Base Language - Feature #1644

implement SOAP support

10/21/2012 12:00 PM - Greg Shah

Status:	Closed	**Start date:**	02/13/2013
Priority:	Normal	**Due date:**	07/12/2013
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Runtime Support for Server Features		
billable:	No	**vendor_id:**	GCD

Description

Subtasks:

Feature # 1994: conversion support for SOAP	**Closed**
Feature # 1995: runtime support for SOAP	**Closed**

History

#1 - 10/21/2012 12:01 PM - Greg Shah

Priority features:

Statements:

CREATE-SOAP-HEADER
CREATE-SOAP-HEADER-ENTRYREF

ATTRIBUTES:

SOAP-FAULT-STRING
ERROR-STATUS:ERROR-OBJECT-DETAIL

METHODS:

add-header-entry()
set-must-understand()
set-node()

#2 - 10/31/2012 01:37 PM - Greg Shah

- Target version set to Milestone 7

#3 - 01/29/2013 11:36 AM - Greg Shah

The full set of related features:

SOAP Header Object:

STATEMENTS: CREATE-SOAP-HEADER

ATTRIBUTES: NUM-HEADER-ENTRIES, INSTANTIATING-PROCEDURE, UNIQUE-ID
METHODS: add-header-entry(), get-header-entry()

SOAP Header Entryref Object:

STATEMENTS: CREATE-SOAP-HEADER-ENTRYREF
ATTRIBUTES: ACTOR, LOCAL-NAME, INSTANTIATING-PROCEDURE, MUST-UNDERSTAND, NAMESPACE-URI, UNIQUE-ID
METHODS: delete-header-entry(), get-node(), get-serialized(), set-actor(), set-must-understand(), set-node(), set-serialized()

SOAP Fault Object:

ATTRIBUTES: INSTANTIATING-PROCEDURE, SOAP-FAULT-ACTOR, SOAP-FAULT-CODE, SOAP-FAULT-DETAIL, SOAP-FAULT-STRING
ERROR-OBJECT-DETAIL attribute on ERROR-STATUS system handle

SOAP Fault Detail Object:

ATTRIBUTES: INSTANTIATING-PROCEDURE
METHODS: get-node(), get-serialized()

#4 - 02/14/2013 12:10 PM - Constantin Asofiei

```
Attributes/methods common to multiple resources:
- local-name: SOAP-header-entryref object handle, X-noderef object handle
- namespace-uri: Buffer object handle, ProDataSet object handle, SOAP-header-entryref object handle, Temp-tabl
e object handle, X-document object handle, X-noderef object handle
- get-node(): SOAP-fault-detail object handle, SOAP-header-entryref object handle
- get-serialized(): SOAP-fault-detail object handle, SOAP-header-entryref object handle
- INSTANTIATING-PROCEDURE: all
- UNIQUE-ID: as I recal, there is already an interface for this

SOAP Header Object (TYPE = SOAP-HEADER):
- STATEMENTS: CREATE SOAP-HEADER handle [ IN WIDGET-POOL widget-pool-name ]
- ATTRIBUTES: NUM-HEADER-ENTRIES(r/o, integer), INSTANTIATING-PROCEDURE, UNIQUE-ID
- METHODS: <logical> add-header-entry(<handle>), <logical> get-header-entry(<handle>, <index>)

SOAP Header Entryref Object (TYPE = SOAP-HEADER-ENTRYREF):
- STATEMENTS: CREATE SOAP-HEADER-ENTRYREF hshEntry [ IN WIDGET-POOL widget-pool-name ]
- ATTRIBUTES: ACTOR(r/o, character), LOCAL-NAME (r/o, character), INSTANTIATING-PROCEDURE, MUST-UNDERSTAND(r/o
, logical), NAMESPACE-URI(r/w, character), UNIQUE-ID
- METHODS: <logical> delete-header-entry(), <logical> get-node(<x-node-ref>), <longchar> get-serialized(), <lo
gical> set-actor(), <logical> set-must-understand(<logical>), <logical> set-node(<x-node-ref>), <logical> set-
serialized(<longchar>)

SOAP Fault Object (TYPE = SOAP-FAULT):
- ATTRIBUTES: INSTANTIATING-PROCEDURE, SOAP-FAULT-ACTOR(r/o, character), SOAP-FAULT-CODE(r/o, character), SOAP
-FAULT-DETAIL(r/o, handle), SOAP-FAULT-STRING(r/o, character)
- ERROR-OBJECT-DETAIL(r/o, handle) attribute on ERROR-STATUS system handle

SOAP Fault Detail Object (TYPE = SOAP-FAULT-DETAIL):
- ATTRIBUTES: INSTANTIATING-PROCEDURE
- METHODS: <logical> get-node(<x-node-ref>), <longchar> get-serialized()
```

