Date_01.lkt

```
/***************************************************************************
 * SOURCEFILE => Data_01.lkt
 ***************************************************************************/

lektaProject

        projectHead
                projectDescriptor       : "Date_01"
                projectLanguageScope    : [ es ]
                projectOperationScope   : [ test ]
                projectInterfaceScope   : [ dialogue ]
                projectCompileOutput    : ".Date_01.olk"

        projectSetup
                setupParserRoots = Command

        classModel
                classDef:ElementRange ( askInfo:
                                { ’day’, ’hour’} )
                classDef:StructureComplex ( askCommand, Command : (askInfo) )

        lexicalModel forLanguage es
                ("dime el dÃ-a",   askCommand,      (askInfo:’day’))
                ("dime la hora",  askCommand,      (askInfo:’hour’))

        grammaticalModel forLanguage es
                (R1: [ Command −> askCommand ]
                        { ^ <− #1; } )

        conversationalModel
                MindBoardStructure: {
                        (Mind / Command)
                }

                SensoSchemata
                        (SensoScheme SenseCommand : [Command]
                                SensoAction {
                                        $MINDBOARD@Mind <− #OBJSENSO−1;
                                }
                        )

                RespondoSchemata
                        (RespondoScheme Command: [Command]
                                RespondoAction {
                                        ^OBJRESPONDO <− $MINDBOARD@Mind;
                                }
                        )

        scriboModel forLanguage es
                ScriboSchemata
                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’day’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Es el dÃ-a ");
                                        SetMainAnswer(ClockAskDayOfTheMonth());
                                }
                        )

                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’hour’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Son las ");
                                        SetMainAnswer(ClockAskHour());
                                        SetMainAnswer(":");
                                        SetMainAnswer(ClockAskMinute());
                                }
                        )

```

jueves julio 18, 2013

Date_01.slk

Page 1/1

```
LaunchLektaKernel()

UseProject (ProjectCompile        : ".Date_01.olk")

DisplayProcessAbstractOn

CreateDialogue()
```

```
/****************************************************************************
 * SOURCEFILE => Data_02.lkt
 ****************************************************************************/

lektaProject

        projectHead
                projectDescriptor       : "Date_02"
                projectLanguageScope    : [ es ]
                projectOperationScope   : [ test ]
                projectInterfaceScope   : [ dialogue ]
                projectCompileOutput    : ".Date_02.olk"

        projectSetup
                setupParserRoots = Command

        classModel
                classDef:ElementRange ( askInfo:
                        { ’day’, ’hour’} )
                classDef:StructureComplex ( askCommand, Command : (askInfo) )
                classDef:StructureBatch ( Commands : (Command ) )
                classDef:StructureComplex ( Task : ( Commands ) )

        lexicalModel forLanguage es
                ("dime el dÃ-a",   askCommand,      (askInfo:’day’))
                ("dime la hora",  askCommand,      (askInfo:’hour’))

        grammaticalModel forLanguage es
                (R1: [ Command −> askCommand ]
                        { ^ <− #1; } )

        conversationalModel
                MindBoardStructure: {
                        (Mind / Task)
                }

                SensoSchemata
                        (SensoScheme SenseCommand : [Command]
                                SensoAction {
                                        BatchInsertEnd($MINDBOARD@Mind.Commands, #OBJSENSO−1);
                                }
                        )

                RespondoSchemata
                        (RespondoScheme DS: [Command]
                                RespondoCapture {
                                        (BatchSize($MINDBOARD@Mind.Commands) > 0)
                                }
                                RespondoAction {
                                        Command command;
                                        BatchExtractInit($MINDBOARD@Mind.Commands,command);
                                        ^OBJRESPONDO <− command;
                                        RespondoRetry();
                                }
                        )

        scriboModel forLanguage es
                ScriboSchemata
                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’day’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Es el dÃ-a ");
                                        SetMainAnswer(ClockAskDayOfTheMonth());
                                        SetMainAnswer(". ");
                                }
                        )

                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’hour’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Son las ");
                                        SetMainAnswer(ClockAskHour());
                                        SetMainAnswer(":");
                                        SetMainAnswer(ClockAskMinute());
                                        SetMainAnswer(". ");
                                }
                        )

