.NET

On this page:

- Detailed technology support
  - C# / VB.NET
- Required third-party software
- .NET objects
- CAST Delivery Manager Tool
  - Discovery and extraction
  - What you should package?
  - Information extracted
    - C# and VB.NET
    - C# specific
    - VB.NET specific
- Notes
  - Supported syntax for Visual Studio 2015/.NET Framework 4.6
  - ASP.NET web applications
  - JavaScript
  - Silverlight
  - Language Integrated Query (LINQ)
    - LINQ to Objects
    - LINQ to DataSets (from CAST AIP 8.2.2 only)
  - .NET WebServices
    - Links from web front end > WebService > back-end database
      - Scenario 1
      - Scenario 2
      - Scenario 3
      - Scenario 4
    - WBS Linker Extension
      - Check if the extension is already installed
      - If the WBS Linker Extension is already installed
      - If the WBS Linker Extension is not installed
  - Generated code
  - COM objects
  - Miscellaneous
  - .NET Core
- What to expect when upgrading to AIP 8.2.x from 7.3.x and using the new .NET analyzer
  - Upgrade
  - Technical differences
  - GUI differences
    - CAST Delivery Manager Tool
    - CAST Management Studio
      - Execution Units
  - Metamodel type changes for VB.NET types
  - Discovery and extraction in the CAST Delivery Manager Tool
    - Packages
  - Analysis results
    - ASP.NET
      - Improvements for ASP.NET directives support
      - Improvements for web.config support
    - CAST Engineering Dashboard
    - Improved handling of .NET dependencies
      - Framework assemblies
      - Assembly dependencies
    - Namespace objects

Target audience:

CAST Administrators

Summary: this page provides detailed information about CAST's support for the .NET technology.

Detailed technology support

C# / VB.NET

Visual Studio version	.NET Framework version	C# version	VB.NET version	Supported

2003	1.1	1.2	7.1	✓
2005	2.0, 3.0	2.0	8.0	✓
2008	2.0, 3.0, 3.5	3.0	9.0	✓
2010	2.0, 3.0, 3.5, 4.0	4.0	10.0	✓
2012/2013	2.0, 3.0, 3.5, 4.0, 4.5, 4.5.1, 4.5.2	5.0	12.0	✓
2015	2.0, 3.0, 3.5, 4.0, 4.5, 4.5.1, 4.5.2, 4.6*	6.0	14.0	✓

*Note that the **.NET Framework 4.6** can be analyzed but not all syntax provided in this version is supported. See below for a list of supported syntax.

Required third-party software

Note that if the CAST Delivery Manager Tool (DMT) is being run (whether from the CAST AIC Portal or direct from the CAST Management Studio (CMS)) from the SAME workstation as the one used for CMS to run the analysis/snapshot generation, then any IDEs or APIs will need to be installed on the workstation running CMS so that the DMT can access them.

To successfully deliver and analyze .NET code, the following third-party software is required:

Install on workstation running the DMT (for extraction)	Install on workstation running CMS (for analysis)	Remarks
The **.NET framework must be installed** so that you can extract and package any system assemblies required by your applications' source code. You must install: • Either the same version of the framework that is used by your application • Or a more recent version of the framework (for example 4.6 when the application uses 3.5) If your application uses **multiple framework versions**, you can install the most recent one used by your application or a newer version if it is available on the workstation - this will match all projects. For example, an application uses frameworks 3.5 and 4.0: you can install 4.0 or a newer version.	**Not required**	**CMS** Note that a CAST .NET framework is required to run the .NET analyzer in the CAST Management Studio, however, this is installed as part of the CAST AIP setup (see Install CAST AIP components).

.NET objects

The following section lists the **objects** that the .NET analyzer is capable of detecting and storing in the CAST Analysis Service:

Icon	Name	Parent
	Project	-
	Application	Project
	Directory	Application
	C# Source File	Application, Directory
	VB.NET Source File	Application, Directory
	ASA / Asax Source File	Application, Directory
	ASP Source File	Application, Directory

	Aspx Source File	Application, Directory
	Ascx Source File	Application, Directory
	Asmx Source File	Application, Directory
	Configuration Source File	Application, Directory
	Data Set Source File	Application, Directory
	Htc / Htm / Html Source File	Application, Directory
	Java Script Source File	Application, Directory
	Master Source File	Application, Directory
	Settings Source File	Application, Directory
	Skin Source File	Application, Directory
	Xaml Source File	Application, Directory
	Visual Basic Script File	Application, Directory
	.NET AppSetting found in configuration file	Configuration Source File
	Class	Namespace, Interface, Class, Generic Class
	Constant (field)	Class, Generic Class
	Delegate	Class, Generic Class
	Enumeration	Class, Generic Class
	Enumeration Item	Enumeration
	Event	Class, Generic Class
	Event Add On	Event
	Event Remove On	Event
	Interface	Namespace, Interface, Class, Generic Class
	Field (non constant)	Interface, Class, Generic Class
-	Finalizer	Class, Generic Class
	Generic Class	Namespace, Interface, Class, Generic Class
	Generic Method	Generic Class
	Generic Property	Generic Class
	Indexer	Method, Generic Method
	Indexer Getter	Interface, Class, Generic Class
	Indexer Setter	Interface, Class, Generic Class
	Method	Interface, Class, Generic Class
	Module (VB.NET only)	-

{ }	Namespace	Application, Namespace
	Operator	Interface, Class, Generic Class
	Property	Interface, Class, Generic Class
	Property Getter	Interface, Class, Generic Class
	Property Setter	Interface, Class, Generic Class
	Structure	Interface, Class, Generic Class, Method, Generic Method

