STRAST

A Model-based Framework for developing Real-Time Safety Ada Systems

Emilio Salazar, Alejandro Alonso,
Miguel A. de Miguel, Juan A. de la Puente

dit
UPM

Índice

- Introducción
- UML & MARTE
- Framework basado en modelos
- Generación de Ravenscar Ada 2005
- Caso de uso: UPMSat2
- Trabajo futuro

Introducción

- Objetivos
  - Generación de Ravenscar Ada 2005 a partir de modelos UML anotados con un subconjunto de MARTE
  - Integrar análisis de planificabilidad en las herramientas de generación de código
  - Consistencia entre el comportamiento temporal del modelo y el del código generado
  - Trazabilidad entre modelo y código (y viceversa)

Introducción

- Implementación del framework
  - Aprovechar las ventajas del MDD
  - Basado en estándares
    - » UML & MARTE
    - » QVTo, MTL
  - Integrable en varias herramientas
    - » RSA
    - » Eclipse
    - » Papyrus

UML & MARTE

- MARTE
  - Perfil estándar de la OMG para modelar sistemas de tiempo real
  - Extiende UML
    - » Perfiles
    - » Estereotipos
  - Muy amplio
  - Genérico

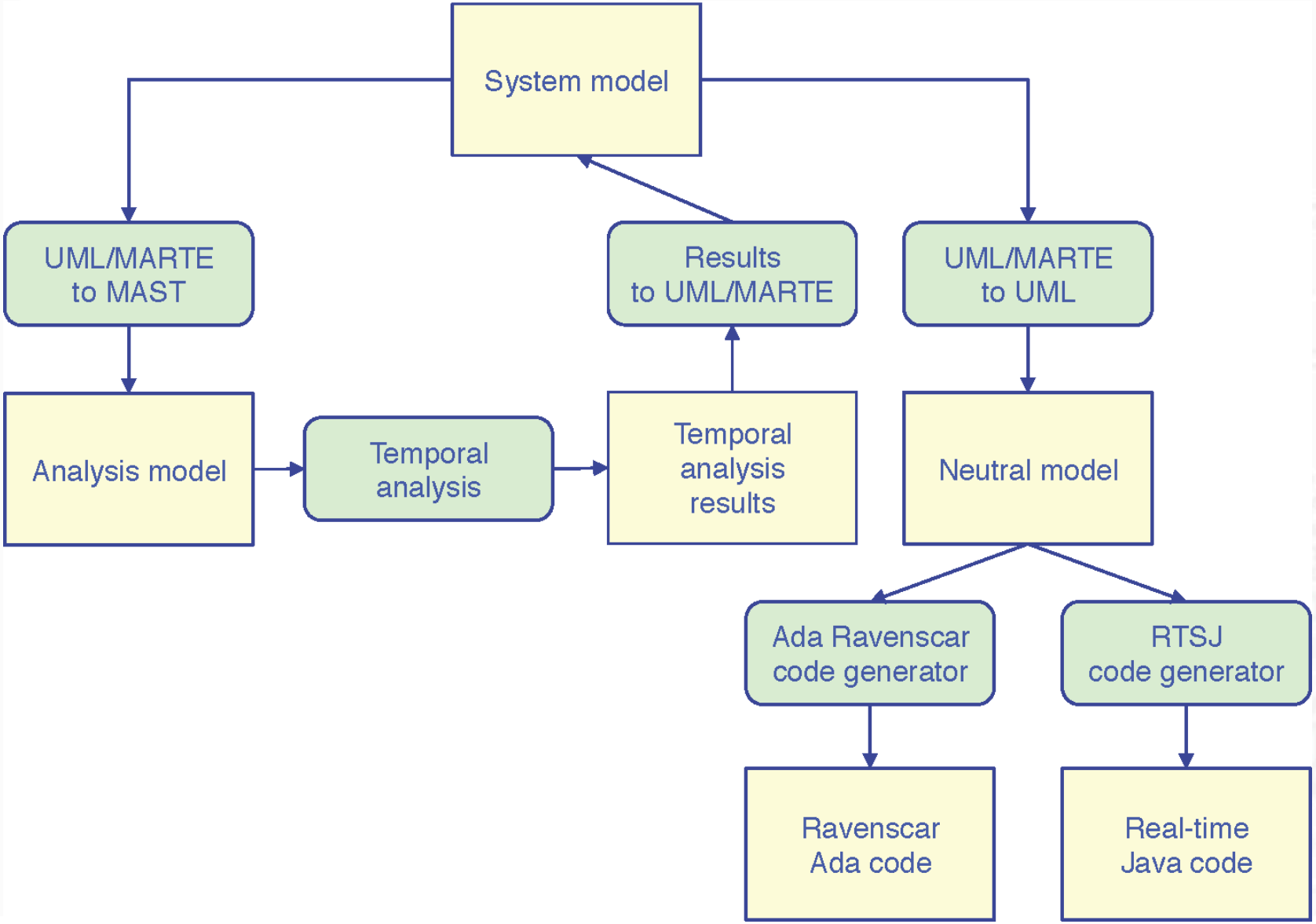
UML & MARTE

- Análisis de planificabilidad
  - Subperfiles
    - » Schedulability Analysis Modeling (SAM)
    - » Generic Quantitative Analysis Modeling (GQAM)
  - Integración con MAST
  - Realimentación de los resultados en los modelos de entrada

UML & MARTE

- Generación de código
  - Subperfiles
    - » Generic Resource Modeling (GRM)
    - » Generic Quantitative Analysis Modeling (GQAM)
  - Gestión de información duplicada
  - Restricciones en el modelo UML de entrada

Framework



Framework

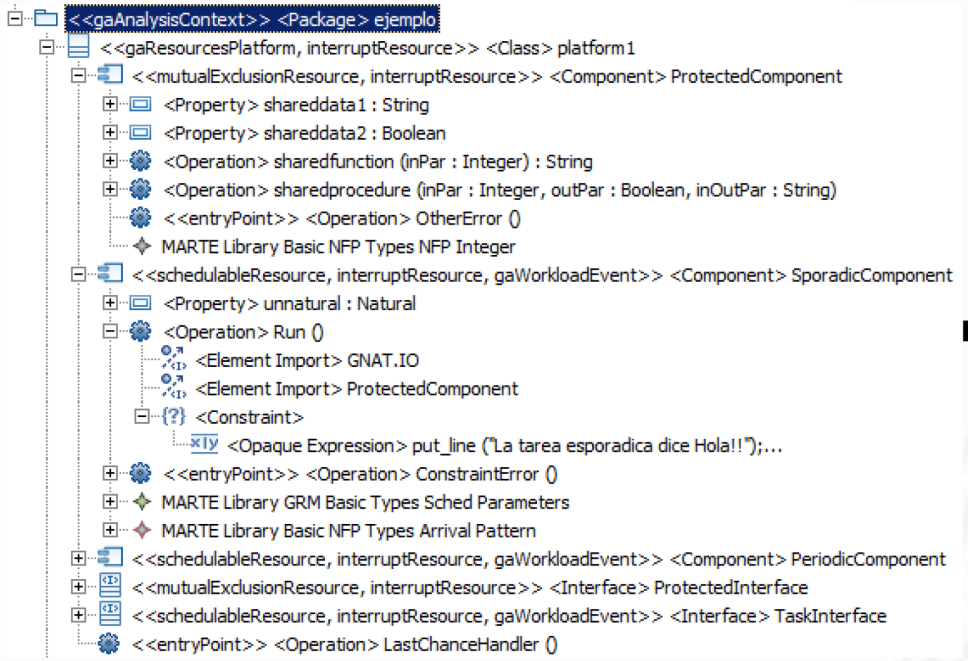
- Aplicar MDE en el desarrollo de STR
  - Modelo de sistema
    - » UML & MARTE
    - » PIM
  - Modelo de análisis
    - » MAST
  - Modelo neutral
    - » UML
    - » Facilitar la generación de varios lenguajes de programación

