Base Language - Feature #1606

implement persistent procedures/super procedures

10/19/2012 10:03 AM - Greg Shah

Status:	Closed	**Start date:**	11/09/2012
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Runtime Support for Server Features		
billable:	No	**version:**	
vendor_id:	GCD		

Description

Subtasks:

Feature # 1920: implement persistent procedures	**Closed**
Feature # 1921: implement super-procedures	**Closed**

Related issues:

Related to Base Language - Feature #2138: finish SEARCH-TARGET support	**Closed**	**08/09/2013**	**08/16/2013**
Related to Base Language - Feature #2311: rewrite parsing of handle chains an...	**Closed**		
Blocks Base Language - Feature #1607: implement full named events support	**Closed**	**01/29/2013**	**07/01/2013**

History

#1 - 10/19/2012 10:05 AM - Greg Shah

- Estimated time set to 120.00

The following support is needed: RUN PERSISTENT, DELETE PROCEDURE/OBJECT, RUN IN, FUNCTION IN, DYNAMIC FUNCTION/DYNAMIC-FUNCTION IN, RUN SUPER, procedure handle support (especially THIS-PROCEDURE, TARGET-PROCEDURE, SOURCE-PROCEDURE, super(), internal-entries, private-data, next-sibling, prev-sibling, first-procedure, last-procedure,super-procedures, persistent, type, get-signature()), add-super-procedure() (especially on the session handle), remove-super-procedure(), more “dynamic”/indirect execution model for internal procedures/functions.

#2 - 10/31/2012 01:47 PM - Greg Shah

- Target version set to Milestone 7

#3 - 11/08/2012 10:21 AM - Greg Shah

- Assignee set to Constantin Asofiei

#4 - 11/09/2012 08:04 AM - Constantin Asofiei

- Status changed from New to WIP

#5 - 11/09/2012 08:10 AM - Greg Shah

One of the major design decisions will relate to how we handle the super procedure chain. The idea of super procedures and the ability to walk the chain of parents is the Progress 4GL approach to add a kind of runtime inheritance to the language. This was an attempt to provide a similar feature to true object-oriented languages which were becoming popular in the 1990s.

The thing is that this chain is built at runtime NOT at compile time. AND it can be dynamically changed at runtime. In practice, it is my understanding that actually modifying the super procedure chain is not often done. However, any usage of that feature at all will invalidate certain obvious approaches in Java.

For example, we might wish to use real Java inheritance to implement super procedures. While I can hope that there may be some applications that can live in this approach, we MUST implement the full and correct dynamic runtime approach first.

I want you to think about this issue carefully record put your thoughts here.

#6 - 11/09/2012 08:18 AM - Greg Shah

One approach is to use reflection to fully and "naturally" implement the features. We know that will work and it can be done. A downside of this is a potential loss in performance. Possibly some form of caching of the reflection results could limit the cost to the first time a given method is called.

Another idea that may at least limit the "damage" of reflection would be to have the conversion create custom Java interfaces that represent the API provided by the super procedure(s). The various super procedures could implement the interfaces. If the super procedures cannot implement the interface, we could provide a dynamic proxy that implements it and which maps those calls to the right backing methods in the converted procedure. Again, if reflection is needed, then some caching would be warranted.

#7 - 11/09/2012 08:41 AM - Constantin Asofiei

I agree that we must work around the fact that the super-procedure chain is built at runtime NOT at compile time. You mention inheritance, but there are IMO some strong reasons on why this approach can not be used:
1. Java does not support "dynamic" inheritance. In 4GL, the super-procedure chain is determined at runtime, and can change if i.e external procB is called from procC instead of procA. Even if we could determine the call chain during conversion, how do we handle cases when the target external procedure of a RUN statement is computed at runtime ?
2. Even if during conversion we make the converted java class for external procedure procB to "extend" the external procedure procA (as conversion determined that procB is called from procA and procB also has a RUN SUPER statement), this might complicates things. Assume that when RUN SUPER is called, we are in the internal procedure foo within external procedure procB (called from procB). This calls the foo procedure within external procedure procA, which in turn calls the bar internal procedure within procA. But, if procB also has a bar internal procedure and the conversion rules end up with a converted code looking like:

```
public class ProcA
{

   public void foo()
   {
      System.out.println("proca: foo");
      bar();
   }

   public void bar()
   {
      System.out.println("proca: bar");
   }
}
public class ProcB
extends ProcA
{
   public void foo()
   {
      super.foo();
      System.out.println("procb: foo");
   }

   public void bar()
   {
      System.out.println("procb: bar");
   }

   public static void main(String[] args)
   {
      new ProcB().foo();
   }
}
```

the output will be like:

```
proca: foo
procb: bar
procb: foo
```

As you can see, the bar procedure is called from the procB external procedure, which is not what the 4GL wants (when RUN SUPER sends control to the "super" procedure, I think all other procedure calls will be inside that super procedure).

What I'm trying to say is that we can't rely on Java inheritance to solve this... instead, we will need to keep a call chain (which saves each external procedure instance from the call stack) in our own data structure and use that.

#8 - 11/09/2012 12:07 PM - Constantin Asofiei

I've added #1920 for persistent procedures; I suggest to leave this task for tracking the implementation of the super-procedures; this includes the following 4GL features:
FUNCTION ... IN SUPER
PROCEDURE ... IN SUPER
RUN SUPER statement
SUPER function

SESSION handle:
super-procedures
add-super-procedure()
remove-super-procedure()

THIS-PROCEDURE/TARGET-PROCEDURE/SOURCE-PROCEDURE handle:
super-procedures
add-super-procedure()
remove-super-procedure()

#9 - 09/18/2013 08:49 AM - Greg Shah

- Status changed from WIP to Closed

#10 - 11/16/2016 11:43 AM - Greg Shah

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