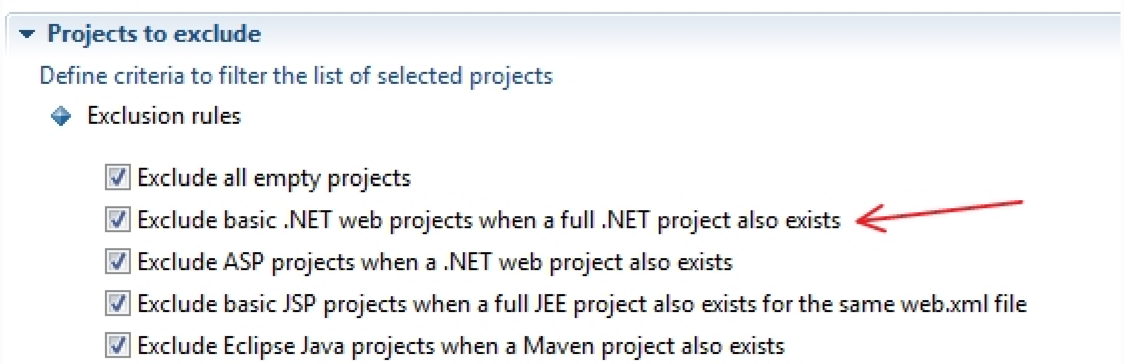
CAST Delivery Manager Tool

Discovery and extraction

The discovery and extraction of your application's source code via the CAST Delivery Manager Tool will function as follows (an **Analysis Unit** will be created for **each project** that has been identified during the Discovery process and is not excluded by a rule or filter):

C#/VB.NET project support	Configures one project for each Microsoft Visual Studio C# project (***.csproj** file) or VB.NET project (***.vbproj** file) identified.
C#/VB.NET Web Site support	Configures one project for each Microsoft Visual Studio C#/VB.NET Web Site identified. In order to be identified as a Web Site project, the root folder must contain a **web.config** file. If a .csproj or .vbproj file is found in a sub-folder of the root Web Site project folder, a corresponding C# or VB.NET project will be created and the sub-folder will not be included in the Web Site delivery.

Note that an exclusion rule is active by default, which will force basic .NET web site projects to be discovered but **ignored** when a full .NET project exists:



What you should package?

When creating packages to discover and extract your .NET application you should create them as follows:

Package	Package name /type	Mandatory?	Location/path	Notes
1	Source code	✓	Source code root folder	Use the "Files on your file system" / SVN / TFS options in the CAST Delivery Manager Tool.
2	External assemblies (third party DLL)	If required, one package per vendor.	N/A: depends on the third party	Use the "Automated extraction of .NET assemblies on your file system" option in the CAST Delivery Manager Tool. **When to create this package** This package should only be created when your application uses third party assemblies.

3	.NET framework (system assemblies)	✓	C:\Windows\Microsoft.NET\Framework64\<use the version used by your projects>	Use the "Automated extraction of .NET assemblies on your file system" option in the CAST Delivery Manager Tool. If "C:\Windows\Microsoft.NET\Framework64\<use the version used by your projects>" is NOT available on the machine, please use the most recent framework installed on the machine instead: "C:\Windows\Microsoft.NET\Framework64\<use the most recent>". Unlike in previous releases of CAST AIP, the .NET analyzer no longer tries to extract the .NET framework assemblies during the analysis: this is why a specific package for external assemblies must be created in the CAST Delivery Manager Tool. If this package is not created and delivered, **the .NET analyzer will skip all projects** having a dependency to .NET assemblies, leading to incomplete analysis results.
4	Custom assemblies	If required (typically when there are unresolved alerts)	Source code root folder (or "bin" sub-folder)	Use the "Automated extraction of .NET assemblies on your file system" option in the CAST Delivery Manager Tool. **When to create this package** • When the custom assemblies are stored in the correct location as specified in the project definition, then package #1 will retrieve them. In this case no alert will be raised by the CAST Delivery Manager Tool about missing assemblies, so in this case, **there is no need** to create package #4. • When the custom assemblies are not stored in the correct location as specified in the project definition, then package #1 will not be able to retrieve them. "Missing assemblies" alerts will be raised for package #1, so the creation of package #4 for custom assemblies located either in the source code root folder or in the "bin" sub-folder **is required** to clear these alerts.

Please avoid **accidentally duplicating source code** as it might considerably increase analysis time during the "linker" phase. When CAST schemas are hosted on an Oracle Server, the analysis might not complete at all. Source code can easily get duplicated by accident when upgrading Visual Studio. Indeed, Visual Studio automatically creates a backup folder inside the project folder. This backup folder contains a copy of the project file (.csproj, .vbproj) which will lead to duplicate analysis of the source code if this folder is delivered via the CAST Delivery Manager Tool - you should therefore ensure that the backup folder is excluded from delivery.

Information extracted

C# and VB.NET

- Source files and their associated **BuildAction** (e.g. Compile, Content, EmbeddedResource, None, etc) - benefit: no longer rely on extensions to know which files to compile
- Assembly name (can be different from the project name)
- .NET framework version
- Compilation constants, including their values for VB.NET

C# specific

- Default namespace (the default namespace to be used when creating new files; will be used when processing .xsd files)
- Option "Allow unsafe code"

VB.NET specific

- Root namespace (the default namespace to be used instead of the global namespace)
- Option explicit (On / Off)
- Option strict (On / Off)
- Option infer (On / Off)
- Imported namespaces (namespaces that are automatically imported by each file in the project)

Notes

The following section lists technical and functional aspects with regard to analysis of .NET source code:

Please also see Web technologies for more information about web technology specifics (i.e. XHTML, JScript, JavaScript).