Framework

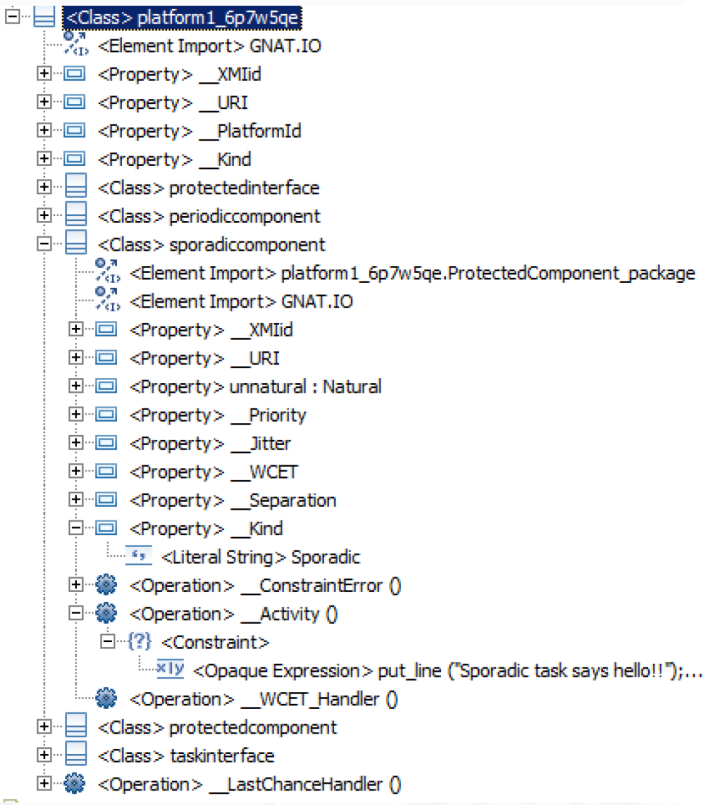
- Aplicar MDE en el desarrollo de STR
  - Código fuente
    - » Ada 2005
      - Compatible con el perfil de Ravenscar
    - » Java Real-Time

Generación de código

MARTE



NEUTRAL



Generación de código

- QVT Operational

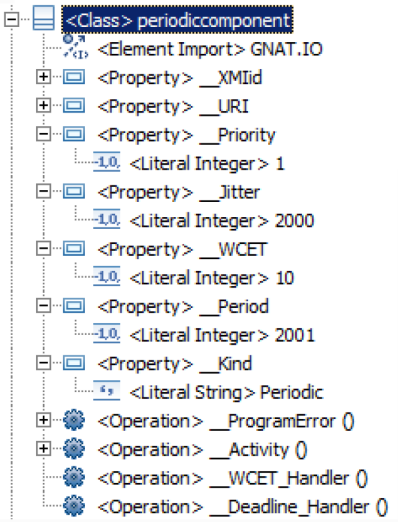
```
-- Generates the NEUTRAL model of each MARTE platform
query Set(SAM::SaAnalysisContext)::generateAdaNeutralModel(inout NEUTRALModel : UML::Model)
{
    self [GQAM::GaAnalysisContext] -> forEach (context)
    {
        context.platform [GQAM::GaResourcesPlatform] -> forEach (platform)
        {
            -- Stores in MARTEResources all MARTE supported element that will be converted to NEUTRAL
            platform.ProcessMARTEResources();

            -- Replicates in the NEUTRAL model the package structure
            var NEUTRALRootPackge : UML::Package := platform.ProcessNEUTRALPackages(MARTEResources, NEUTRALPackag

            -- Converts all Passive elements found in MARTEResources to NEUTRAL elements. Then it stores the NEUT
            -- element into NEUTRALResources and allocates the NEUTRAL resource where NEUTRALPackages dict specif
            PassiveResources();

            -- Converts all actives elements found in MARTEResources to NEUTRAL elements. Then it stores the NEUT
            -- element into NEUTRALResources and allocates the NEUTRAL resource where NEUTRALPackages dict specif
            ActiveResources();

            -- Converts MARTE type's references to NEUTRAL type's references
            ProcessTypes();

            -- Adds all imports to the NEUTRAL model
            ProcessImports();
```

Generación de código

```
task body Periodic_Task_Type is
      Interval   :     constant Time_Span          := Milliseconds (Period);
      Next_Time  :     Time                        := Clock + Milliseconds (Offset);
      Id         :     aliased constant Task_Id    := Current_Task;
      WCET_Timer :     Ada.Execution_Time.Timers.Timer (Id'Access);
      Canceled   :     Boolean;
begin
      delay until Next_Time;
      Next_Time := Clock + Interval;
      loop
            delay until Next_Time;

            Next_Time := Next_Time + Interval;

            -- Temporizador para el WCET de la tarea
            Ada.Execution_Time.Timers.Set_Handler (     WCET_Timer,
                                                        Milliseconds (WCET_Budget),
                                                        WCET_Ovr_Handler);

            -- Temporizador para el plazo de la tarea
            Ada.Real_Time.Timing_Events.Set_Handler (   Deadline_Overrun,
                                                        Next_Time + Interval,
                                                        Deadline_Ovr_Handler);

            Periodic_Activity;

            Ada.Real_Time.Timing_Events.Cancel_Handler (Deadline_Overrun, Canceled);

      end loop;

      exception
            when e : Constraint_Error  => Constraint_Error_Handler.all (e);
            when e : Program_Error     => Program_Error_Handler.all (e);
            when e : Storage_Error     => Storage_Error_Handler.all (e);
            when e : Tasking_Error     => Tasking_Error_Handler.all (e);
            when e : others            => Other_Error_Handler.all (e);
end Periodic_Task_Type;
```