```

jueves julio 18, 2013

Date_02.slk

```
LaunchLektaKernel()

UseProject (ProjectCompile        : ".Date_02.olk")

DisplayProcessAbstractOn

CreateDialogue()
```

```
/****************************************************************************
 * SOURCEFILE => Data_03.lkt
 ****************************************************************************/

lektaProject

        projectHead
                projectDescriptor       : "Date_03"
                projectLanguageScope    : [ es ]
                projectOperationScope   : [ test ]
                projectInterfaceScope   : [ dialogue ]
                projectCompileOutput    : ".Date_03.olk"

        projectSetup
                setupParserRoots = Command

        classModel
                classDef:ElementBool (boolean)
                classDef:ElementInt (integer)
                classDef:ElementLiteral (string)
                classDef:ElementRange ( askInfo:
                        { ’date’, ’day’, ’hour’} )
                classDef:StructureComplex ( askCommand, Command : (askInfo) )
                classDef:StructureBatch ( Commands : (Command ) )
                classDef:StructureComplex ( Task : ( Commands ) )

        lexicalModel forLanguage es
                ("dime el dÃ-a",   askCommand,      (askInfo:’day’))
                ("dime la hora",  askCommand,      (askInfo:’hour’))
                ("dime la fecha",  askCommand,     (askInfo:’date’))

        grammaticalModel forLanguage es
                (R1: [ Command −> askCommand ]
                        { ^ <− #1; } )

        conversationalModel
                MindBoardStructure: {
                        (Mind / Task)
                }

                SensoSchemata
                        (SensoScheme SenseCommand : [Command]
                                SensoAction {
                                        BatchInsertEnd($MINDBOARD@Mind.Commands, #OBJSENSO−1);
                                }
                        )

                RespondoSchemata
                        (RespondoScheme DS: [Command]
                                RespondoCapture {
                                        (BatchSize($MINDBOARD@Mind.Commands) > 0)
                                }
                                RespondoAction {
                                        Command command;
                                        BatchExtractInit($MINDBOARD@Mind.Commands,command);
                                        ^OBJRESPONDO <− command;
                                        RespondoRetry();
                                }
                        )

        functionModel
                string monthName (integer month) {
                        switch (month) {
                                case 1 return ’enero’;
                                case 2 return ’febrero’;
                                case 3 return ’marzo’;
                                case 4 return ’abril’;
                                case 5 return ’mayo’;
                                case 6 return ’junio’;
                                case 7 return ’julio’;
                                case 8 return ’agosto’;
                                case 9 return ’septiembre’;
                                case 10 return ’octubre’;
                                case 11 return ’noviembre’;
                                case 12 return ’diciembre’;
                        }
                }

                string weekdayName (integer day) {
                        switch (day) {
                                case 1 return ’lunes’;
                                case 2 return ’martes’;
                                case 3 return ’miercoles’;
                                case 4 return ’jueves’;
                                case 5 return ’viernes’;
                                case 6 return ’sabado’;
                                case 7 return ’domingo’;
                        }
                }

                string timeExact (integer hour, integer minute, integer second) {
                        boolean am;
                        string time;

                        if (hour > 12) {
                                am <− False;
                                hour −= 12;
                        } else {
                                am <− True;
                        }

                        if (hour == 1) {
                                time <− ’Es la 1 ’;
                        } else {
                                time <− ’Son las ’;
                                time <− LiteralConcat(time,ShapeToLiteral(hour));
                                time <− LiteralConcat(time,’ ’);
                        }

                        if (am) {
                                time <− LiteralConcat(time,’ de la maÃ±ana, ’);
                        } else {
                                if (hour < 10)
                                        time <− LiteralConcat(time,’ de la tarde, ’);
                                else
                                        time <− LiteralConcat(time,’ de la noche, ’);
                        }

                        time <− LiteralConcat(time,ShapeToLiteral(minute));
                        time <− LiteralConcat(time,’ minutos y ’);
                        time <− LiteralConcat(time,ShapeToLiteral(second));
                        time <− LiteralConcat(time,’ segundos.’);

                        return time;
                }


        scriboModel forLanguage es
                ScriboSchemata
                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’day’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Es el dÃ-a ");
                                        SetMainAnswer(ClockAskDayOfTheMonth());
                                        SetMainAnswer(". ");
                                }
                        )

                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’hour’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Son las ");
                                        SetMainAnswer(ClockAskHour());
                                        SetMainAnswer(":");
                                        SetMainAnswer(ClockAskMinute());
                                        SetMainAnswer(". ");
                                }
                        )

                        (ScriboScheme Command: [Command]
                                ScriboCapture {
                                        #OBJLOCUTIO−1 == (askInfo:’date’)
                                }
                                ScriboAction {
                                        SetMainAnswer("Hoy es el ");
                                        SetMainAnswer(ClockAskDayOfTheMonth());
                                        SetMainAnswer(" ");
                                        SetMainAnswer(monthName(ClockAskMonth()));
                                        SetMainAnswer(" de ");
                                        SetMainAnswer(ClockAskYear());
                                        SetMainAnswer(". Es ");
                                        SetMainAnswer(weekdayName(ClockAskDayOfTheWeek()));
                                        SetMainAnswer(". ");
                                        SetMainAnswer(timeExact(
                                                        ClockAskHour(),
                                                        ClockAskMinute(),
                                                        ClockAskSecond()));
                                }
                        )