Supported syntax for Visual Studio 2015/.NET Framework 4.6

The following syntax is supported:

- Expression-bodied members (C#)
- Auto-property initializers (VB.NET/C#)

- nameof operator (VB.NET/C#)
- Getter-only auto-properties (VB.NET/C#)
- Ctor assignment to getter-only autoprops (VB.NET/C#)
- Static members (VB.NET/C#)
- Index initializer (C#)
- Await in catch / finally (C#)
- Partial modules (VB.NET)
- Partial interfaces (VB.NET)
- Multiline string literals (VB.NET)
- year-first date literals (VB.NET)
- Line continuation comments - those within LINQ expressions, and after implicit line continuations (VB.NET)
- TypeOf IsNot (VB.NET)
- #pragma directives (VB.NET)
- Smart Name Resolution (VB.NET)
- ReadWrite props can implement ReadOnly (VB.NET)
- #region inside methods (VB.NET/C#)
- Methods declared with the keyword Overrides also seen as Overloads (VB.NET)
- CObj in attributes (VB.NET/C#)
- CRef and parameter name in comments (VB.NET/C#)
- Extension Add in collection initializers (VB.NET/C#)
- String interpolation (VB.NET/C#)
- Exceptions filter (C#)

ASP.NET web applications

- Unused files inside of web site application folders are ignored during the analysis.

JavaScript

The .NET analyzer supports the analysis of JavaScript delivered with the .NET project files. Please see the JavaScript section in Web technologies.

Silverlight

Silverlight is supported through a CAST AIP Extension, however, .NET projects are only recognized as Silverlight projects if their CSPROJ file contains the following XML tag:

```
<TargetFrameworkIdentifier>Silverlight</TargetFrameworkIdentifier>
```

If a project is a Silverlight project but it does not contain this tag, then during the packaging process with the CAST Delivery Manager Tool alerts may be generated indicating missing .NET libraries and assemblies. These alerts may persist even though the required files are present in one or more .NET packages. To avoid this issue, ensure that the required XML tag is present in the source code and then repackage your packages.

Language Integrated Query (LINQ)

Current support of **LINQ** is limited to the **LINQ to Objects** and **LINQ to DataSets** (from **CAST AIP 8.2.2** only) providers only. No other provider is supported.

LINQ to Objects

With regard to LINQ to Objects, in the following example:

- A link will be created to the object "customer" (if it is not declared as a local variable)
- A link will be created to the field "city"

```
var queryLondonCustomers = from cust in customers
        where cust.City == "London"
        select cust;
```

Note that currently the following is NOT supported:

- Call to LINQ extension methods, such as the Where method, for example:

```
public static IEnumerable<TSource> Where<TSource>(
        this IEnumerable<TSource> source,
        Func<TSource,bool> predicate
)
```

- Where a LINQ statement implicitly calls a LINQ extension method - no Access link is created for these calls

LINQ to DataSets (from CAST AIP 8.2.2 only)

With regard to LINQ to DataSets, in the following example:

- A **Use Select** link will be created between the "myMethod" containing the declaration of the table of the dataset and the "Product" database table:

```
// Fill the DataSet.
public void myMethod()
{
        DataSet ds = new DataSet();
        ds.Locale = CultureInfo.InvariantCulture;
        FillDataSet(ds);
        DataTable products = ds.Tables["Product"];
        IEnumerable<string> query =
            from product in products.AsEnumerable()
            select product.Field<string>("Name");
        Console.WriteLine("Product Names:");
        foreach (string productName in query)
                {
                    Console.WriteLine(productName);
                }
}
```

.NET WebServices

- **ASP.NET WebServices** are supported "out of the box" by the .NET analyzer as standard C#/VB.NET projects.
- **WCF WebServices** are supported through the separate installation of a CAST AIP extension - **WCF Support for C# and VB.NET** see http://doc.castsoftware.com/display/DOCEXT/For+.NET for more information.

Links from web front end > WebService > back-end database

If the entire transaction from a **web front end** > **WebService** > **back-end database** needs to be resolved, then further configuration is required to support this, as outlined below:

> Please note that if the web front end is an ASP.NET "website" (i.e. no project file (.csproj or .vbproj) exists), then no links will be created from this website to the WebServices. This is a limitation in the .NET analyzer.

Scenario 1

If your web front end is written in **ASP.NET** and you use **ASP.NET WebServices**, then:

- The .NET analyzer will handle the analysis of both the **front end web application** and the **WebServices**
- You will need to install, separately, the **WBS Linker Extension** to resolve the full transaction from web application to database. See below for more information about this.

In this scenario, the following will be resolved:

- In C# and VB.NET projects the web application counter part of ASP.NET WebServices technology is called a "Web Reference" object. Links will be created from the Web Reference objects to ASP.NET WebServices. ASP.NET WebServices may also be targeted by some other technologies.
- A call link will be created from the proxy method (a C# or VB.NET method having the attribute System.Web.Services.Protocols.SoapDocumentMethodAttribute) to an object typed ".NET SOAP service reference". These kind of objects are created for each proxy method, their name is the URL of the targeted web method.
- For the database side, an object typed ".NET SOAP operation" will be created, from which is drawn a call link to the web method (a C# or VB.NET method having the attribute System.Web.Services.WebMethodAttribute). An object ".NET SOAP operation" is created for each web method, their name is the URL of the web method.
- The **WBS Linker Extension** will create a call link from ".NET SOAP service reference" objects to their matching database counterpart ".NET SOAP operation" object.

Scenario 2

If your web front end is written in **ASP.NET** and you use **WCF WebServices**, then:

- The .NET analyzer will handle the analysis of the **front end web application only**
- You will need to install, separately, the WCF Support for C# and VB.NET Extension to handle the **WCF WebServices** - doing so will mean you will automatically get the **WBS Linker Extension** as a dependency to the **WCF Support for C# and VB.NET Extension** to resolve the full transaction from web application to database

Scenario 3

If your web front end is written in some other language (such as **HTML5/AngularJS**) and you use **ASP.NET WebServices**, then:

- You will need to install the appropriate extension to handle the web front end code - see http://doc.castsoftware.com/display/DOCEXT/CAST+AIP+Extensions+Documentation
- The .NET analyzer will handle the analysis of the **WebServices**
- You will need to install, separately, the **WBS Linker Extension** to resolve the full transaction from web application to database. See below for more information about this.

