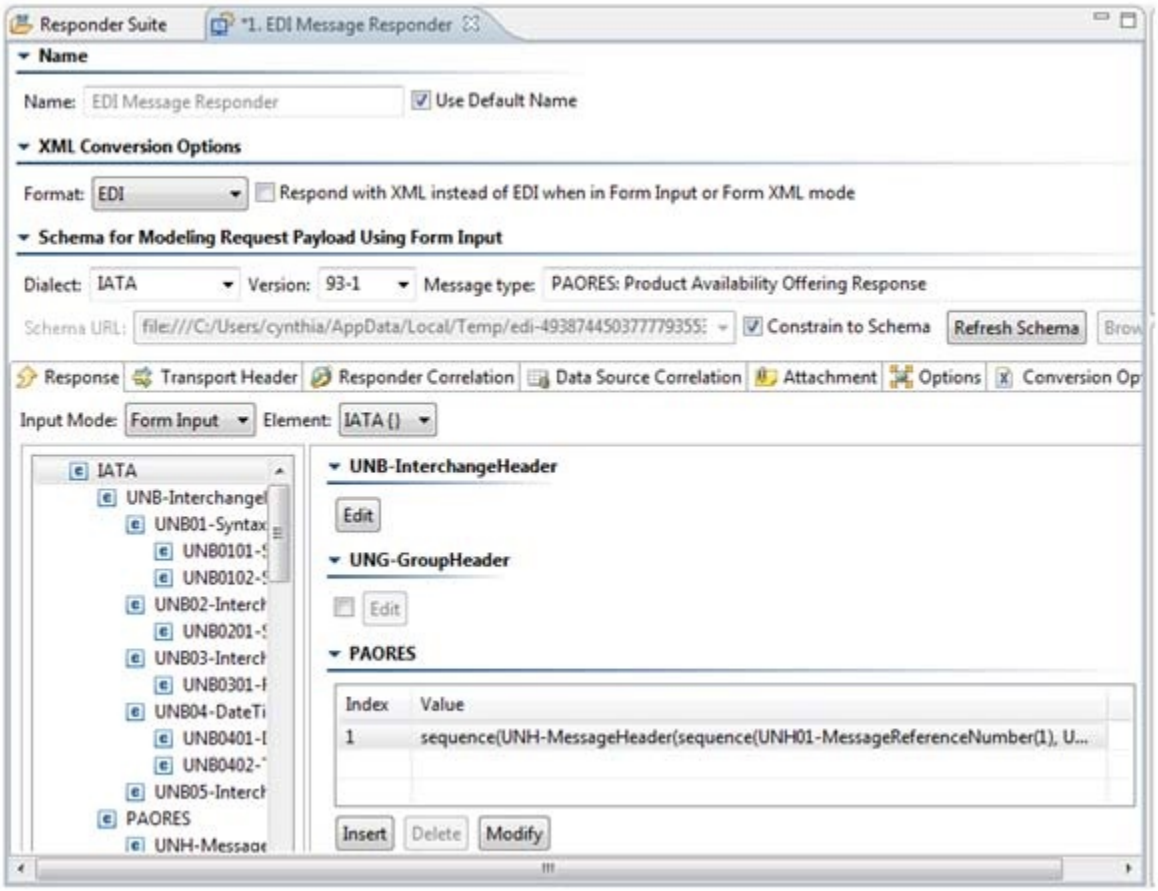
EDI Message Responder

This topic introduces the EDI Message Responder and explains configuration options that are unique to this type of Message Responder.

Sections include:

About the EDI Message Responder

EDI Message Responder is a Message Responder that is designed to simplify the use of EDI, which is often difficult to read, by allowing you to utilize XML. You can model your EDI response payload as an XML document; the responder then automatically converts the XML to EDI before sending the message. If the responder receives an EDI message, the responder can convert the message to XML so that you can define message correlations using XPaths or attach tools. Message Responders are protocol agnostic. The transport protocol or API to access a responder is defined in the deployment configuration of the PVA.



You can create a EDI Message Responder tool directly from the Add Responder wizards.

> **Message Packs License Required**
>
> The EDI format requires the Message Packs license feature to be enabled. The Message Packs license feature enables the EDI format in the EDI Responder and XML Converter.