#5 - 02/15/2013 11:02 AM - Constantin Asofiei

Implementation details:

- SOAPFactory - class with APIs for CREATE SOAP-HEADER and CREATE SOAP-HEADER-ENTRYREF statements
- SOAPEntity - interface with APIs for get-node and get-serializable methods, common for SOAP-fault-detail object handle, SOAP-header-entryref object handle

```
- METHODS: <logical> get-node(<x-node-ref>), <longchar> get-serialized()
```

- SOAPFault - interface with APIs for SOAP Fault Object, with SOAPFaultImpl the implementation class. Defines these attrs/methods:

```
- ATTRIBUTES: SOAP-FAULT-ACTOR(r/o, character), SOAP-FAULT-CODE(r/o, character), SOAP-FAULT-DETAIL(r/o, ha
ndle), SOAP-FAULT-STRING(r/o, character)
```

- SOAPFaultDetail - interface which defines the SOAP Fault Detail Object (although we can skip this as all methods for this object ar in SOAPEntity, and we can let the SOAPFaultDetailImpl class defining the resource)
- SOAPHeader - interface with APIs for these:

```
- ATTRIBUTES: NUM-HEADER-ENTRIES(r/o, integer)
- METHODS: <logical> add-header-entry(<handle>), <logical> get-header-entry(<handle>, <index>)
```

- SOAPHeaderDEtails - interface (extending SOAPEntity) with APIs for these:

```
- ATTRIBUTES: ACTOR(r/o, character), MUST-UNDERSTAND(r/o, logical)
- METHODS: <logical> delete-header-entry(), <logical> set-actor(), <logical> set-must-understand(<logical>
), <logical> set-node(<x-node-ref>), <logical> set-serialized(<longchar>)
```

Other notes:

- UNIQUE-ID, INSTATING-PROCEDURE are not touched.
- local-name (for SOAP-header-entryref object handle, X-noderef object handle) - I see that our X-noderef impl doesn't define it either, so I've ignore it
- namespace-uri (Buffer object handle, ProDataSet object handle, SOAP-header-entryref object handle, Temp-table object handle, X-document object handle, X-noderef object handle) - I see that is not touched by any other implemented resource, so I've ignore it
- SOAP Header and SOAP Header EntryRef both need access to NAME and PRIVATE-DATA attributes, but these are in CommonHandleChain (together with next-/prev-sibling). These need to be moved in a separate interface (and also extract them from HandleChain in a separate superclass).

#6 - 02/16/2013 09:08 AM - Greg Shah

I like the plan.

In regard to the "common" attributes (UNIQUE-ID, INSTANTIATING-PROCEDURE, LOCALNAME, NAMESPACE-URI, NAME, PRIVATE-DATA), the main update can ignore these. But after that is done, go ahead and build an update that properly handles these for all of the current resource types. I understand that will require some minor refactoring and creation of new interfaces (and implementing those in the right places).

#7 - 02/16/2013 12:07 PM - Constantin Asofiei

- File ca_upd20130216c.zip added

- File ca_upd20130216b.zip added

Attached update adds conversion support for SOAP-related attributes/methods and statements. Everything is stubbed, unless it was obvious what it needed to do.
Built on top of bzr revision 10173.

Notes:

- the documentation for some methods (like GET-NODE) states that "Returns a handle to an X-noderef" but it actually returns a logical value, and I think the X-noderef instance (in this case) is set to the passed handle var.
- SOAPEntity doesn't have its own implementation class (to allow SOAPFaultFetail and SOAPHeaderEntry to extend it), because SOAPFaultDetail doesn't have NAME attribute, while SOAPHeaderEntry has (so a compromise needs to be made).

#8 - 02/16/2013 12:26 PM - Greg Shah

I will review it later today. Be prepared to merge it Monday morning with the pending 10174-10176 that is currently being tested.

> the documentation for some methods (like GET-NODE) states that "Returns a handle to an X-noderef" but it actually returns a logical value, and I think the X-noderef instance (in this case) is set to the passed handle var

In each case where we are doing something different that the 4GL documentation, please make a statement like this:

"This method returns a logical even though the Progress 4GL documentation explicitly states otherwise. Actual working code has proven that the original documentation is incorrect. This implementation matches the actual runtime behavior of the 4GL."

> SOAPEntity doesn't have its own implementation class (to allow SOAPFaultFetail and SOAPHeaderEntry to extend it), because SOAPFaultDetail doesn't have NAME attribute, while SOAPHeaderEntry has (so a compromise needs to be made).

Aren't we about to move the NAME attribute (along with some other things like PRIVATE-DATA...) to a separate set of common interfaces? Just like we have Connectable, we would have the minimum set of common interfaces that can be implemented as needed.