Generación de código

NEUTRAL



Sin WCET

Con WCET

Ada

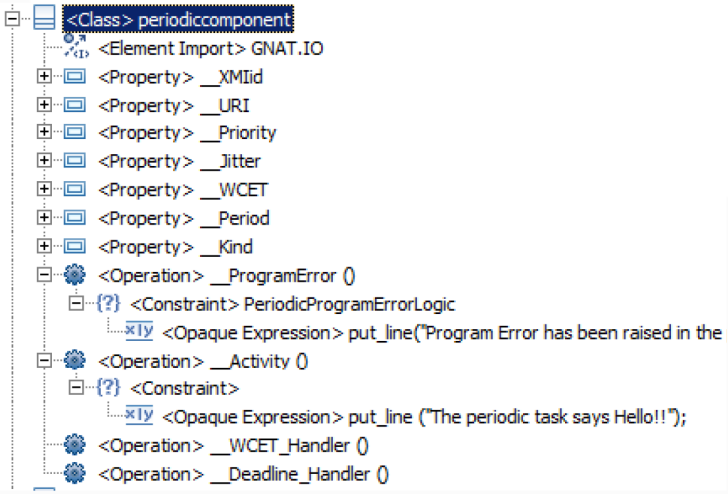
```
package periodiccomponent_periodic_task is
     new uml2ada.periodic_tasks(
          Priority => 1,
          Period => 2001,
          Offset => 2000,
          Periodic_Activity => Activity,
          Deadline_Ovr_Handler => DeadlineHandler.Handler'Access,
          Constraint_Error_Handler => Default_Exception_Handler,
          Program_Error_Handler => ProgramError'Access,
          Storage_Error_Handler => Default_Exception_Handler,
          Tasking_Error_Handler => Default_Exception_Handler,
          Other_Error_Handler => Default_Exception_Handler);
```

```
package periodiccomponent_periodic_task is
     new uml2ada.wcet_periodic_tasks (
          Priority => 1,
          Period => 2001,
          Offset => 2000,
          Periodic_Activity => Activity,
          Deadline_Ovr_Handler => DeadlineHandler.Handler'Access,
          WCET_Budget => 10,
          WCET_Ovr_Handler => WCETHandler.Handler'Access,
          Constraint_Error_Handler => Default_Exception_Handler,
          Program_Error_Handler => ProgramError'Access,
          Storage_Error_Handler => Default_Exception_Handler,
          Tasking_Error_Handler => Default_Exception_Handler,
          Other_Error_Handler => Default_Exception_Handler);
```

- Acceleo MTL

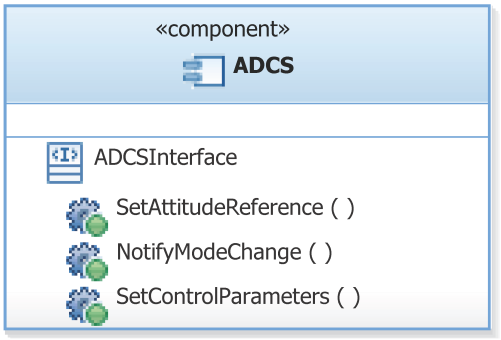
```
[template private EntryHeader(op : Operation) ? (op <> null and isEntry(op))]
[if (op.ownedParameter -> isEmpty())]
entry [getOperationName(op)/]
[else]
entry [getOperationName(op)/] ([Parameters(op)/])
[/if]
[/template]

[template private OperationBody(op : Operation)]
[if (hasUserCode(op))]
is
begin
[getUserCode(op)/]
[else]
is
    pragma Compile_Time_Warning (True, "[op.name/] is unimplemented");
begin
    raise Program_Error with "[op.name/] is unimplemented";
[if (op.isFunction())]
    return res:[op.getReturnResult().type.name/];
[/if]
[/if]
end [getOperationName(op)/];
[/template]
```

