




Wireshark (Ethereal) PDML to Sequence Diagram Conversion Tutorial

Exporting to PDML from Wireshark (Ethereal)


VisualEther requires the Wireshark (Ethereal) logs to be exported to the PDML XML format. This requires the following steps:

1. Invoke Wireshark (Ethereal).
2. Use the "File > Open" command to open the example Wireshark (Ethereal) log "Web Browsing Ethereal Capture.bin" located in the "Examples\Web Browsing" directory in the VisualEther installation directory (typically "C:\Program Files\EventHelix.com\VisualEther Protocol Analyzer\Examples\Web Browsing").
3. Now invoke the "File > Export > as XML PDML (packet details file)" Wireshark (Ethereal) menu.
4. Name this file as "Web Browsing Ethereal Capture.pdml.xml". (Note that the file should be given a **.pdml.xml** extension.)

 **Ethereal PDML export problem**

In some releases of Ethereal the Export to PDML functionality is broken. An error dialog box, similar to the one on the right, is shown.

The problem has been addressed in the latest release of Wireshark (Ethereal has been renamed Wireshark). The new release can be downloaded from: <http://www.wireshark.org/download.html>



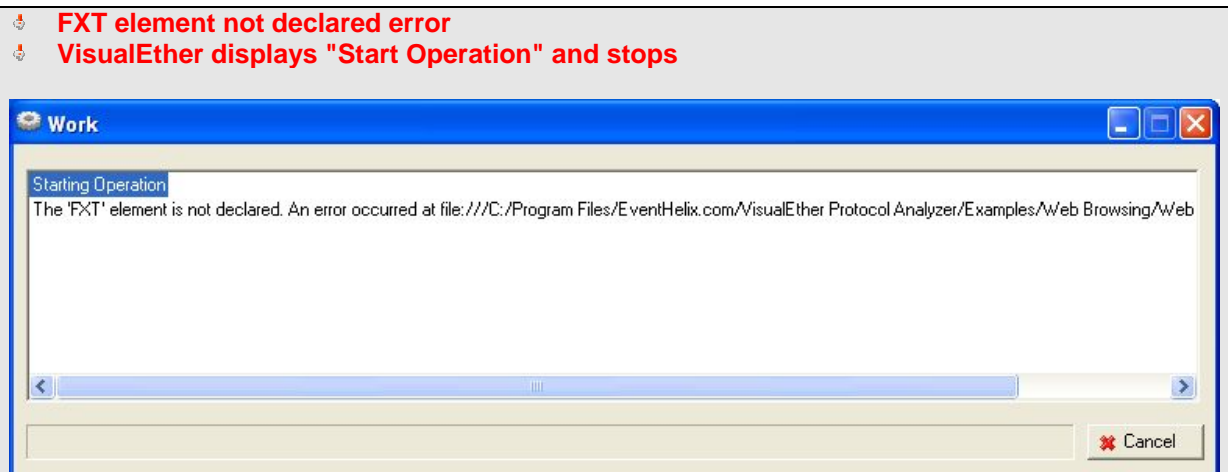
Generating Sequence Diagrams

The next step is to generate Sequence Diagrams output from the PDML file:

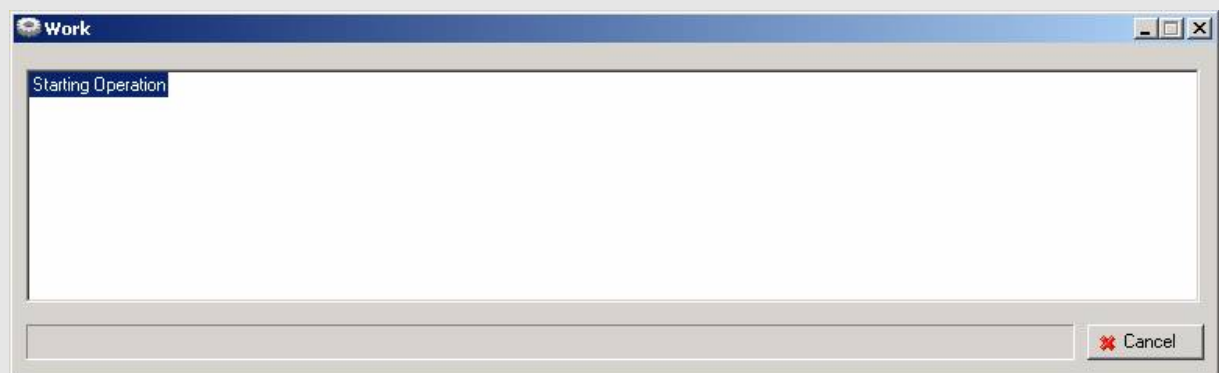
1. Double click on the VisualEther icon.
2. Select the Wireshark (Ethereal) PDML file that was generated in the previous section.
 - a. Click the "Browse..." button for the Wireshark (Ethereal) PDML file.
 - b. VisualEther will display a file selection dialog box.
 - c. Change directory to "**Examples\Web Browsing**" in the VisualEther installation directory.
("c:\Program Files\EventHelix.com\VisualEther Protocol Analyzer\Examples\Web Browsing" is the path for a default installation.)
 - d. Select the "**Web Browsing Ethereal Capture.pdml.xml**" file from the list (note that the PDML files have a **.pdml.xml** file extension).

3. Now select the Field Extraction Template (FXT) file to specify the fields that need to be included in the diagram.
 - a. Click the "Browse..." button for the FXT file.
 - b. VisualEther will display a file selection dialog box.
 - c. Change directory to "**Examples\Web Browsing**" in the VisualEther installation directory.
 - d. Select the "**Web Browsing.fxt.xml**" file from the list (note that the FXT files have a **.fxt.xml** file extension).
4. Click on the "Generate Documents" button.
5. VisualEther will perform the following steps:
 - a. Extract the fields from the PDML file based on the FXT file template.
 - b. Generate the FDL and SCN files needed by the EventStudio System Designer.
 - c. Invoke the EventStudio System Designer to generate the sequence diagram.
 - d. Invoke Adobe Acrobat Reader to open the sequence diagram.

The above example covered sequence diagram generation for web browsing. You can generate sequence diagrams for pretty much any protocol. All you need is an .fxt.xml file with the appropriate extraction templates. For templates for a large number of protocols, refer to section titled **All.fxt.xml**.



The "FXT element not declared" is displayed when a Firewall blocks access to the FXT file's schema definition or internet access is not available from the machine running VisualEther.



VisualEther displays "Start Operation" but does not proceed beyond that point. This condition is caused by the firewall blocking access to the FXT file's schema definition.

VisualEther attempts to load the schema definition from http://www.eventhelix.com/Schemas/FXT_0_3.xsd. If Internet access is not available or a firewall is blocking the access, you can get one of the two conditions

described above.

If Internet access is available on the machine, the likely cause of the problem is the firewall. We recommend adding VisualEther to the Firewall's safe application list.

If Internet access is not available or updating the Firewall is not an option, please follow the procedure listed below:

1. Open the following FXT schema link in the Internet Explorer Browser:
http://www.eventhelix.com/Schemas/FXT_0_3.xsd
2. Now save this file to your local drive
 - a. Right click the Internet Explorer window and select the "View Source".
 - b. This will open the schema file in notepad.
 - c. Invoke the File->Save As menu.
 - d. Change the "Save As Type" to "All Files".
 - e. Change directory to C:\
 - f. Save the file with the name FXT_0_3.xsd.
3. Edit the FXT files to replace the following line:
<FXT xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.eventhelix.com/Schemas/FXT_0_3.xsd">
with
<FXT xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\FXT_0_3.xsd">

Exploring the Field Extraction Template (FXT)

VisualEther uses FXT files to define the contents of the generated sequence diagrams. The FXT file specifies the fields in the Wireshark (Ethereal) capture that should be included in the generated sequence diagram.