```

Date_03.slk

```
LaunchLektaKernel()

UseProject (ProjectCompile        : ".Date_03.olk")

DisplayProcessAbstractOn

CreateDialogue()
```

Devices_01.lkt

```
/****************************************************************************
 * SOURCEFILE => Devices_01.lkt
 ****************************************************************************/

lektaProject

   projectHead
                projectDescriptor       : "Devices"
                projectLanguageScope    : [ es ]
                projectOperationScope   : [ test ]
                projectInterfaceScope   : [ dialogue ]
                projectCompileOutput    : ".Devices_01.olk"

   projectSetup
      setupParserRoots = Device, Operation, Command
      setupRuntimeSpyMessageOn

   classModel
      #Include "Devices_01_FM.lkt"

   lexicalModel forLanguage es
      #Include "Devices_01_NLU_L.lkt"

   grammaticalModel forLanguage es
      #Include "Devices_01_NLU_G.lkt"

   functionModel
      #Include "Devices_01_FUN_Decl.lkt"

   conversationalModel
      MindBoardStructure: {
         (Input  / InputModel)
         (Control / Control)
         (Output / OutputModel)
      }

         LaunchConversation: {
            LaunchSendToMindBoard:
               (Control = (Position: ’start’) )
         }

      SensoSchemata
         #Include "Devices_01_DM_Senso.lkt"

      CogitoSchemata
         #Include "Devices_01_DM_Cogito.lkt"

      RespondoSchemata
         #Include "Devices_01_DM_Respondo.lkt"


   functionModel
      #Include "Devices_01_FUN_Spec.lkt"

   scriboModel forLanguage es
      ScriboSchemata
         #Include "Devices_01_NLG.lkt"
```

Devices_01_FM.lkt

```
/***************************************************************************
 * SOURCEFILE => Devices_01_FM.lkt
 ***************************************************************************/

      // AUXILIAR
      classDef:ElementInt       (integer)
      classDef:ElementLiteral   (string)

      // INPUT
      classDef:ElementRange     (DeviceId:
                        { ’alarm’, ’heating’, ’irrigation’ } )

      classDef:ElementRange     (OperationId:
                        { ’activate’, ’deactivate’ } )

      classDef:ElementRange     (CommandId:
                        { ’help’, ’back’, ’exit’ } )

      classDef:StructureComplex (Device: (DeviceId) )
      classDef:StructureComplex (Operation: (OperationId) )
      classDef:StructureComplex (Command: (CommandId) )

      // OUTPUT
      classDef:ElementRange     (AnswerCommand:
                        { ’interactivate’, ’mensaje’, ’outcome’ })

      classDef:ElementRange     (AnswerType:
                        { ’welcome’, ’goodbye’,
                          ’askDevice’, ’askHelp’,
                          ’askOperationAlarm’, ’askOperationHeating’, ’askOperationIrrigation’,
                          ’informAlarmOn’, ’informAlarmOff’, ’helpAlarm’,
                          ’informHeatingOn’, ’informHeatingOff’, ’helpHeating’,
                          ’informIrrigationOn’, ’informIrrigationOff’, ’helpIrrigation’ })

      classDef:StructureComplex         (OutputItem: (AnswerCommand, AnswerType) )

      classDef:StructureBatch   (OutputBatch: (OutputItem) )

      classDef:StructureComplex         (OutputModel: (OutputBatch) )

      // MEMORY
      classDef:StructureComplex (InputModel: (Device, Operation, Command) )

      classDef:ElementRange     (Position: { ’start’, ’quit’,
                        ’askDevice’, ’askHelp’,
                        ’askOperationAlarm’, ’askOperationHeating’, ’askOperationIrrigation’ } )

      classDef:StructureComplex (Control: (Position) )
```

jueves julio 18, 2013

Devices_01_NLU_G.lkt

```
/***************************************************************************
 * SOURCEFILE => Devices_01_NLU_G.lkt
 ***************************************************************************/

      (R1: [ Device -> DeviceId ]
         { ^.DeviceId <- #1; } )

      (R2: [ Operation -> OperationId ]
         { ^.OperationId <- #1; } )

      (R3: [ Command -> CommandId ]
         { ^.CommandId <- #1; } )
```

jueves julio 18, 2013

Devices_01_NLU_L.lkt

```
/****************************************************************************
 * SOURCEFILE => Devices_01_NLU_L.lkt
 ****************************************************************************/

      ("alarma",       DeviceId,   ’alarm’)
      ("calefaccion",  DeviceId,   ’heating’)
      ("riego",        DeviceId,   ’irrigation’)

      ("activar",      OperationId,   ’activate’)
      ("desactivar",   OperationId,   ’deactivate’)

      ("ayuda",        CommandId,   ’help’)
      ("volver",       CommandId,   ’back’)
      ("adios",        CommandId,   ’exit’)
```