About the EDI Format

Parasoft's EDI format support covers the following standards:

- AL3 - Also known as ACORD AL3
- CARGO - Also known as IATA Cargo-IMP
- EANCOM
- EDIFACT
- EDIFACT Basic - Described below
- EDIGAS - Also known as Edig@s
- HIPAA
- HL7
- IATA - Also known as IATA PADIS
- NCPDP - Also known as NCPDP SCRIPT
- TELCO - Also known as NCPDP Telecommunication
- TRADACOMS
- X12

For details on what versions and message types within each standard are supported, see EDI Support Details.

Basic EDIFACT Support

Basic EDIFACT support does not require a Message Packs license. It supports a subset of the dialects and versions that the Message Packs license covers. For example, EDIFACT is the only available dialect, and only versions 10B, 96B, S3, S4, and S41 are supported.

Note that Basic EDIFACT and EDI conversions generate different XML representations of the same EDI data. As a result, you cannot convert from EDI to XML using "Basic EDIFACT" and then convert back to EDI using the "EDI" conversion. Likewise, you cannot convert from EDI to XML using "EDI" and then convert back to EDI using "Basic EDIFACT."

Workflows

There are two expected workflows that can be used to configure the EDI Message Responder...

Existing Literal EDI Message

(Recommended) If you already have a literal EDI message:

1. Create a new EDI Message Responder.
2. Switch to Literal mode and paste in your EDI message.
3. Switch to Form Input or Form XML mode, which will then get populated based on the message you entered. You can now configure and parameterize your message as you wish.

Creating an EDI Message From Scratch

If you want to create an EDI message from scratch:

1. Create a new EDI Message Responder.
2. Select a message type using the **Dialect**, **Version**, and **Message** type fields, which will populate the Form Input view.
3. Manually specify the message using Form Input.

Form Input will constrain your message to the schema for the selected message type, and will not allow you to add fields to the message that are not part of the message. It will also warn you when you are attempting to use value types that are not supported in a given field (like trying to put a string into an integer field). Form XML does not perform either of these checks.

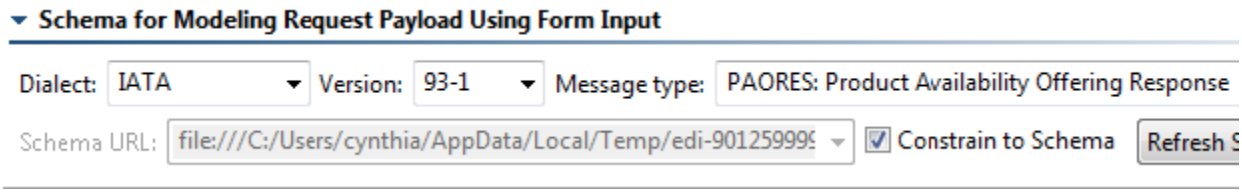
Note that you will likely need to manually add fields that your message uses because Virtualize generates a minimal valid EDI message based on the message type you selected. The message will include default values, which you might want to modify. Using the right-click **Populate** option in Form Input is generally discouraged. It will add all optional fields to the message, but it will not necessarily add valid default values. After using **Populate**, you would need to manually review and enter the value of each field. For most EDI messages, this can be quite tedious.

Specifying the Response

For recommended workflows for specifying the response, see Workflows.

Note that when you switch between Form Input or Form XML and Literal, the content in Form Input or Form XML is automatically converted to the native format, which is shown in Literal view. Content in Literal view is automatically converted to an XML format that is then shown in Form Input or Form XML.

In EDI format mode, if you paste in an EDI message into Literal view and then switch to Form Input or Form XML, Virtualize will automatically detect the message type and automatically populate the **Dialect**, **Version**, and **Message Type** combo boxes.



You can now configure the EDI Message Responder as you could configure any other Message Responder. For details on configuring standard Message Responder behavior (e.g., correlations, performance profiles, etc.), see Message Responder Overview.

Sending custom EDI messages

If you want to send a custom EDI message, Form XML view will let you add custom fields that Form Input will not allow you to enter.

What if the view is not populated after I switch views?

If you paste a message into Literal view, and switch to Form Input but Form Input does not get populated, then switch from Literal view to Form XML instead. Form Input will not be populated in the cases where Virtualize is not able to find a matching schema for the EDI message that was pasted in. However, with Form XML, Virtualize can often still support those messages.

Specifying Conversion Options

In EDI format, when an option is blank, the conversion will use the default option for that option. Otherwise, if a value is selected or typed in, the conversion will attempt to use that value. Note that if an invalid value is entered manually, you may get an error when trying to switch between Form Input /Form XML and Literal views, or when running the test.