A typical FXT template file is shown in Figure 1.

```
<?xml version="1.0" encoding="utf-8"?>

<!--
Description:
  Template for documenting Web Browsing interactions with VisualEther.

Protocols:
  DNS: Domain Name System
  HTTP: Hypertext Transfer Protocol

Copyright © EventHelix.com Inc. 2006. All Rights Reserved.
-->

<FXT xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.eventhelix.com/Schemas/FXT_0_3.xsd">

  <!-- Message Template for Domain Name System (DNS) Extraction -->
  <udp-message>
    <opcode>dns</opcode>
    <param>dns.flags</param>
    <param>dns.qry.name</param>
    <param>dns.qry.type</param>
    <param>dns.qry.class</param>
    <param>dns.resp.name</param>
    <param>dns.resp.type</param>
    <param>dns.resp.class</param>
    <param>dns.resp.ttl</param>
    <param>dns.Addr</param>
  </udp-message>

  <!-- Message Template for Hypertext Transfer Protocol (HTTP) Request Extraction -->
  <tcp-message>
    <opcode>http.request.method</opcode>
    <param>http.request.uri</param>
    <param>http.request.version</param>
    <param>http.response.code</param>
    <param>http.If-Modified-Since</param>
    <param>tcp.len</param>
  </tcp-message>

  <!-- Message Template for Hypertext Transfer Protocol (HTTP) Response Extraction -->
  <tcp-message>
    <opcode>http.response.code</opcode>
    <param>http.request.uri</param>
    <param>http.request.version</param>
    <param>tcp.len</param>
  </tcp-message>

  <!-- Default Message Template for Hypertext Transfer Protocol (HTTP) -->
  <tcp-message>
    <opcode>http</opcode>
    <param>http.request.uri</param>
    <param>http.request.version</param>
    <param>http.response.code</param>
    <param>tcp.len</param>
  </tcp-message>

</FXT>
```

Figure 1 Field Extraction Template (FXT) Example

Basic Structure

The basic structure of the Field Extraction Template (FXT) file is shown below.

- The file begins with a standard `xml` starting tag.
- Following XML syntax, all elements are enclosed in a "begin tag" and an "end tag". (`<begin-tag>` `</end-tag>`)
- `<fxt>` is the root element; the element also specifies the location of the schema file (http://www.eventhelix.com/Schemas/FXT_0_3.xsd). The schema file defines the validation rules for the FXT format.
- The root element contains pattern matching templates enclosed by the `<message>`, `<ip-message>`, `<udp-message>`, `<tcp-message>` or `<sctp-message>` element. Each template definition specifies the protocol fields that need to be extracted. Typically you would define a pattern definition for every protocol needed in the final diagram.

```
<?xml version="1.0" encoding="utf-8"?>

<FXT xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.eventhelix.com/Schemas/FXT_0_3.xsd">

  <!-- Message Template for Domain Name System (DNS) Extraction -->
  <udp-message>

  </udp-message>

  <!-- Message Template for Hypertext Transfer Protocol (HTTP) Request Extraction -->
  <tcp-message>

  </tcp-message>

  <!-- Message Template for Hypertext Transfer Protocol (HTTP) Response Extraction -->
  <tcp-message>

  </tcp-message>

  <!-- Default Message Template for Hypertext Transfer Protocol (HTTP) -->
  <tcp-message>

  </tcp-message>

</FXT>
```

Figure 2 Basic Structure of a Field Extraction Template (FXT) File

Message Template

A typical message template is shown in the following figure. The template contains specification for extraction of various components of a captured packet. The following field extraction templates are commonly used:

<code><opcode></code>	The opcode extraction template is used to extract the packet name in the sequence diagram. (Details in section 0)
<code><param></code>	Message parameters are extracted by the parameter extraction template. (Details in section 0)

These extraction templates will be discussed in detail in the following sections.

```
<!-- Message Template for Domain Name System (DNS) Extraction -->
<udp-message>
  <opcode>dns</opcode>
  <param>dns.flags</param>
  <param>dns.qry.name</param>
  <param>dns.qry.type</param>
  <param>dns.qry.class</param>
  <param>dns.resp.name</param>
  <param>dns.resp.type</param>
  <param>dns.resp.class</param>
  <param>dns.resp.ttl</param>
  <param>dns.Addr</param>
</udp-message>
```