Devices_01_DM_Senso.lkt

```
/***************************************************************************
 * SOURCEFILE => Devices_01_DM_Senso.lkt
 ***************************************************************************/

         (SensoScheme RSetup : [Control]
            SensoCapture {
               #OBJSENSO-1.Position == ’start’
            }
            SensoAction {
               $MINDBOARD@Control <- #OBJSENSO-1;
            }
         )

         (SensoScheme RDis : [Device]
            SensoAction {
               $MINDBOARD@Input.Device <- #OBJSENSO-1;
            }
         )

         (SensoScheme RFun : [Operation]
            SensoAction {
               $MINDBOARD@Input.Operation <- #OBJSENSO-1;
            }
         )

         (SensoScheme RGen : [Command]
            SensoAction {
               $MINDBOARD@Input.Command <- #OBJSENSO-1;
            }
         )
```

```
/***************************************************************************
 * SOURCEFILE => Devices_01_DM_Cogito.lkt
 ***************************************************************************/

        (CogitoScheme CogitoStart :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’start’
           }
           CogitoAction {
              OutputItem outputitem <− (AnswerCommand:’interactivate’,AnswerType:’welcome’);
              BatchInsertEnd($MINDBOARD@Output.OutputBatch,outputitem);
              $MINDBOARD@Control.Position <− ’askDevice’;
           }
        )

        (CogitoScheme CogitoQuit :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’quit’
           }
           CogitoAction {
              Quit();
           }
        )

        (CogitoScheme CogitoAskDevice :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’askDevice’
           }
           CogitoAction {
              if ($MINDBOARD@Input.Device) {
                 switch ($MINDBOARD@Input.Device.DeviceId) {
                    case ’alarm’ {
                       $MINDBOARD@Control.Position <− ’askOperationAlarm’;
                    }
                    case ’heating’ {
                       $MINDBOARD@Control.Position <− ’askOperationHeating’;
                    }
                    case ’irrigation’ {
                       $MINDBOARD@Control.Position <− ’askOperationIrrigation’;
                    }
                 }
                 Erase($MINDBOARD@Input.Device);
              } else {
                 if ($MINDBOARD@Input.Command) {
                    switch ($MINDBOARD@Input.Command.CommandId) {
                       case ’help’ {
                          $MINDBOARD@Control.Position <− ’askHelp’;
                       }
                       case ’exit’ {
                          $MINDBOARD@Control.Position <− ’quit’;
                          Quit();
                       }
                    }
                    Erase($MINDBOARD@Input.Command);
                 } else {
                    Menu();
                 }
              }
           }
        )

        (CogitoScheme CogitoAskHelp :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’askHelp’
           }
           CogitoAction {
              OutputItem outputitem <− (AnswerCommand:’mensaje’,AnswerType:’askHelp’);
              BatchInsertEnd($MINDBOARD@Output.OutputBatch,outputitem);
              $MINDBOARD@Control.Position <− ’askDevice’;
           }
        )

        (CogitoScheme CogitoAskOperationAlarm :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’askOperationAlarm’
           }
           CogitoAction {
              if ($MINDBOARD@Input.Operation) {
                 switch ($MINDBOARD@Input.Operation.OperationId) {
                    case ’activate’ {
                       AlarmActivate();
                       $MINDBOARD@Control.Position <− ’askDevice’;
                       Menu();
                    }
                    case ’deactivate’ {
                       AlarmDeactivate();
                       $MINDBOARD@Control.Position <− ’askDevice’;
                       Menu();
                    }
                 }
                 Erase($MINDBOARD@Input.Operation);
              } else {
                 if ($MINDBOARD@Input.Command) {
                    switch ($MINDBOARD@Input.Command.CommandId) {
                       case ’help’ {
                          AlarmHelp();
                       }
                       case ’back’ {
                          $MINDBOARD@Control.Position <− ’askDevice’;
                          Menu();
                       }
                       case ’exit’ {
                          $MINDBOARD@Control.Position <− ’quit’;
                          Quit();
                       }
                    }
                    Erase($MINDBOARD@Input.Command);
                 } else {
                    OutputItem outputitem <− (AnswerCommand:’mensaje’,
                                              AnswerType:’askOperationAlarm’);
                    BatchInsertEnd($MINDBOARD@Output.OutputBatch,outputitem);
                 }
              }
           }
        )

        (CogitoScheme CogitoAskOperationHeating :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’askOperationHeating’
           }
           CogitoAction {
              if ($MINDBOARD@Input.Operation) {
                 switch ($MINDBOARD@Input.Operation.OperationId) {
                    case ’activate’ {
                       HeatingActivate();
                       $MINDBOARD@Control.Position <− ’askDevice’;
                       Menu();
                    }
                    case ’deactivate’ {
                       HeatingDeactivate();
                       $MINDBOARD@Control.Position <− ’askDevice’;
                       Menu();
                    }
                 }
                 Erase($MINDBOARD@Input.Operation);
              } else {
                 if ($MINDBOARD@Input.Command) {
                    switch ($MINDBOARD@Input.Command.CommandId) {
                       case ’help’ {
                          HeatingHelp();
                       }
                       case ’back’ {
                          $MINDBOARD@Control.Position <− ’askDevice’;
                          Menu();
                       }
                       case ’exit’ {
                          $MINDBOARD@Control.Position <− ’quit’;
                          Quit();
                       }
                    }
                    Erase($MINDBOARD@Input.Command);
                 } else {
                    OutputItem outputitem <− (AnswerCommand:’mensaje’,
                                              AnswerType:’askOperationHeating’);
                    BatchInsertEnd($MINDBOARD@Output.OutputBatch,outputitem);
                 }
              }
           }
        )

        (CogitoScheme CogitoAskOperationIrrigation :
           CogitoCapture {
              $MINDBOARD@Control.Position == ’askOperationIrrigation’
           }
           CogitoAction {
              if ($MINDBOARD@Input.Operation) {
                 switch ($MINDBOARD@Input.Operation.OperationId) {
                    case ’activate’ {
                       IrrigationActivate();
                       $MINDBOARD@Control.Position <− ’askDevice’;
                       Menu();
                    }
                    case ’deactivate’ {
                       IrrigationDeactivate();
                       $MINDBOARD@Control.Position <− ’askDevice’;
                       Menu();
                    }
                 }
                 Erase($MINDBOARD@Input.Operation);
              } else {
                 if ($MINDBOARD@Input.Command) {
                    switch ($MINDBOARD@Input.Command.CommandId) {
                       case ’help’ {
                          IrrigationHelp();
                       }
                       case ’back’ {
                          $MINDBOARD@Control.Position <− ’askDevice’;
                          Menu();
                       }
                       case ’exit’ {
                          $MINDBOARD@Control.Position <− ’quit’;
                          Quit();
                       }
                    }
                    Erase($MINDBOARD@Input.Command);
                 } else {
                    OutputItem outputitem <− (AnswerCommand:’mensaje’,
                                              AnswerType:’askOperationIrrigation’);
                    BatchInsertEnd($MINDBOARD@Output.OutputBatch,outputitem);
                 }
              }
           }
        )