Scenario 4

If your web front end is written in some other language (such as **HTML5/AngularJS**) and you use **WCF WebServices**, then:

- You will need to install the appropriate extension to handle the web front end code - see http://doc.castsoftware.com/display/DOCEXT/CAST+AIP+Extensions+Documentation
- You will need to install, separately, the WCF Support for C# and VB.NET Extension to handle the **WCF WebServices** - doing so will mean you will automatically get the **WBS Linker Extension** as a dependency to the **WCF Support for C# and VB.NET Extension** to resolve the full

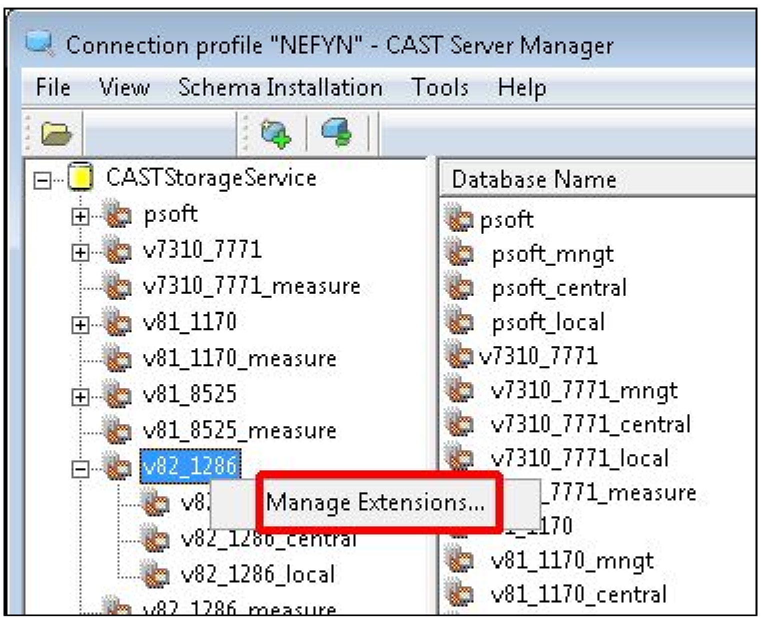
WBS Linker Extension

In **scenarios 1 and 3**, you will need to install, separately, the **WBS Linker Extension** to resolve the full transaction from web application to database. This extension is normally provided as dependency to other standard CAST AIP extensions and is therefore undocumented. If your application source code falls into scenarios 1 or 3, this is what you need to do:

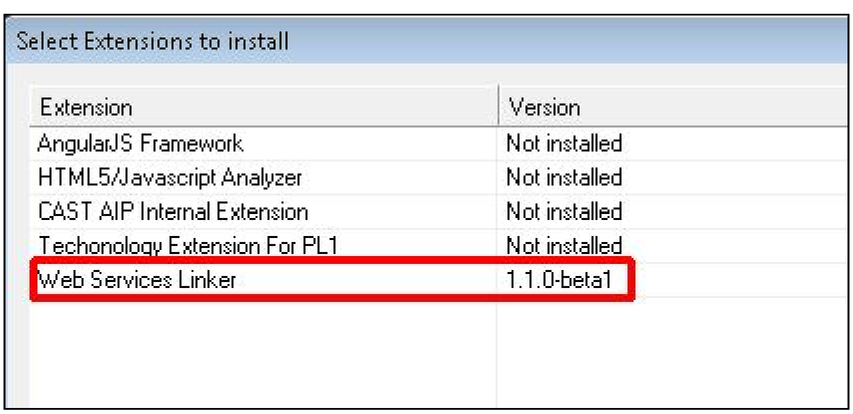
Check if the extension is already installed

First check whether the **WBS Linker Extension** is already installed in your CAST AIP schema triplet. To do so:

- Launch CAST Server Manager
- Right click the root item in your triplet and then select **Manage Extensions**:



- A new dialog will be displayed - the **WBS Linker Extension** is **already installed** if anything other "**Not installed"** is displayed as shown below:



If the WBS Linker Extension is already installed

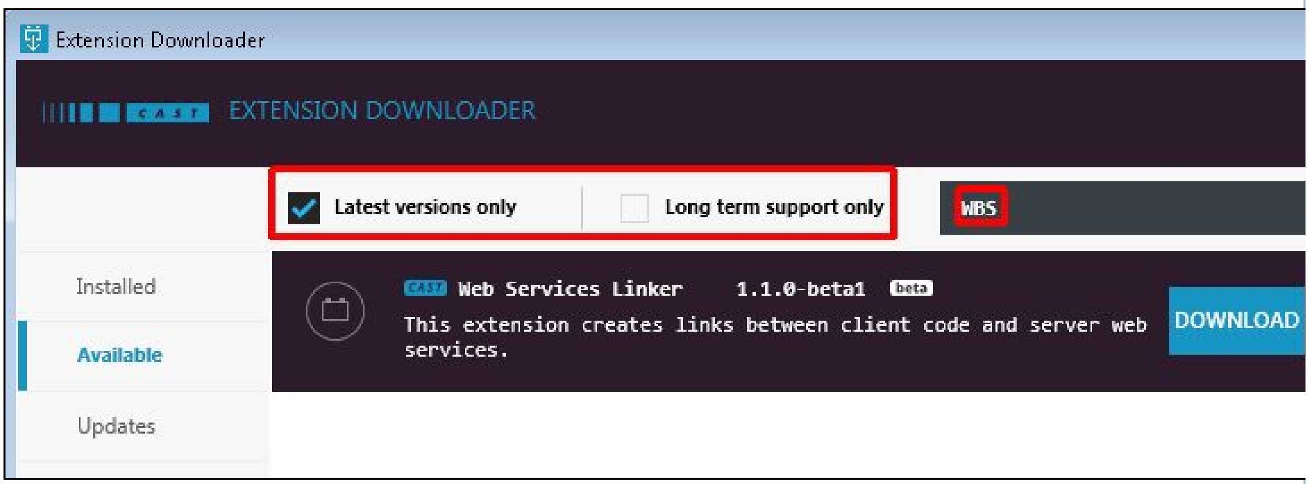
There is nothing more to configure. Running an analysis will resolve the full transaction from web application to database.