Figure 3 Message Template

The pattern matching example will focus on a DNS Query packet that is represented in PDML in the following figure. The figure shows all the interesting fields in color, rest of the fields have been grayed out.

```

<packet>
  <proto name="geninfo" pos="0" showname="General information" size="143">
    <field name="timestamp" pos="0" show="Apr 20, 2005 06:13:42.118718000"
      showname="Captured Time" value="1113992022.118718000" size="143"/>
  </proto>
  <proto name="frame" showname="Frame 4 (143 bytes on wire, 143 bytes captured)"
  size="143" pos="0">
  </proto>
  <proto name="eth" showname="Ethernet II, Src: 00:80:ae:cf:6f:dc, Dst:
  08:00:46:43:79:cf"
  size="14" pos="0">
  </proto>
  <proto name="ip" showname="Internet Protocol, Src Addr: 66.82.4.8 (66.82.4.8), Dst
  Addr: 192.168.0.2 (192.168.0.2)" size="20" pos="14">
    <field name="ip.src" showname="Source: 66.82.4.8 (66.82.4.8)" size="4" pos="26"
    show="66.82.4.8" value="42520408"/>
    <field name="ip.dst" showname="Destination: 192.168.0.2 (192.168.0.2)" size="4"
    pos="30" show="192.168.0.2" value="c0a80002"/>
  </proto>
  <proto name="udp" showname="User Datagram Protocol, Src Port: domain (53), Dst Port:
  1035 (1035)" size="8" pos="34">
    <field name="udp.srcport" showname="Source port: domain (53)" size="2" pos="34"
    show="53" value="0035"/>
    <field name="udp.dstport" showname="Destination port: 1035 (1035)" size="2" pos="36"
    show="1035" value="040b"/>
  </proto>
  <proto name="dns" showname="Domain Name System (response)" size="101" pos="42">
    <field name="dns.id" showname="Transaction ID: 0x82a3" size="2" pos="42"
    show="0x82a3" value="82a3"/>
    <field name="dns.flags" showname="Flags: 0x8180 (Standard query response, No error)"
    size="2" pos="44" show="0x8180" value="8180"/>
  </field>
    <field show="Queries" size="24" pos="54"
    value="037777770a6576656e7468656c697803636f6d0000010001">
      <field show="www.eventhelix.com: type A, class IN" size="24" pos="54"
      value="037777770a6576656e7468656c697803636f6d0000010001">
        <field name="dns.qry.name" showname="Name: www.eventhelix.com" size="20" pos="54"
        show="www.eventhelix.com" value="037777770a6576656e7468656c697803636f6d00"/>
        <field name="dns.qry.type" showname="Type: A (Host address)" size="2" pos="74"
        show="0x0001" value="0001"/>
        <field name="dns.qry.class" showname="Class: IN (0x0001)" size="2" pos="76"
        show="0x0001" value="0001"/>
      </field>
    </field>
    <field show="Answers" size="16" pos="78" value="c00c000100010001518000043f63d153">
      <field show="www.eventhelix.com: type A, class IN, addr 63.99.209.83" size="16"
      pos="78" value="c00c000100010001518000043f63d153">
        <field name="dns.resp.name" showname="Name: www.eventhelix.com" size="2" pos="78"
        show="www.eventhelix.com" value="c00c"/>
        <field name="dns.resp.type" showname="Type: A (Host address)" size="2" pos="80"
        show="0x0001" value="0001"/>
        <field name="dns.resp.class" showname="Class: IN (0x0001)" size="2" pos="82"
        show="0x0001" value="0001"/>
        <field name="dns.resp.ttl" showname="Time to live: 1 day" size="4" pos="84"
        show="86400" value="00015180"/>
        <field name="dns.resp.len" showname="Data length: 4" size="2" pos="88" show="4"
        value="0004"/>
        <field show="Addr: 63.99.209.83" size="4" pos="90" value="3f63d153"/>
      </field>
    </field>
  </proto>
</packet>

```

Figure 4 Packet Representation in PDML

Opcode Extraction Template

The opcode field defines the packet name in the generated sequence diagrams. The opcode extraction template definition for the DNS protocol is shown below.

```
<udp-message>
  <!--Pattern Matching Template -->
  <opcode>dns</opcode>

  <!-- Other Extraction Templates -->
</udp-message>
```

Figure 5 Opcode Extraction Template Shown in a UDP Message Template

The figure below shows an example of an opcode extracted by the <opcode> extraction template.