```

Devices_01_DM_Respondo.lkt

```
/***************************************************************************
 * SOURCEFILE => Devices_01_DM_Respondo.lkt
 ***************************************************************************/

         (RespondoScheme DS: [OutputItem]
            RespondoCapture {
                 (BatchSize($MINDBOARD@Output.OutputBatch) > 0)
            }
            RespondoAction {
                 OutputItem outputitem;
                 BatchExtractInit($MINDBOARD@Output.OutputBatch,outputitem);
                 ^OBJRESPONDO <− outputitem;
                 RespondoRetry();
             }
         )
```

```
/***************************************************************************
 * SOURCEFILE => Devices_01_NLG.lkt
 ***************************************************************************/

         (ScriboScheme GenerateWelcome: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’interactivate’,AnswerType:’welcome’)
             }
             ScriboAction {
                 SetMainAnswer("Bienvenido al sistema Devices. ");
             }
         )

         (ScriboScheme GenerateAskDevice: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’mensaje’,AnswerType:’askDevice’)
             }
             ScriboAction {
                 SetMainAnswer("Indique que dispositivo desea controlar o bien diga ayuda. ");
             }
         )

         (ScriboScheme GenerateAskHelp: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’mensaje’,AnswerType:’askHelp’)
             }
             ScriboAction {
                 SetMainAnswer("Desde este sistema puede controlar la alarma, la calefaccion ");
                 SetMainAnswer("y el riego. ");
             }
         )

         (ScriboScheme GenerateGoodbye: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’interactivate’,AnswerType:’goodbye’)
             }
             ScriboAction {
                 SetMainAnswer("Adios. Muchas gracias por usar Devices. ");
             }
         )

         (ScriboScheme GenerateOperationAlarm: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’mensaje’,AnswerType:’askOperationAlarm’)
             }
             ScriboAction {
                 SetMainAnswer("Indique si desea activar o desactivar la alarma. ");
             }
         )

         (ScriboScheme GenerateHelpAlarm: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’mensaje’,AnswerType:’helpAlarm’)
             }
             ScriboAction {
                 SetMainAnswer("Usted puede activar o desactivar la alarma, pedir ayuda ");
                 SetMainAnswer("o volver al menu principal. ");
             }
         )

         (ScriboScheme GenerateAlarmActivated: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’outcome’,AnswerType:’informAlarmOn’)
             }
             ScriboAction {
                 SetMainAnswer("La alarma ha sido activada. ");
             }
         )

         (ScriboScheme GenerateAlarmDeactivated: [OutputItem]
             ScriboCapture {
                 #OBJLOCUTIO−1 =:= (AnswerCommand:’outcome’,AnswerType:’informAlarmOff’)
             }
             ScriboAction {
                 SetMainAnswer("La alarma ha sido desactivada. ");
             }
         )
```

Devices_01_NLG.lkt

```

        (ScriboScheme GenerationOperationHeating: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’mensaje’,AnswerType:’askOperationHeating’)
            }
            ScriboAction {
                SetMainAnswer("Indique si desea activar o desactivar la calefaccion. ");
            }
        )

        (ScriboScheme GenerateHelpHeating: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’mensaje’,AnswerType:’helpHeating’)
            }
            ScriboAction {
                SetMainAnswer("Usted puede activar o desactivar la calefaccion, pedir ayuda ");
                SetMainAnswer("o volver al menu principal. ");
            }
        )

        (ScriboScheme GenerateHeatingActivated: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’outcome’,AnswerType:’informHeatingOn’)
            }
            ScriboAction {
                SetMainAnswer("La calefaccion ha sido activada. ");
            }
        )

        (ScriboScheme GenerateHeatingDeactivated: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’outcome’,AnswerType:’informHeatingOff’)
            }
            ScriboAction {
                SetMainAnswer("La calefaccion ha sido desactivada. ");
            }
        )

        (ScriboScheme GenerateOperationIrrigation: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’mensaje’,AnswerType:’askOperationIrrigation’)
            }
            ScriboAction {
                SetMainAnswer("Indique si desea activar o desactivar el riego. ");
            }
        )

        (ScriboScheme GenerateHelpIrrigation: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’mensaje’,AnswerType:’helpIrrigation’)
            }
            ScriboAction {
                SetMainAnswer("Usted puede activar o desactivar el riego, pedir ayuda ");
                SetMainAnswer("o volver al menu principal. ");
            }
        )

        (ScriboScheme GenerateIrrigationActivated: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’outcome’,AnswerType:’informIrrigationOn’)
            }
            ScriboAction {
                SetMainAnswer("El riego ha sido activado. ");
            }
        )

        (ScriboScheme GenerateIrrigationDeactivated: [OutputItem]
            ScriboCapture {
                #OBJLOCUTIO-1 =:= (AnswerCommand:’outcome’,AnswerType:’informIrrigationOff’)
            }
            ScriboAction {
                SetMainAnswer("El riego ha sido desactivado. ");
            }
        )

