Ada Issue 00354 Group Execution-Time Budgets

!standard D.14.2 (01) 05-03-24 AI95-00354/08
!class amendment 03-09-27
!status Amendment 200Y 04-06-25
!status ARG Approved 7-0-1 04-06-13
!status work item 03-09-27
!status received 03-09-27
!priority Low
!difficulty Medium
!subject Group execution-time budgets

!summary

This AI proposes a child package of Ada.Execution_Time to allow more than one task to share an execution-time budget.

!problem

Currently Ada has no mechanisms to allow the implementation of aperiodic servers such as sporadic servers and deferrable servers. This severely limits the language's ability to handle aperiodic activities at anything other than a background priority. The fundamental problem that prohibits the implementation of a periodic server algorithms is that tasks cannot share CPU budgets.

!proposal

(See wording.)

!wording

Add new section:

D.14.2 Group Execution Time Budgets

This clause describes a language-defined package to assign execution time budgets to groups of tasks.

Static Semantics

The following language-defined library package exists:

```
with System;
package Ada.Execution_Time.Group_Budgets is

  type Group_Budget is limited private;

  type Group_Budget_Handler is access
      protected procedure (GB : in out Group_Budget);
  type Task_Array is array (Positive range <>) of
                         Ada.Task_Identification.Task_Id;

  Min_Handler_Ceiling : constant System.Any_Priority :=
    *implementation-defined*;

  procedure Add_Task (GB : in out Group_Budget;
                  T  : in Ada.Task_Identification.Task_Id);
  procedure Remove_Task (GB : in out Group_Budget;
                    T  : in Ada.Task_Identification.Task_Id);
  function Is_Member (GB : Group_Budget;
                  T  : Ada.Task_Identification.Task_Id) return Boolean;
  function Is_A_Group_Member (
     T : Ada.Task_Identification.Task_Id) return Boolean;
  function Members (GB : Group_Budget) return Task_Array;

  procedure Replenish (GB : in out Group_Budget; To : in Time_Span);
  procedure Add (GB : in out Group_Budget; Interval : in Time_Span);
  function Budget_Has_Expired (GB : Group_Budget) return Boolean;
  function Budget_Remaining (GB : Group_Budget) return Time_Span;

  procedure Set_Handler (GB      : in out Group_Budget;
                    Handler : in Group_Budget_Handler);
  function Current_Handler (GB : Group_Budget) return Group_Budget_Handler;
  procedure Cancel_Handler (GB        : in out Group_Budget;
                      Cancelled : out Boolean);

  Group_Budget_Error : exception;
private
   --  not specified by the language
end Ada.Execution_Time.Group_Budgets;
```

The type Group_Budget represents an execution time budget to be used by a group of tasks. The type Group_Budget needs finalization (see 7.6). A task can belong to at most one group. Tasks of any priority can be added to a group.

An object of type Group_Budget has an associated non-negative value of type Time_Span known as its *budget*, which is initially Time_Span_Zero.
The type Group_Budget_Handler identifies a protected procedure to be executed by the implementation when the budget is *exhausted*, that is, reaches zero. Such a protected procedure is called a *handler*.

An object of type Group_Budget also includes a handler, which is a value of type Group_Budget_Handler. The handler of the object is said to be *set* if it is not null and *cleared* otherwise. The handler of all Group_Budget objects is initially cleared.

Dynamic Semantics

The procedure Add_Task adds the task identified by T to the group GB; if that task is already a member of some other group, Group_Budget_Error is raised.