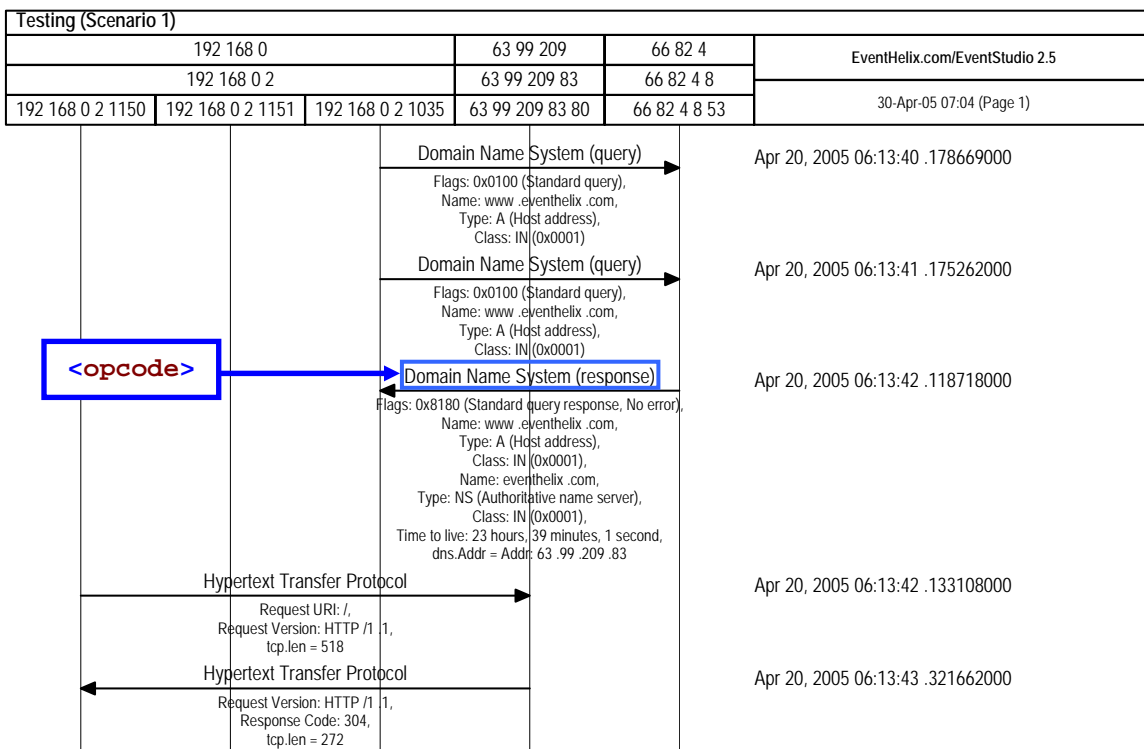


Figure 6 Extracted Opcode Shown in a Sequence Diagram

The selected field has been picked from the highlighted PDML entry shown in Figure 7. The field name specified in the extraction template (<opcode>dns</opcode>) is matched with the name attribute specified in the PDML file (<proto name="dns"...).