#9 - 02/16/2013 12:40 PM - Constantin Asofiei

> In each case where we are doing something different that the 4GL documentation, please make a statement like this:

OK

> Aren't we about to move the NAME attribute (along with some other things like PRIVATE-DATA...) to a separate set of common interfaces? Just like we have Connectable, we would have the minimum set of common interfaces that can be implemented as needed.

I was refering to the fact that we can't put the implementation of these methods (get-node and get-serialized) in the same super class, as a class can't extend two super-classes (and SOAPHeaderEntryImpl would need in this case to extend two superclasses, one for name/private-data and one for these attributes).

#10 - 02/17/2013 11:25 AM - Greg Shah

OK, I understand now. Yes, this is an unfortunate side effect of our implementation. But at least the duplicate implementations are only needed in the minority of cases (most attributes are resource-specific).

#11 - 02/17/2013 12:05 PM - Greg Shah

Code Review Feedback:

1. The ErrorManager.WorkArea.soapFault member needs javadoc.

2. For int parameters to methods like getHeaderEntry(handle target, int idx), please switch these to double instead. I'm pretty sure that Progress would allow an index of 11.0 (or even 1.4111) to be passed to these methods. Your use of NumberType solves this for wrapper types, but by using double we can handle dec_literal and num_literal in a single signature.

3. Should we define new signatures using Text instead of character to "get ahead" of the coming work of making longchar interoperate?

#12 - 02/18/2013 03:37 AM - Constantin Asofiei

> 1. The ErrorManager.WorkArea.soapFault member needs javadoc.

OK

> 2. For int parameters to methods like getHeaderEntry(handle target, int idx), please switch these to double instead. I'm pretty sure that Progress would allow an index of 11.0 (or even 1.4111) to be passed to these methods. Your use of NumberType solves this for wrapper types, but by using double we can handle dec_literal and num_literal in a single signature.

For these, 4GL allows only integer and int64 values (confirmed by testing).

> 3. Should we define new signatures using Text instead of character to "get ahead" of the coming work of making longchar interoperate?

Same, from testing, 4GL doesn't allow a longchar for character or character (event character constant) for longchar, in the case of these SOAP methods.

#13 - 02/18/2013 07:58 AM - Greg Shah

> For these, 4GL allows only integer and int64 values (confirmed by testing).

OK. Please use long instead of int and add javadoc to explain that based on testing, only integer and int64 values are allowed in the 4GL.

> Same, from testing, 4GL doesn't allow a longchar for character or character (event character constant) for longchar, in the case of these SOAP methods.

OK. Please add javadoc comment to explain this.

The idea of adding the javadoc is to make it clear to future readers that:
- these limitations are intentional
- that these limitations duplicate the 4GL limitations

#14 - 02/18/2013 09:14 AM - Constantin Asofiei

- File ca_upd20130218c.zip added

Added update merged with bzr revision 10177.

#15 - 02/18/2013 09:24 AM - Greg Shah

I am OK with the update. The only thing I suggest is that the new javadoc entries have an unmatched trailing double quote character. Search on 4GL." to find these.

I don't see a reason for runtime regression testing on this one. Apply the change in staging, run conversion and confirm that there is no unexpected change. if so, then you will check in next. I've chown'd staging to cas.

#16 - 02/18/2013 09:37 AM - Constantin Asofiei

I've removed the trailing quotes. The conversion has been started on staging (with previous generated sources backedup).

#17 - 02/18/2013 09:37 AM - Constantin Asofiei

- File ca_upd20130218e.zip added

And the latest update...

#18 - 02/18/2013 02:25 PM - Constantin Asofiei

No changes in generated MAJIC sources, committed to bzr revision 10178.

#19 - 01/03/2014 07:09 AM - Constantin Asofiei

I've started working on finishing the SOAP Fault support (part of #1645). The SOAP Header support is highly dependent on #2208 - as we need custom services to test it.

#20 - 01/09/2014 03:05 PM - Constantin Asofiei

All updates and discussions for this task was moved to #1645.

#21 - 01/23/2014 04:41 PM - Greg Shah

- Status changed from New to Closed

#22 - 11/16/2016 11:42 AM - Greg Shah

- Target version changed from Milestone 7 to Runtime Support for Server Features

Files

ca_upd20130216b.zip	83 KB	02/16/2013	Constantin Asofiei
ca_upd20130216c.zip	1.01 KB	02/16/2013	Constantin Asofiei
ca_upd20130218c.zip	88.9 KB	02/18/2013	Constantin Asofiei
ca_upd20130218e.zip	88.9 KB	02/18/2013	Constantin Asofiei