If the WBS Linker Extension is not installed

If the WBS Linker Extension is **not installed**, you need to **download** it (with the CAST Extension Downloader) and then **install** it (with CAST Server Manager). This process is explained in more detail here:

- http://doc.castsoftware.com/display/EXTEND/Download+an+extension
- http://doc.castsoftware.com/display/EXTEND/Install+an+extension

Note that when downloading the extension using the CAST Extension Downloader, you must ensure that you untick the "Long term support only" option, and then search for "WBS" as shown in the image below:



Generated code

The .NET analyzer handles auto generated code like all other CAST AIP analyzers:

- Auto generated code is analyzed to help understand the entire code being analyzed
- Objects are created from the code and saved in the CAST Analysis Service schema (to help trace transactions for example) and these objects are marked as being "generated"
- Any Quality Rule violations that are caused by these "generated" objects do not contribute to grade calculations
- "Generated" objects are excluded from any aggregated metrics (for example Lines of Code (LOC))

The .NET analyzer determines whether code is auto generated using a combination of the following factors:

Files that end with the following:

- "_CastGenerated.cs"
- "_CastGenerated.vb"
- ".Designer.cs"
- ".Designer.vb"
- "Reference.cs"
- "Reference.vb"

Note that to be considered "generated", files that end with **"Reference.cs"** and **"Reference.vb**" MUST ALSO be located in one of the following folders:

- "Web References"
- "Service References"

Symbols marked with one of the following attributes:

- "System.CodeDom.Compiler.GeneratedCodeAttribute"
- "System.Diagnostics.DebuggerNonUserCodeAttribute"

The following object types:

- Accessors for auto-generated properties
- Implicitly generated functions (constructor by default)

- Anonymous types
- Instances of generic types or methods
- Objects with no bookmark in the source code

Files that contain top level comments that include the following:

- "<autogenerated>"
- "<auto-generated>"

COM objects

The CAST .NET analyzer does not support the resolution of COM objects (such as ADODB) referenced in .NET projects.

Miscellaneous

- If a .NET project (either C# or VB.NET) depends on an assembly (or a project) that declares a class with the same name full name (i.e. same name, in the same namespace) as a class in said project, then that class is not saved to the Analysis Service.
- XML code embedded into VB is not taken into account. The code will be analyzed, but its contents will be ignored.
- Note that **.exe** and **.netmodules** dependencies are not supported.
- If a DLL file is placed in the "bin" folder of a .NET project, but its exact location is not defined in the .csproj or .vbproj project file, the CAST Delivery Manager Tool may not be able to identify the reference during the package action. To resolve this issue, you may need to create an additional package using the **Automated extraction or required .NET assemblies** option specifically to package the DLLs in the "bin folder.
- Please avoid **accidentally duplicating source code** as it might considerably increase analysis time during the "linker" phase. When CAST schemas are hosted on an Oracle Server, the analysis might not complete at all. Source code can easily get duplicated by accident when upgrading Visual Studio. Indeed, Visual Studio automatically creates a backup folder inside the project folder. This backup folder contains a copy of the project file (.csproj, .vbproj) which will lead to duplicate analysis of the source code if this folder is delivered via the CAST Delivery Manager Tool - you should therefore ensure that the backup folder is excluded from delivery.

.NET Core

.NET Core is **not supported** by the .NET Analyzer.

What to expect when upgrading to AIP 8.2.x from 7.3.x and using the new .NET analyzer

In CAST AIP 8.0.0, CAST introduced a brand new .NET analyzer to replace the two existing analyzers for VB.NET and C# provided in CAST AIP 7.3.x. This analyzer has been redesigned and written from the ground up to provide support for VB.NET and C# at the same time. This section of documentation lists the major differences that you should expect if you have upgraded to AIP 8.2.x from 7.3.x and are starting to use the new .NET analyzer with your existing .NET applications.

Upgrade

Please see Upgrading to CAST AIP 8.2.x for more information about upgrading from previous releases of CAST AIP with .NET applications (C# and VB.NET).

Technical differences

Technically, the new .NET analyzer is radically different than the .NET analyzer shipped with previous releases of CAST AIP. Some of the fundamental technical differences are listed below:

- Based on the open source Roslyn .NET Compiler Platform.
- Requires **uncompiled source code** for VB.NET and C# rather than **compiled source code** (as is the case for **existing analyzer for VB.NET**).
- Web site applications built with ASP.NET no longer need to be deployed in IIS or in the CAST Management Studio embedded web server for compilation before analysis. Web site applications can therefore be delivered uncompiled and the analysis will proceed just like any other .NET application.
- Uses a **64bit architecture** to take advantage of increased memory utilisation during an analysis and therefore increase performance.
- Execution Units are no longer required due to the use of 64bit technology. This will improve accuracy of results and removes a complex technical constraint introduced to work around the limitations of the 32bit architecture used in the .NET analyzer in previous releases of CAST AIP.
- Support for the following .NET technologies is now provided through a **CAST AIP extension** rather than being handled by the **.NET analyzer**. As such, if you used the CAST .NET analyzer in previous releases of CAST AIP to analyze .NET applications containing WPF/WCF/Silverlight source code, after an upgrade to CAST AIP 8.2.x and the generation of a post upgrade snapshot, you will find that the WPF/WCF/Silverlight objects will **not be present in the analysis results** and links to these objects from other .NET code will also **not be present**. To resolve this issue, you must install the WPF/WCF/Silverlight extensions and re-analyze your .NET applications (please see http://doc.castsoftware.com/display/DOCEXT/For+.NET for more information about the extensions):
    - **WPF** (Windows Presentation Framework)
    - **WCF** (Windows Communication Foundation)
    - **Silverlight**

GUI differences

CAST Delivery Manager Tool

None.