Note that VisualEther picks the **showname** attribute's value for representing the opcode.¹

¹ By default VisualEther picks the entry defined by the **showname** attribute. The **show** attribute can be selected by using the **display="brief"** attribute in the extraction template.

```
<packet>

  <proto name="geninfo" pos="0" showname="General information" size="143">
  </proto>

  <proto name="frame" showname="Frame 4 (143 bytes on wire, 143 bytes captured)"
  </proto>

  <proto name="ip" showname="Internet Protocol, Src Addr: 66.82.4.8 (66.82.4.8), Dst
  Addr: 192.168.0.2 (192.168.0.2)" size="20" pos="14">
  </proto>

  <proto name="udp" showname="User Datagram Protocol, Src Port: domain (53), Dst Port:
  1035 (1035)" size="8" pos="34">
  </proto>

  <proto name="dns" showname="Domain Name System (response)" size="101"
  pos="42">
    <field name="dns.id" showname="Transaction ID: 0x82a3" size="2" pos="42"
  show="0x82a3" value="82a3"/>
  </proto>
</packet>
```

Figure 7 PDML Field Selected for Opcode Extraction

Parameter Extraction Template

The parameter extraction template is used to display addition attributes of the packet. Packet parameters are represented below the arrow in a sequence diagram. The parameter contents are extracted using the `<param>` extraction template. Extracted representation of two such parameters is shown in the following figure.

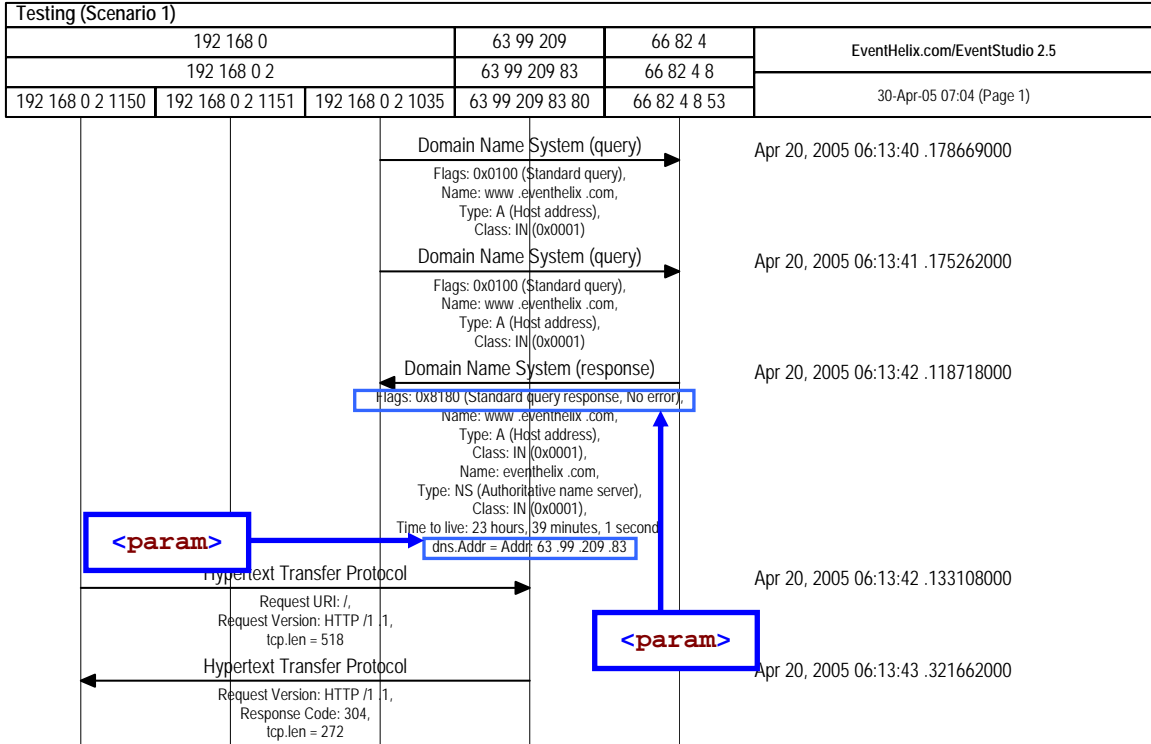


Figure 8 Some of the Extracted Parameters Shown in a Sequence Diagram

Here we will have a closer look at the extraction of the first and the last parameters of the "Domain Name System (response)" packet. The template for these parameters is shown below.

```

<udp-message>
  <!-- Opcode Extraction Template -->
  <opcode>dns</opcode>

  <!-- Parameter Extraction Template -->
  <param>dns.flags</param>

  <!-- More Parameters -->
  <param>dns.Addr</param>

  <!-- More Extraction Templates -->
</udp-message>
    
```