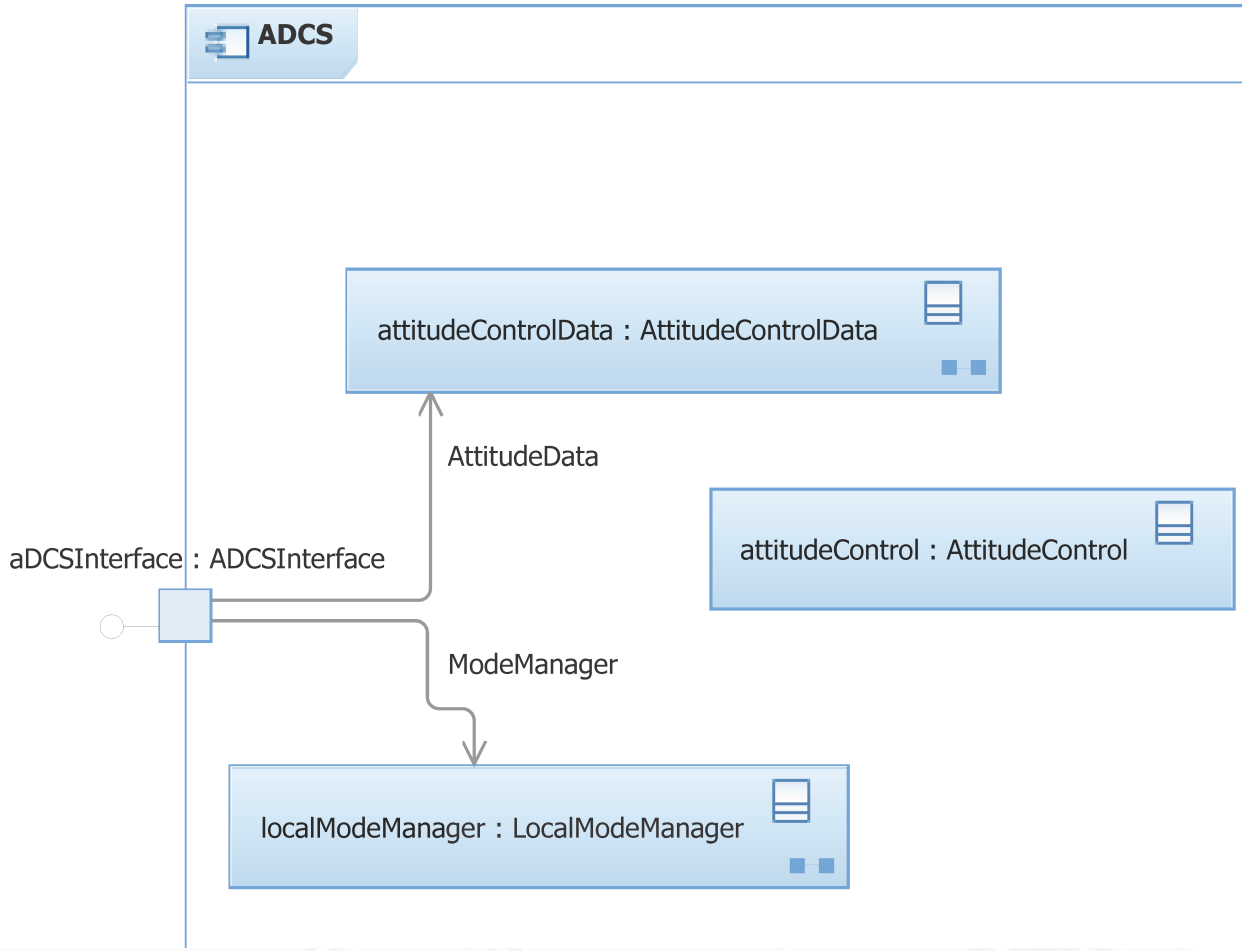
Generación de código



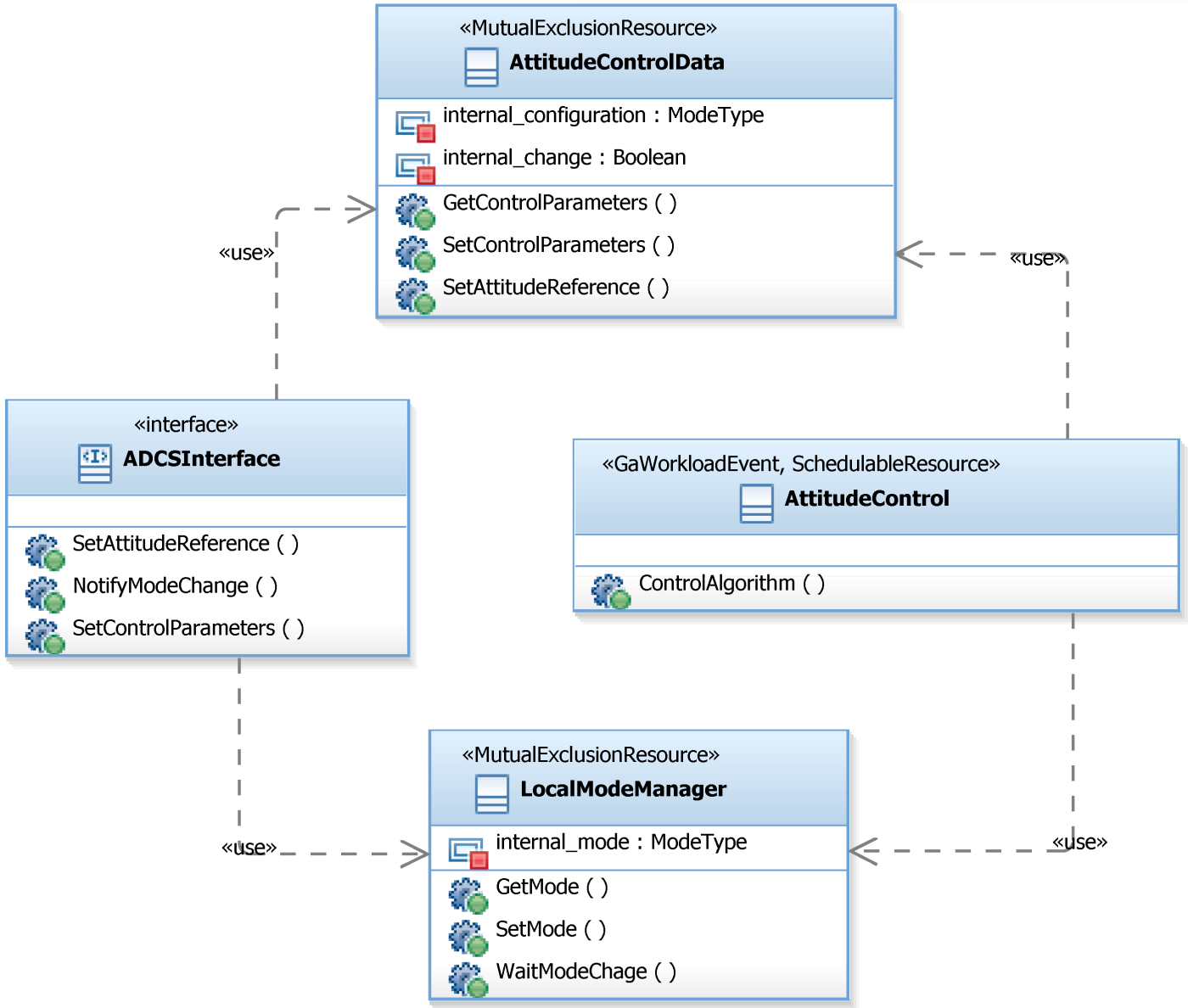

```
package platform1_cnij5so.periodiccomponent_package is

      -- Actividad periodica a ejecutar
      procedure Activity;

      -- OP con el manejador del vencimiento del WCET de la tarea
      protected WCETHandler is
            procedure Handler (Event : in out Ada.Execution_Time.Timers.Timer);
      end WCETHandler;

      -- OP con el manejador del vencimiento del plazo de la tarea
      protected DeadlineHandler is
            procedure Handler (Event : in out Timing_Event);
      end DeadlineHandler;

      -- Código del usuario para la excepción Program error
      procedure ProgramError (e : in Exception_Occurrence);

      -- Instanciación del paquete genérico
      package periodiccomponent_periodic_task is
            new uml2ada.periodic_tasks(
            Priority => 1,
            Period => 2001,
            Offset => 2000,
            Periodic_Activity => Activity,
            Deadline_Ovr_Handler => DeadlineHandler.Handler'Access,
            Constraint_Error_Handler => Default_Exception_Handler,
            Program_Error_Handler => ProgramError'Access,
            Storage_Error_Handler => Default_Exception_Handler,
            Tasking_Error_Handler => Default_Exception_Handler,
            Other_Error_Handler => Default_Exception_Handler);

end platform1_cnij5so.periodiccomponent_package;
```

Caso de uso: UPMSat2





Caso de uso: UPMSat2



Trabajo futuro

- Generación de particionado (Xtratum)
- Generación (parcial) de la lógica de negocio