For EDI, available conversion options are:

- **Treat all segments and elements as optional** - If set to yes, all mandatory segments and mandatory data elements are treated as optional. If set to no (the default), a missing mandatory segment or mandatory data element triggers an error. Enabling this option can be useful if your provider declines to provide segments and elements that are considered mandatory according to the EDI specification.
- **Enable validation** - When set to yes (the default), the version, release, messages and segments of the EDI file (input or output) are compared to the relevant EDI repository. If the EDI file contains a value that is not in the EDI repository, an error is thrown.
  If validation is disabled (the value is set to "no"), processing continues even if the EDI file contains a version, release, message, or segment that is not in the repository. When processing an unknown version, release, message, or segment, some checks cannot be performed because the structure of the required data is unknown.
  For example, if a file is of a known version but contains an unknown segment, data type checking for that segment is not performed, but checks on the remainder of the file are performed as usual. Similarly, if a message does not exist in the EDI repository, the file is still processed, but segment order checking is not performed.
- **Component value separator** - When an element is a composite element, the character that is used to separate the component elements from each other within the composite element. This option affects EDI to XML conversion and vice versa.
- **Line continuation character** - Character appended to the segment terminator when each segment in an EDI message is split onto a new line. This character indicates to the host system that the end of the interchange has not been reached. Appended to all segments in an interchange but the last one. This option affects EDI to XML conversion and vice versa. This option is supported for all EDI dialects except CARGO.
- **Decimal character** - Symbol used for the decimal character in the converted file. Usually a period or comma. This option affects EDI to XML conversion and vice versa.
- **Element separator** - The character used to separate elements in a segment. This option affects EDI to XML conversion and vice versa.
- **Segment name/content separator (TRADACOMS)** - In TRADACOMS data streams, the default character used to separate the segment name and segment contents is the equal sign (=). You can use this option to override the default character. This option affects EDI to XML conversion and vice versa. This option only applies to the TRADACOMS dialect.
- **Invalid character replacement** - Used with the REPLACE value for the Character repertoire override (EDIFACT) option to specify the character that should be used to replace invalid characters. The default (if this option is not specified) is an underscore ("_")). Valid values are:
  \u#### – Use to specify a Unicode value (substitute the #### for the appropriate value).
  \d#### – Use to specify a decimal value, (substitute the #### for the appropriate value).
  This option affects EDI-to-XML conversions and vice versa.
- **Release (escape) symbol** - The release, or escape, character. It turns off special processing of the next character. Suppose your text message uses the same character that is also used to separate elements. The specified character is used to instruct the EDI processor to treat that character as a normal character and not the end of the text. This option affects EDI to XML conversion and vice versa.
- **Repeat symbol** - The repeat symbol for EDI dialects that use it. This option affects EDI to XML conversion and vice versa.
- **Segment separator** - The character you want to use for segment separators. This option affects EDI to XML conversion and vice versa.
- **Stop processing of EDI content character** - Specifies a character that stops all processing of EDI content when that character is encountered. For example, if an EDI stream is terminated when it encounters a ^Z (0x1A), specifying terminate=26 (26 is the decimal equivalent of 0x1A) causes EDI parsing to end when the ^Z is encountered and any remaining file content is ignored.
- **Subcomponent (tertiary separator)** - The character you want to use for subcomponent separators. This option affects EDI to XML conversion and vice versa.
- **Character repertoire override (EDIFACT)** - Allows the conversion to override and alter character encodings for EDIFACT-based documents (like EANCOM and IATA).

You can use one or more of the following values, concatenated with a "+" symbol (FINNISH+REPLACE, for example):
DEFAULT – The encoding specified in the file is used. This value cannot be used with any others.
EANCOM – Support for these extra EANCOM characters to UNOA and UNOB are added: #, @, [,], {, }, \, |, ‘, and ^.
SYMBOL – Forces all characters, including special characters such as element and segment separators that might otherwise be permitted, to be validated against the encoding.
REPLACE – Replaces any invalid characters with the character specified by the invalid property. An underscore ("_") is used if the invalid property is not specified. If REPLACE is not specified, the conversion reports an error.
FINNISH – Changes the meaning of certain characters in the Finnish character set for UNOA and UNOB (and adds UNOY and UNOZ as synonyms for UNOA and UNOB respectively).

- **Extension map file** - Allows you to enter the path to an extension map file that describes how the custom/proprietary EDI message type you are using differs from the EDI standard on which it is based. Extension map files can use the EDI Standard Exchange Format (SEF), an open standard that allows you to describe proprietary EDI message types. The ${ENV_VAR} format can be used to specify the file path.

Note that no conversion options are applicable for Basic EDIFACT.