Figure 9 Parameter Extraction Template Shown in a Message Template

The first highlighted parameter in Figure 9 extracts the value of `dns.flags`. The `showname` attribute is extracted from a field with the name `"dns.flags"`. (See Figure 10)

The second highlighted parameter in Figure 9 really defines a special case. Here Wireshark (Ethereal) has not included a name attribute in the field element (See the second highlighted field

in Figure 10). Such fields are addressed by internally naming the field. In this case the field has been named `dns.Addr`. The name has been derived from the name of the enclosing `<proto>` attribute (`dns`) and the first part of the `show` string (`Addr` - The part before the colon).

```

<proto name="dns" showname="Domain Name System (response)" size="101" pos="42">
  <field name="dns.id" showname="Transaction ID: 0x82a3" size="2" pos="42"
  show="0x82a3" value="82a3"/>
  <field name="dns.flags" showname="Flags: 0x8180 (Standard query
  response, No error)" size="2" pos="44" show="0x8180" value="8180">
  </field>
  <field show="Queries" size="24" pos="54" >
    <field show="www.eventhelix.com: type A, class IN" size="24" pos="54">
      <field name="dns.qry.name" showname="Name: www.eventhelix.com" size="20" pos="54"
      show="www.eventhelix.com" value="037777770a6576656e7468656c697803636f6d00"/>
      <field name="dns.qry.type" showname="Type: A (Host address)" size="2" pos="74"
      show="0x0001" value="0001"/>
      <field name="dns.qry.class" showname="Class: IN (0x0001)" size="2" pos="76"
      show="0x0001" value="0001"/>
    </field>
  </field>
  <field show="Answers" size="16" pos="78" value="c00c000100010001518000043f63d153">
    <field show="www.eventhelix.com: type A, class IN, addr 63.99.209.83" size="16"
    pos="78" value="c00c000100010001518000043f63d153">
      <field name="dns.resp.name" showname="Name: www.eventhelix.com" size="2" pos="78"
      show="www.eventhelix.com" value="c00c"/>
      <field name="dns.resp.type" showname="Type: A (Host address)" size="2" pos="80"
      show="0x0001" value="0001"/>
      <field name="dns.resp.class" showname="Class: IN (0x0001)" size="2" pos="82"
      show="0x0001" value="0001"/>
      <field name="dns.resp.ttl" showname="Time to live: 1 day" size="4" pos="84"
      show="86400" value="00015180"/>
      <field name="dns.resp.len" showname="Data length: 4" size="2" pos="88" show="4"
      value="0004"/>
      <field show="Addr: 63.99.209.83" size="4" pos="90"
      value="3f63d153"/>
    </field>
  </field>
</proto>

```

Figure 10 PDML Fields Selected for Parameter Extraction

All.fxt.xml

VisualEther includes a predefined extraction file that supports a large number of protocols. The file can be found in the VisualEther installation directory.

("c:\Program Files\EventHelix.com\VisualEther Protocol Analyzer\All.fxt.xml" is the path for a default installation.)

The following protocols are covered in All.fxt.xml

SIP	Session Initiation Protocol
H.225	Narrowband visual telephone services
H.245	Negotiate channel use and capabilities
Q.931	Manage call setup and termination
RADIUS	Remote Authentication Dial In User Service
RTP	Real-time Protocol
RTCP	RTP Control Protocol
SNMP	Simple Network Management Protocol
NFS V3	Network File System (Version 3)
RPC	Remote Procedure Call
TCAP	Transaction Capabilities Application Part
SCTP	Stream Control Transport Protocol
POP3	Post Office Protocol (Version 3)
IGMP	Internet Group Management Protocol
ARP	Address Resolution Protocol
DNS	Domain Name System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
NBNS	Net Bios Name Service
BOOTP	Bootstrap Protocol
OSPF	Open Shortest Path First Routing Protocol
BGP	Border Gateway Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
IP	Internetworking Protocol