Returning Data Through SOAP

By David Adams
TN 04-12

Introduction

4th Dimension 2003 and later support publishing methods as Web Services. A single method may often be called both locally within the database and remotely as a Web Service without any special coding or modifications. When called as Web Services, however, methods have greater flexibility in the data they return than when called locally by other 4th Dimension methods. Instead of being limited to the $0 result parameter, methods can return $0, and/or any number and combination of process variables and arrays. Returning multiple values with SOAP is comparable to returning a whole record instead of a single field. This technical note discusses how to return data through Web Services and what values may be returned. It also provides a sample database that illustrates 4th Dimensions features and behavior.

Review: Web Services, SOAP and XML

Internally, Web Service method requests and responses are SOAP (Simple Object Access Protocol) messages sent through the native 4th Dimension Web server. SOAP messages, and the data they contain, follow the syntactic rules and restrictions of XML. 4th Dimension automatically handles all of these tasks:

- Receiving incoming SOAP messages.
- Converting XML data types into native 4th Dimension data types.
- Running security checks.
- Dispatching the SOAP request to the correct 4th Dimension project method.
- Binding 4th Dimension values to XML elements.
- Converting 4th Dimension values to XML values.
- Formatting and sending the SOAP response.

These processes are invisible to us as 4th Dimension developers; they just work. When we develop and debug Web Services, however, we should become comfortable with looking at the underlying SOAP messages.

This technical note includes XML listings for SOAP messages to illustrate behavior and demonstrate features. None of these features or behavior, however, require looking at, working with, or manipulating XML directly. For more information on how to capture and examine SOAP messages, review The 4D Web Companion, or 4D, Inc. Technical Note 03-21, "Tracing and Troubleshooting TCP/IP".

Review: Activating Web Services

In order to publish methods as Web Services, the following conditions must be met:

1- The native 4th Dimension Web server mush be running and accessible.
2- The following Database Preference must be enabled:
   Preferences -> Web -> Web Services -> Allow Web Services Requests

3- The following Method Property must be enabled for each published method: Method Properties -> Offered as a Web Service

Defining Returned Data

4th Dimension provides two ways to define the data returned in a SOAP response:

- The $0 parameter of any method offered as a Web Service can be included as an output automatically. No special coding is required.
- Process variables and arrays may be bound to the output using the SOAP DECLARATION command.

The $0 Result Parameter

The sample code takes the text passed in $1 and returns it in $0:

```
C_TEXT($0;$1)
$0:=$1
```

While the code above is trivial, it illustrates how simple it is to publish a Web Service. Below is the XML of the SOAP response corresponding to a call to this method with an input of Hello world! (highlighting and whitespace adjusted for clarity):

<?xml version="1.0" encoding="UTF-8" ?>

```
<SOAP-ENV:Envelope
       SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
       xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
       xmlns:xsd="http://www.w3.org/2001/XMLSchema"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<SOAP-ENV:Body>

       <ns1:returnInputTextResponse
       xmlns:ns1="http://www.4d.com/namespace/default"
       SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

              <FourD_arg0 xsi:type="xsd:string">Hello world!</FourD_arg0>

       </ns1:returnInputTextResponse>

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

As described in the 4th Dimension Language Reference, the value of $0 is returned in an XML element named FourD_arg0. The SOAP DECLARATION command can rename the XML element name in the output, as in the example below:

```
SOAP DECLARATION($0;Is Text ;SOAP Output ;"outEchoInputString")
```

The XML fragment below illustrates how the call to SOAP DECLARATION modifies the SOAP response (highlighted for clarity):

```
       <outEchoInputString xsi:type="xsd:string">Hello world!</outEchoInputString>
```

Defining clear and descriptive XML element names is only one of the SOAP DECLARATION command's applications. As we'll discuss next, it's through calls to SOAP DECLARATION that a single method may return multiple outputs, including variables and arrays.

SOAP DECLARATION and Outputs Other than $0

The SOAP DECLARATION command can be called any number of times within a method to define multiple input and output parameters. The code listed below defines three process variables to return in a single SOAP response:

```
SOAP DECLARATION(returnLongint_l;Is Longint ;SOAP Output ;"outSecondsSinceMidnight")
SOAP DECLARATION(returnReal_r;Is Real ;SOAP Output ;"outPi")
SOAP DECLARATION(returnString_s;Is String Var ;SOAP Output ;"outServerVersion")