CAST Management Studio

Options removed:

- The option "**Analyzer > Analyzer to invoke**" (visible at Application and Analysis Unit level in the Analysis tab) has been removed as it is no longer required: there is one analyzer for both .NET source code types.
- The option "**Embedded web server > Port number**" (visible at Application and Analysis Unit level in the Analysis tab specifically for ASP.NET applications that required compilation prior to analysis) has been removed as it is not longer required: web site applications built with ASP.NET are handled without any special configuration.
- The option "**Process Settings - Execution > Memory Threshold for Instances**" (visible at Technology level and at Application level in the Analysis tab) has been removed as it is no longer required: the use of 64bit technology renders this option obsolete.
- The option "**Process Settings - Execution > Execution Unit max size (MB)**" (visible at Technology level) has been removed as it is no longer required.
- The option "**Analysis Unit implementation > Is Application entry point**" (visible at Analysis Unit level in the Execution tab) has been removed as it is no longer required.

Options added:

- A new option "**References Assemblies > External assemblies**" (visible at Analysis Unit level in the Analysis tab) has been added. This is specifically to allow external assemblies to be added to the Analysis Unit if the source code delivery from the CAST Delivery Manager Tool is incomplete and does not contain them.

Execution Units

The ability to manually manage Execution Units has been removed (and with it the corresponding options): the use of 64bit technology renders the manual configuration of Execution Units obsolete. One Execution Unit is now always created per Analysis Unit. It is still possible to set an annotation on an Analysis Unit to manually group Analysis Units into one Execution Unit, but CAST highly recommends that the analysis configuration is left entirely to the analyzer to manage.

Metamodel type changes for VB.NET types

In CAST AIP 8.1.x, the type names used by CAST AIP to define all **VB.NET objects** have been modified and have been given new values, therefore, any configuration (for example in the CAST Transaction Configuration Center - see **.NET analyzer specifics for the CAST Transaction Configuration Center** in Upgrading to CAST AIP 8.2.x) that specifically uses these type names will need to be reviewed. The following table lists the type names in previous releases of CAST AIP, and the corresponding new type name

Legacy VB.NET type name	New VB.NET type name
<type name="NET_WINDOWS_APPLICATION_VB" id="372"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_CLASS_LIBRARY_VB" id="373"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_WINDOWS_CONTROL_LIBRARY_VB" id="374"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_CONSOLE_APPLICATION_VB" id="375"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_WINDOWS_SERVICE_VB" id="376"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_ASPNET_WEB_APPLICATION_VB" id="377"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_ASPNET_WEB_SERVICE_VB" id="378"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_ASPNET_WEB_CONTROL_LIBRARY_VB" id="379"><attribute name="KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_ProjectVB" id="141898"><attribute name="KEY_CLASS" intValue="39350"/>
<type name="NET_CLASS_VB" id="406"><attribute name="KEY_CLASS" intValue="12090"/>	<type name="CAST_DotNet_ClassVB" id="137106"><attribute name="KEY_CLASS" intValue="39140"/>
<type name="NET_MODULE_VB" id="407"><attribute name="KEY_CLASS" intValue="12105"/>	<type name="CAST_DotNet_ModuleVB" id="137107"><attribute name="KEY_CLASS" intValue="39070"/>
<type name="NET_NAMESPACE" id="388"><attribute name="KEY_CLASS" intValue="12070"/>	<type name="CAST_DotNet_NamespaceDotNet" id="137114"><attribute name="KEY_CLASS" intValue="38010"/>

<type name="NET_WEB_CUSTOM_CONTROL" id="401"><attribute name="KEY_CLASS" intValue="12170"/>	<type name="CAST_DotNet_SkinControlType" id="137593"><attribute name="KEY_CLASS" intValue="40020"/>
<type name="NET_EVENT_RAISE" id="112003"><attribute name=" KEY_CLASS" intValue="12345"/>	<type name="CAST_DotNet_EventRaiseVB" id="141799"><attribute name="KEY_CLASS" intValue="39215"/>
<type name="NET_ASPNET_WEB_SITE" id="974"><attribute name=" KEY_CLASS" intValue="12040"/>	<type name="CAST_DotNet_AspNetWebSite" id="137028"><attribute name="KEY_CLASS" intValue="37040"/>
<type name="NET_CLASS" id="389"><attribute name="KEY_CLASS" intValue="12090"/>	<type name="CAST_DotNet_ClassVB" id="137106"><attribute name=" KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternal" id="137088"><attribute name=" KEY_CLASS" intValue="37130"/>
<type name="NET_INTERFACE" id="390"><attribute name=" KEY_CLASS" intValue="12100"/>	<type name="CAST_DotNet_InterfaceVB" id="137102"><attribute name=" KEY_CLASS" intValue="39030"/> <type name="CAST_DotNet_InterfaceExternal" id="137089"><attribute name="KEY_CLASS" intValue="37150"/>
<type name="NET_STRUCTURE" id="391"><attribute name=" KEY_CLASS" intValue="12230"/>	<type name="CAST_DotNet_StructureVB" id="137103"><attribute name=" KEY_CLASS" intValue="39060"/> <type name="CAST_DotNet_StructureExternal" id="137090"><attribute name="KEY_CLASS" intValue="37170"/>
<type name="NET_ENUM" id="392"><attribute name="KEY_CLASS" intValue="12240"/>	<type name="CAST_DotNet_EnumerationVB" id="137104"><attribute name="KEY_CLASS" intValue="39090"/> <type name="CAST_DotNet_EnumerationExternal" id="137091"><attribute name="KEY_CLASS" intValue="37210"/>
<type name="NET_DELEGATE" id="393"><attribute name=" KEY_CLASS" intValue="12250"/>	<type name="CAST_DotNet_DelegateVB" id="137105"><attribute name=" KEY_CLASS" intValue="39110"/> <type name="CAST_DotNet_DelegateExternal" id="137092"><attribute name="KEY_CLASS" intValue="37190"/>
<type name="NET_EXCEPTION" id="394"><attribute name=" KEY_CLASS" intValue="12260"/>	<type name="CAST_DotNet_ClassVBException" id="141672"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalException" id="139130" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_COMPONENT_CLASS" id="396"><attribute name=" KEY_CLASS" intValue="12160"/>	<type name="CAST_DotNet_ClassVBComponent" id="141678"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalComponent" id="139136" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_CUSTOM_CONTROL" id="397"><attribute name=" KEY_CLASS" intValue="12130"/>	<type name="CAST_DotNet_ClassVBCustomControl" id="141677" ><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalCustomControl" id="139135" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_DATASET" id="400"><attribute name=" KEY_CLASS" intValue="12190"/>	<type name="CAST_DotNet_ClassVBDataSet" id="141674"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalDataSet" id="139132" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_WINDOWS_SERVICE_CLASS" id="403"><attribute name="KEY_CLASS" intValue="12180"/>	<type name="CAST_DotNet_ClassVBServiceBase" id="141679"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalServiceBase" id="139137" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_TRANSACTIONAL_COMPONENT" id="404" ><attribute name="KEY_CLASS" intValue="12200"/>	<type name="CAST_DotNet_ClassVBServicedComponent" id="141680" ><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalServicedComponent" id=" 139138"><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_CRYSTAL_REPORT_CLASS" id="405"><attribute name="KEY_CLASS" intValue="12220"/>	<type name="CAST_DotNet_ClassVBCrystalReportClass" id="141684" ><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalCrystalReportClass" id="139142" ><attribute name="KEY_CLASS" intValue="37130"/>