```

Devices_01_FUN_Decl.lkt

```
/***************************************************************************
 * SOURCEFILE => Devices_01_FUN_Decl.lkt
 ***************************************************************************/

      integer Menu();
      integer Quit();

      integer AlarmActivate();
      integer AlarmDeactivate();
      integer AlarmHelp();

      integer HeatingActivate();
      integer HeatingDeactivate();
      integer HeatingHelp();

      integer IrrigationActivate();
      integer IrrigationDeactivate();
      integer IrrigationHelp();
```

```
/***************************************************************************
 * SOURCEFILE => Devices_01_FUN_Spec.lkt
 ***************************************************************************/

      integer Menu() {
         OutputItem ol <- (AnswerCommand:’mensaje’,AnswerType:’askDevice’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
      }

      integer Quit() {
         OutputItem ol <- (AnswerCommand:’interactivate’,AnswerType:’goodbye’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         CloseDialogue();
         return 1;
      }

      integer AlarmActivate() {
         OutputItem ol <- (AnswerCommand:’outcome’,AnswerType:’informAlarmOn’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer AlarmDeactivate() {
         OutputItem ol <- (AnswerCommand:’outcome’,AnswerType:’informAlarmOff’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer AlarmHelp() {
         OutputItem ol <- (AnswerCommand:’mensaje’,AnswerType:’helpAlarm’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer HeatingActivate() {
         OutputItem ol <- (AnswerCommand:’outcome’,AnswerType:’informHeatingOn’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer HeatingDeactivate() {
         OutputItem ol <- (AnswerCommand:’outcome’,AnswerType:’informHeatingOff’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer HeatingHelp() {
         OutputItem ol <- (AnswerCommand:’mensaje’,AnswerType:’helpHeating’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer IrrigationActivate() {
         OutputItem ol <- (AnswerCommand:’outcome’,AnswerType:’informIrrigationOn’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer IrrigationDeactivate() {
         OutputItem ol <- (AnswerCommand:’outcome’,AnswerType:’informIrrigationOff’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

      integer IrrigationHelp() {
         OutputItem ol <- (AnswerCommand:’mensaje’,AnswerType:’helpIrrigation’);
         BatchInsertEnd($MINDBOARD@Output.OutputBatch,ol);
         return 1;
      }