returnLongint_l:=Current time(*)+0  ` Convert time to seconds since midnight.
returnReal_r:=Pi
returnString_s:="Version #"+Application version
```

The calls to SOAP DECLARATION listed above bind 4th Dimension values to XML output elements, as summarized in the table below:

4th Dimension Object	XML Element Name
returnLongint_l	outSecondsSinceMidnight
returnReal_r	outPi
returnString_s	outServerVersion

The return element names and values are highlighted in the XML of the SOAP response below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope
       SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
       xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
       xmlns:xsd="http://www.w3.org/2001/XMLSchema"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<SOAP-ENV:Body>
<ns1:returnThreeResultsResponse
       xmlns:ns1="http://www.4d.com/namespace/default"
       SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

       <outSecondsSinceMidnight xsi:type="xsd:float">40174</outSecondsSinceMidnight>
       <outPi xsi:type="xsd:float">3.14159265358979311 6</outPi>
       <outServerVersion xsi:type="xsd:string">Version #0703</outServerVersion>

</ns1:returnThreeResultsResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Any number of 4th Dimension values may be included in a SOAP response, within the limits of available resources and the restrictions of supported data types. The ability to return several values—including entire arrays—from a single method offers greater flexibility than 4th Dimension's single $0 result parameter.

Supported Data Types and Sources

SOAP output values may come from the $0 result parameter, process variables, and process arrays, as summarized in the table below:

Source	Supported?
$0	Yes
Expressions	No
Fields	No
Interprocess arrays	No
Interprocess variables	No
Literals	No
Local arrays	No
Local variables	No
Parameters other than $0	No
Process arrays	Yes
Process variables	Yes

4th Dimension translates between SOAP and 4D data types transparently and automatically using the conversions listed in the table below.

4D Data Type	Variables Supported	Arrays Supported	SOAP Translation	Example
BLOB	Yes	N/A	base64Binary	U09BUF9CTE9C
Boolean	Yes	Yes	boolean	0
Date	Yes	Yes	date	2003-08-23
Integer	Yes	Yes	int	1
Longint	Yes	Yes	int	1
Picture	No (see note 1)	N/A	N/A	N/A
Pointer	No	No	N/A	N/A
Real	Yes	Yes	float	1.1
StringVar	Yes	Yes	string	Hello world!
Text	Yes	Yes	string	Hello world!
Time	Yes	No (see note 2)	time	16:54:55-10:00

Note 1: Picture data can be sent if converted to a BLOB, as demonstrated in the sample database provided with this note and as explained and demonstrated in detail in Technical Note, "Returning Pictures through SOAP".

Note 2: 4th Dimension doesn't have a distinct array type for time values. Instead, it uses a longint array to hold time values. This is a long-standing feature of the 4th Dimension language and is unchanged by Web Services.

Example Database

Requests and Response

The example database included with this note consists of a series of simple methods illustrating how to send each of variable and array types supported over SOAP, as well as how to send multiple values. The method names listed on the left are the Web Service methods that return the data, and the method names listed on the right are the corresponding SOAP client methods to retrieve the data. Demonstration forms provide a simple-to-use system for running the client methods.

Web Service Method	Client Proxy Method
returnBlobVariable	request_BlobVariable
returnBooleanArray	request_BooleanArray
returnBooleanVariable	request_BooleanVariable
returnDateArray	request_DateArray
returnDateVariable	request_DateVariable
returnInputText	request_InputText
returnIntegerArray	request_IntegerArray
returnIntegerVariable	request_IntegerVariable
returnLongintArray	request_LongintArray
returnLongintVariable	request_LongintVariable
returnPictureVariable	request_PictureVariable
returnRealArray	request_RealArray
returnRealVariable	request_RealVariable
returnStringArray	request_StringArray
returnStringVariable	request_StringVariable
returnTextArray	request_TextArray
returnTextVariable	request_TextVariable
returnThreeResults	request_ThreeResults
returnTimeArray	request_TimeArray
returnTimeVariable	request_TimeVariable

Configuring the Proxy Methods

4th Dimension SOAP proxy methods specify the URL of the target Web Service directly. In the case of this example, the methods look for the local loopback address (the current machine) at http://127.0.0.1/. Instead of hard-coding this address in each method, the proxy methods call a function named requestGetAccessURL to retrieve the correct address. To configure the client methods to use a different server address, simply adjust the single line of code found in the requestGetAccessURL method. In a production system, the target SOAP server URL could be stored in other locations, such as records, lists or resources, for simpler maintenance.

The Client May Be Run in a Distinct Copy of 4th Dimension

The client methods and Web Service methods are included in a single database for convenience. To create a more typical Web Service deployment environment, duplicate the database and run the client methods in one copy of 4th Dimension and the Web Service methods in another copy of 4th Dimension. Be sure to turn off the Web server in the copy of the database working as a client.

Port Numbers

The example database is configured by default to listen to Web requests—including Web Service calls—on port 8080. You may set this port number to any unused port allowed by your operating system.

Note: Compiler_Web

The Compiler_Web project method can be a source of confusion to developers when first working with 4th Dimension's Web and Web Service features. The Compiler_Web method automatically binds input values from Web forms and SOAP requests to process variables and arrays. Values used as outputs do not need to be declared in the Compiler_Web method.

Notes and Reminders about SOAP DECLARATION

Below are key notes and reminders about using SOAP DECLARATION:

- The SOAP DECLARATION command must be called directly within the method called as a Web Service, not in a subroutine.
- XML element names don't allow special characters, notably spaces. Make sure to provide legal XML names to the SOAP DECLARATION command.
- Element names matter. SOAP clients work with element names and may depend on the names to understand the data. Define clear, complete, and descriptive names with the SOAP DECLARATION command.
- The SOAP DECLARATION command has no relationship with 4th Dimension's compiler declarations. Variables, parameters, and arrays must be declared and initialized normally.
- The SOAP DECLARATION command is ignored unless the current method was called as a Web Service. This behavior makes it safe to call the same method locally from another 4th Dimension method or as a Web Service.