the objects types and the expected link type for missing link OR link type for wrong link.
3. *CAST Support Tool (CST) - alias Sherlock with the options* ***Logs*** *and* ***CAST Management and Local Databases*** *Checked*
   In the general case, only the file containing the caller and the file containing the callee are required. Please verify the complementary input in the specific technology pages that can be found in:
   CMS Snapshot Analysis - Run Analyzer - Missing or Wrong Links per technology. Here are some complementary files for special cases :
    a. If the technology of the caller is C++, also provide preprocessed file resulting of the analysis of the Caller source file (you will need to run an analysis with the caller source file only).
    b. If the technology of the caller or callee is Java or .Net, also provide the files containing the lists of ancestors of the class of the caller and of the class of the calee: Please refer to CMS Snapshot Analysis - Run Analyzer - J2EE - Information - How to find out the list of ancestors of a class and to CMS Snapshot Analysis - Run Analyzer - DOTNet - Information - How to find out the list of ancestors of a class

Related Pages

CMS Snapshot Analysis - Run Analyzer - J2EE - Information - How to find out the list of ancestors of a class

CMS Snapshot Analysis - Run Analyzer - Information - How to find out if a framework is used in an application

SQL Query - CAST Management Base - How to check for overlapping analysis units by technology

Notes / Comments

Ticket # 16980