<type name="NET_PROPERTY" id="412"><attribute name="KEY_CLASS" intValue="12280"/>	<type name="CAST_DotNet_IndexerVB" id="141747"><attribute name="KEY_CLASS" intValue="39270"/> <type name="CAST_DotNet_PropertyVB" id="137109"><attribute name="KEY_CLASS" intValue="39300"/> <type name="CAST_DotNet_PropertyExternal" id="137098"><attribute name="KEY_CLASS" intValue="37310"/>
<type name="NET_PROPERTY_SETTER" id="413"><attribute name="KEY_CLASS" intValue="12290"/>	<type name="CAST_DotNet_IndexerSetterVB" id="141748"><attribute name="KEY_CLASS" intValue="39280"/> <type name="CAST_DotNet_PropertySetterVB" id="137110"><attribute name="KEY_CLASS" intValue="39210"/> <type name="CAST_DotNet_PropertySetterExternal" id="137099"><attribute name="KEY_CLASS" intValue="37330"/>
<type name="NET_PROPERTY_GETTER" id="414"><attribute name="KEY_CLASS" intValue="12300"/>	<type name="CAST_DotNet_IndexerGetterVB" id="141749"><attribute name="KEY_CLASS" intValue="39290"/> <type name="CAST_DotNet_PropertyGetterVB" id="137111"><attribute name="KEY_CLASS" intValue="39320"/> <type name="CAST_DotNet_PropertyGetterExternal" id="137100"><attribute name="KEY_CLASS" intValue="37320"/>
<type name="NET_EVENT" id="415"><attribute name="KEY_CLASS" intValue="12310"/>	<type name="CAST_DotNet_EventVB" id="141736"><attribute name="KEY_CLASS" intValue="39190"/> <type name="CAST_DotNet_EventExternal" id="139060"><attribute name="KEY_CLASS" intValue="37240"/>
<type name="NET_EVENT_ADDON" id="416"><attribute name="KEY_CLASS" intValue="12320"/>	<type name="CAST_DotNet_EventAddonVB" id="141737"><attribute name="KEY_CLASS" intValue="39200"/> <type name="CAST_DotNet_EventAddExternal" id="139061"><attribute name="KEY_CLASS" intValue="37250"/>
<type name="NET_EVENT_REMOVEON" id="417"><attribute name="KEY_CLASS" intValue="12330"/>	<type name="CAST_DotNet_EventRemoveonVB" id="141738"><attribute name="KEY_CLASS" intValue="39210"/> <type name="CAST_DotNet_EventRemoveExternal" id="139062"><attribute name="KEY_CLASS" intValue="37260"/>
<type name="NET_CONST" id="419"><attribute name="KEY_CLASS" intValue="12390"/>	<type name="CAST_DotNet_ConstantVB" id="140915"><attribute name="KEY_CLASS" intValue="39180"/> <type name="CAST_DotNet_ConstantExternal" id="137096"><attribute name="KEY_CLASS" intValue="37280"/>
<type name="NET_ENUM_ITEM" id="420"><attribute name="KEY_CLASS" intValue="12380"/>	<type name="CAST_DotNet_EnumerationItemVB" id="141733"><attribute name="KEY_CLASS" intValue="39100"/> <type name="CAST_DotNet_EnumerationItemExternal" id="137097"><attribute name="KEY_CLASS" intValue="37220"/>
<type name="NET_CTOR" id="421"><attribute name="KEY_CLASS" intValue="12360"/>	<type name="CAST_DotNet_ConstructorVB" id="137112"><attribute name="KEY_CLASS" intValue="39330"/> <type name="CAST_DotNet_ConstructorExternal" id="137094"><attribute name="KEY_CLASS" intValue="37380"/>
<type name="NET_DTOR" id="422"><attribute name="KEY_CLASS" intValue="12370"/>	<type name="CAST_DotNet_DestructorVB" id="137113"><attribute name="KEY_CLASS" intValue="39340"/> <type name="CAST_DotNet_DestructorExternal" id="137095"><attribute name="KEY_CLASS" intValue="37370"/>
<type name="NET_INSTALLER_CLASS" id="433"><attribute name="KEY_CLASS" intValue="12210"/>	<type name="CAST_DotNet_ClassVBInstaller" id="141681"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalInstaller" id="139139"><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_WEB_SERVICE" id="578"><attribute name="KEY_CLASS" intValue="12264"/>	<type name="CAST_DotNet_ClassVBWebService" id="141682"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalWebService" id="139140"><attribute name="KEY_CLASS" intValue="37130"/>

