Base Language - Feature #12007

implement Progress.Lang.SoapFaultError

10/15/2026 07:18 PM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**reviewer:**	
vendor_id:	GCD	**production:**	No
case_num:		**env_name:**	
version_reported:		**topics:**	
version_resolved:			

Description

Related issues:

Related to Base Language - Feature #1644: implement SOAP support	**Closed**	**02/13/2013**	**07/12/2013**
Related to Base Language - Feature #1645: implement web services support	**Closed**	**02/20/2013**	**08/09/2013**
Related to Base Language - Bug #4602: fixes for OO 4GL and structured error h...	**Review**		
Related to Base Language - Feature #4658: OO serialization support	**WIP**		
Related to Base Language - Feature #9488: implement all built-in OO classes u...	**WIP**		

History

#1 - 10/15/2026 07:24 PM - Greg Shah

Implement Progress.Lang.SoapFaultError

Project: Base Language
Related: #1644 (SOAP / web services client support), #11933 (SOAP-FAULT object on the NO-ERROR path), #11934 (transport failures reported as SOAP faults), #4602 (structured error handling), #4658 (OO serialization), #9488 (built-in OO classes)

Where This Fits

#1644 and #1645 built FWD's SOAP client with the 4GL's handle-based fault model: the SOAP-FAULT and SOAP-FAULT-DETAIL objects, ERROR-STATUS:ERROR-OBJECT-DETAIL and error 11506. **The structured error class was never part of that work**, and neither ticket mentions it.

#11933 (branch 11933a, rev 16796, in review) fixes the **NO-ERROR** path. Its 30 testcases (testcases rev 1943) establish when the SOAP-FAULT is published to ERROR-OBJECT-DETAIL, and when it is deleted. They also establish the boundary this task starts from. **No SOAP-FAULT is published when the fault is caught by a CATCH block or raised as an error object by UNDO, THROW.** On that path, the 4GL makes the fault reachable only through the caught error. That error is a Progress.Lang.SoapFaultError, and its SoapFault property holds the SOAP-FAULT handle.

This task covers only that structured path. It builds on 11933a and must not change the NO-ERROR behaviour #11933 establishes.

Problem

oo/lang/SoapFaultError.java exists, extends SysError and is marked CVT_LVL_FULL|RT_LVL_STUB. Nothing in FWD creates an instance, and its soapFault field is never assigned. ErrorManager.handleSoapFault() (server side, near ErrorManager.java:5709) raises error 11506 through recordOrThrowError. When that error becomes an error object, it is built as a plain SysError. So a CATCH e AS Progress.Lang.SoapFaultError never matches, and code that tests IsA(get-class(Progress.Lang.SoapFaultError)) never sees one.

Work

1. Establish the OpenEdge 12.8 behaviour first, by extending the #11933 testcases with CATCH variants:
   - Which class is raised for a SOAP fault under a CATCH or UNDO, THROW? Is it always SoapFaultError, or only for some call forms (RUN ... IN hPort, a function call on the port, an asynchronous request)?
   - What do GetMessage(1) and GetMessageNum(1) return? The expectation is 11506 with the same text as the NO-ERROR path.
   - What is the lifetime of the SOAP-FAULT handle held by SoapFault? Is it deleted with the error object, or when the CATCH block ends, and

    - is it still valid after a RETURN ERROR or re-throw of the caught error?
    - Does a SoapFaultError thrown in called code still interact with the "nearest NO-ERROR statement" rule recorded in #11933-3? That note says such a fault is not recorded when it is raised as an error object.
2. In the structured path of handleSoapFault (after 11933a's restructuring), build the SOAP-FAULT and SOAP-FAULT-DETAIL objects with the existing SOAPFactory. Raise a SoapFaultError that carries them, instead of the generic SysError. Reuse the error-object construction that ErrorManager already uses for SysError (the legacyError builder), so that the message list, CallStack and the rest are populated as they are for any other system error.
3. Implement the constructor and the SoapFault getter. Give the handle the lifetime the tests establish.
4. Do not resolve how the handle field serializes. That is #4658's question (#4658-22).
5. Correct the LegacyResourceSupport markings on the class, the constructor and getSoapFault.

Testcases

Add these to the #11933 suite, against the same calculator WSDL:

- a CATCH of Progress.Lang.SoapFaultError, a CATCH of Progress.Lang.SysError and a CATCH of Progress.Lang.Error around a faulting call, checking which block catches the error and what class it is;
- the SoapFault attributes and SOAP-FAULT-DETAIL:GET-SERIALIZED() read inside the CATCH block;
- ROUTINE-LEVEL / BLOCK-LEVEL ON ERROR UNDO, THROW propagating the error to a caller that catches it;
- the handle's validity after the CATCH block, after DELETE OBJECT of the error, and after a re-throw;
- confirmation that every #11933 NO-ERROR test still passes unchanged;
- a transport failure, which is not a SOAP fault, to confirm the class is not raised for it (see #11934).

#2 - 10/15/2026 07:24 PM - Greg Shah

- Related to Feature #1644: implement SOAP support added

#3 - 10/15/2026 07:25 PM - Greg Shah

- Related to Feature #1645: implement web services support added

#6 - 10/15/2026 07:25 PM - Greg Shah

- Related to Bug #4602: fixes for OO 4GL and structured error handling added

#7 - 10/15/2026 07:25 PM - Greg Shah

- Related to Feature #4658: OO serialization support added

#8 - 10/15/2026 07:25 PM - Greg Shah

- Related to Feature #9488: implement all built-in OO classes using the open source ADE code from 12.8.4 added