```

```
/***************************************************************************
 * SOURCEFILE => Calculator_02.lkt
 ***************************************************************************/

lektaProject

   projectHead
        projectDescriptor       : "Calculator"
        projectLanguageScope    : [ es ]
        projectOperationScope   : [ test ]
        projectInterfaceScope   : [ dialogue ]
        projectCompileOutput    : ".Calculator.olk"

   projectSetup
      setupParserRoots = Expresion

   classModel
      classDef:Void ( suma, resta, multiplica, divide, elevado )
      classDef:Void ( abre_parentesis, cierra_parentesis )
      classDef:ElementInt ( integer )

      classDef:ElementInt ( Valor, ValorActual )
      classDef:ElementRange ( Operador: { ’+’, ’-’, ’*’, ’/’, ’^’ } )
      classDef:StructureComplex ( Expresion, Expresion1, Expresion2, ExpresionEntrada, ExpresionCalculo, ExpresionSalida:
            (Valor, Operador, Expresion1, Expresion2) )

      classDef:StructureComplex ( Memoria: ( ExpresionEntrada, ExpresionCalculo ) )

   lexicalModel forLanguage es
      setupTokenizerIntegerClass   (integer,".,")
      setupTokenizerPunctuation   ("+", suma)
      setupTokenizerPunctuation   ("-", resta)
      setupTokenizerPunctuation   ("*", multiplica)
      setupTokenizerPunctuation   ("/", divide)
      setupTokenizerPunctuation   ("^", elevado)
      setupTokenizerPunctuation   ("(", abre_parentesis)
      setupTokenizerPunctuation   (")", cierra_parentesis)

      ("cero",   integer,      0)
      ("uno",    integer,      1)
      ("dos",    integer,      2)
      ("tres",   integer,      3)
      ("cuatro", integer,      4)
      ("cinco",  integer,      5)
      ("seis",   integer,      6)
      ("siete",  integer,      7)
      ("ocho",   integer,      8)
      ("nueve",  integer,      9)

      ("mas",     suma)
      ("menos",   resta)
      ("entre",   divide)
      ("por",     multiplica)
      ("elevado a",   elevado)

      ("abre",             abre_parentesis)
      ("abre parentesis",  abre_parentesis)
      ("cierra",           cierra_parentesis)
      ("cierra parentesis",cierra_parentesis)


   grammaticalModel forLanguage es

      (R1: [ Expresion/0 -> integer ]
         { ^.Valor <- #1; } )

      (R2: [ Expresion/0 -> &[-Expresion] resta integer ]
         { ^.Valor <- - #2; } )

      (R3: [ Expresion/0 -> abre_parentesis Expresion/10 cierra_parentesis ]
         { ^ <- #2; } )

      (R4: [ Expresion/3 -> Expresion/3 suma Expresion/2 ]
         { ^.Operador <- ’+’;
           ^.Valor <- #1.Valor + #3.Valor;
           ^.Expresion1 <- #1;
           ^.Expresion2 <- #3; } )

      (R5: [ Expresion/3 -> Expresion/3 resta Expresion/2 ]
         { ^.Operador <- ’-’;
           ^.Valor <- #1.Valor - #3.Valor;
           ^.Expresion1 <- #1;
           ^.Expresion2 <- #3; } )

      (R6: [ Expresion/2 -> Expresion/2 multiplica Expresion/1 ]
         { ^.Operador <- ’*’;
           ^.Valor <- #1.Valor * #3.Valor;
           ^.Expresion1 <- #1;
           ^.Expresion2 <- #3; } )

      (R7: [ Expresion/2 -> Expresion/2 divide Expresion/1 ]
         { ^.Operador <- ’/’;
           ^.Valor <- #1.Valor / #3.Valor;
           ^.Expresion1 <- #1;
           ^.Expresion2 <- #3; } )

      (R8: [ Expresion/1 -> Expresion/1 elevado Expresion/0 ]
         { ^.Operador <- ’^’;
           ^.Valor <- Pow(#1.Valor,#3.Valor);
           ^.Expresion1 <- #1;
           ^.Expresion2 <- #3; } )

      (R9: [ Expresion/20 -> suma Expresion/10 ]
         { ^.Operador <- ’+’;
           ^.Expresion2 <- #2; } )

      (R10: [ Expresion/20 -> resta Expresion/10 ]
         { ^.Operador <- ’-’;