<type name="NET_WEB_SERVICE_PROXY" id="579"><attribute name="KEY_CLASS" intValue="12265"/>	<type name="CAST_DotNet_ClassVBWebServiceProxy" id="141683" ><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalWebServiceProxy" id="139141" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_METHOD" id="410"><attribute name="KEY_CLASS" intValue="12270"/>	<type name="CAST_DotNet_MethodVB" id="140916"><attribute name=" KEY_CLASS" intValue="39220"/> <type name="CAST_DotNet_MethodExternal" id="137093"><attribute name="KEY_CLASS" intValue="37290"/>
<type name="NET_WEB_SERVICE_METHOD" id="540"><attribute name="KEY_CLASS" intValue="12270"/>	<type name="CAST_DotNet_MethodVB" id="140916"><attribute name=" KEY_CLASS" intValue="39220"/> <type name="CAST_DotNet_MethodExternal" id="137093"><attribute name="KEY_CLASS" intValue="37290"/>
<type name="NET_CONTROL_EVENT" id="556"><attribute name=" KEY_CLASS" intValue="12270"/>	<type name="CAST_DotNet_MethodVB" id="140916"><attribute name=" KEY_CLASS" intValue="39220"/> <type name="CAST_DotNet_MethodExternal" id="137093"><attribute name="KEY_CLASS" intValue="37290"/>
<type name="CAST_LegacyDotNet_Control" id="555"><attribute name=" KEY_CLASS" intValue="12350"/>	<type name="CAST_DotNet_FieldVBServerControl" id="141740"><attribute name="KEY_CLASS" intValue="39170"/> <type name="CAST_DotNet_FieldVBCustomControl" id="141846" ><attribute name="KEY_CLASS" intValue="39170"/> <type name="CAST_DotNet_FieldExternal" id="139064"><attribute name=" KEY_CLASS" intValue="37270"/>
<type name="NET_FIELD" id="411"><attribute name="KEY_CLASS" intValue="12350"/>	<type name="CAST_DotNet_FieldVB" id="140914"><attribute name=" KEY_CLASS" intValue="39170"/> <type name="CAST_DotNet_FieldExternal" id="139064"><attribute name=" KEY_CLASS" intValue="37270"/>
<type name="NET_FORM" id="395"><attribute name="KEY_CLASS" intValue="12110"/>	<type name="CAST_DotNet_ClassVBForm" id="141676"><attribute name=" KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalForm" id="139134"><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_INHERITED_FORM" id="398"><attribute name=" KEY_CLASS" intValue="12120"/>	<type name="CAST_DotNet_ClassVBForm" id="141676"><attribute name=" KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalForm" id="139134"><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_USER_CONTROL" id="399"><attribute name=" KEY_CLASS" intValue="12140"/>	<type name="CAST_DotNet_ClassVBUserControl" id="141675"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalUserControl" id="139133" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_INHERITED_USER_CONTROL" id="402"><attribute name="KEY_CLASS" intValue="12150"/>	<type name="CAST_DotNet_ClassVBUserControl" id="141675"><attribute name="KEY_CLASS" intValue="39140"/> <type name="CAST_DotNet_ClassExternalUserControl" id="139133" ><attribute name="KEY_CLASS" intValue="37130"/>
<type name="NET_EVENT_FIRE" id="418"><attribute name=" KEY_CLASS" intValue="12340"/>	No longer exists
<type name="NET_BALISE" id="112000"><attribute name=" KEY_CLASS" intValue="12410"/>	No longer exists
<type name="NET_BALISE_ATTRIBUTE" id="112001"><attribute name="KEY_CLASS" intValue="12420"/>	No longer exists
<type name="NET_WEB_PROJECT" id="558"><attribute name=" KEY_CLASS" intValue="8050"/>	No longer exists

Discovery and extraction in the CAST Delivery Manager Tool

Packages

When creating packages to discover and extract your .NET application you should create them as listed in the section "**What should you package?**" above.

> Unlike in previous releases of CAST AIP, the .NET analyzer no longer tries to extract the .NET framework assemblies during the analysis: this is why a specific package for external assemblies must be created in the CAST Delivery Manager Tool.

Analysis results

ASP.NET

Improvements for ASP.NET directives support

- The Src attribute of @Page, @Control and @Master directives is now taken into account
- The @Implements directive is now taken into account for master pages and controls (previously, only available for pages) is now taken into account
- The @Reference directive is now taken into account
- The @PreviousPageType and @MasterType directives are now properly taken into account, and the appropriate properties are generated
- The @Register directive now correctly creates project references / assembly references towards the corresponding control / assembly

Improvements for web.config support

- The <codeSubDirectories> tag is now correctly taken into account
- The <profile> tag is now taken into account, and creates the appropriate profile class
- The <clear> and <remove> tags are correctly handled for any list (previously, only the <add> tag was)

CAST Engineering Dashboard

- Bookmarks in source code to highlight violations are now much more accurate: for example, instead of highlighting the **entire class**, the **class identifier** is highlighted instead. Therefore if there are multiple bookmarks for a violation, all can be seen.

Improved handling of .NET dependencies

Framework assemblies

- For VB.NET projects, the CAST Delivery Manager Tool now automatically adds a dependency towards **Microsoft.VisualBasic.dll**. Note that this dll's version differs from the .NET framework's version (for example, .NET framework version 1.1 includes **Microsoft.VisualBasic.dll** version 7.1).

Assembly dependencies

- If an assembly is referenced with the **Specific Version** flag set to **False**, but still references a version, the accepted version range will be **[detected framework version + infinity]** rather than **[referenced version, referenced version]** to ensure we always match the latest version

Namespace objects

Namespace objects are no longer saved to the Analysis Service with a language ID as oppose to previous releases of CAST AIP where Namespaces were saved with a language ID.