           ^.Expresion2 <- #2; } )

      (R11: [ Expresion/20 -> multiplica Expresion/10 ]
         { ^.Operador <- ’*’;
           ^.Expresion2 <- #2; } )

      (R12: [ Expresion/20 -> divide Expresion/10 ]
         { ^.Operador <- ’/’;
           ^.Expresion2 <- #2; } )

      (R13: [ Expresion/20 -> elevado Expresion/10 ]
         { ^.Operador <- ’^’;
           ^.Expresion2 <- #2; } )


   conversationalModel
      MindBoardStructure: {
         (Memoria / Memoria)
      }

      SensoSchemata
         (SensoScheme RExpresion : [Expresion]
            SensoAction {
               $MINDBOARD@Memoria.ExpresionEntrada <- #OBJSENSO-1;
            }
         )

      CogitoSchemata
         (CogitoScheme TExpresion :
            CogitoAction {
               if ($MINDBOARD@Memoria.ExpresionEntrada.Operador) {
                  if ($MINDBOARD@Memoria.ExpresionEntrada.Expresion1) {
                     $MINDBOARD@Memoria.ExpresionCalculo <-
                        $MINDBOARD@Memoria.ExpresionEntrada;
                  } else {
                     Expresion fexp <- $MINDBOARD@Memoria.ExpresionEntrada;
                     fexp.Expresion1 <- $MINDBOARD@Memoria.ExpresionCalculo;
                     $MINDBOARD@Memoria.ExpresionCalculo <- fexp;
                     switch ($MINDBOARD@Memoria.ExpresionCalculo.Operador) {
                        case ’+’ {
                           $MINDBOARD@Memoria.ExpresionCalculo.Valor <-
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion1.Valor +
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion2.Valor;
                        }
                        case ’-’ {
                           $MINDBOARD@Memoria.ExpresionCalculo.Valor <-
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion1.Valor -
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion2.Valor;
                        }
                        case ’*’ {
                           $MINDBOARD@Memoria.ExpresionCalculo.Valor <-
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion1.Valor *
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion2.Valor;
                        }
                        case ’/’ {
                           $MINDBOARD@Memoria.ExpresionCalculo.Valor <-
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion1.Valor /
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion2.Valor;
                        }
                        case ’^’ {
                           $MINDBOARD@Memoria.ExpresionCalculo.Valor <-
                              Pow($MINDBOARD@Memoria.ExpresionCalculo.Expresion1.Valor,
                              $MINDBOARD@Memoria.ExpresionCalculo.Expresion2.Valor);
                        }
                     }
                  }
               } else {
                  $MINDBOARD@Memoria.ExpresionCalculo <- $MINDBOARD@Memoria.ExpresionEntrada;
               }
            }
         )

      RespondoSchemata
         (RespondoScheme DExpresion: [ExpresionSalida]
            RespondoAction {
               ^OBJRESPONDO <- $MINDBOARD@Memoria.ExpresionCalculo;
            }
         )


   scriboModel forLanguage es
      ScriboSchemata
         (ScriboScheme GExpresion: [ExpresionSalida]
            ScriboAction {
               SetMainAnswer(#OBJLOCUTIO-1.Valor);
            }
         )

```

jueves julio 18, 2013

Calculator_02.slk

Page 1/1

```
/****************************************************************************
 * SOURCEFILE => Calculator_02.lkt
 ****************************************************************************/

LaunchLektaKernel()

UseProject
        (ProjectCompile        : ".Calculator.olk")

CreateDialogue()
```