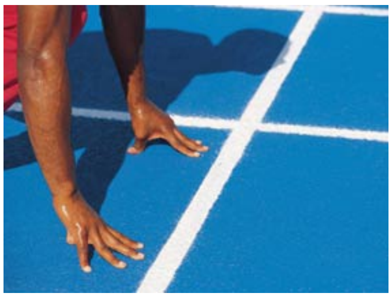
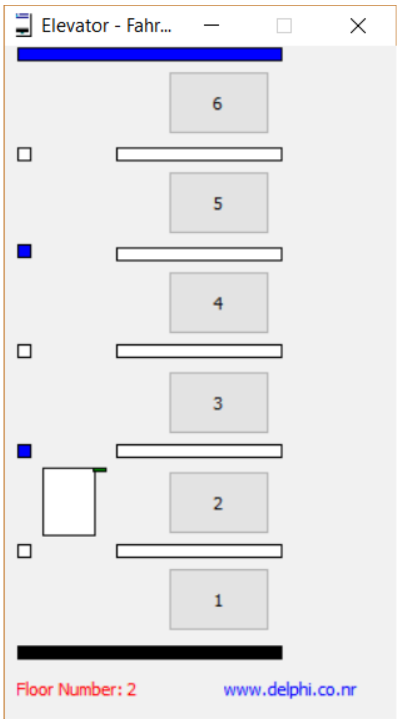
maXbox Starter 53



Real Time UML

1.1 Elevator Example

First we have to clarify the relationship between the object paradigm and real time systems.
Overwhelming complexity is by far the biggest hurdle in most real time software systems. On the other side the structure of real time system tends to persist through time because it reflects the physical entities of the real world. So here's my definition[1] of Real Time UML:
<u>Compute events in the real order of their occurrence.</u>
In complex RT systems, the logical design is strongly influenced by the characteristics of the physical engineering environment. And such an environment can be an elevator system.



[1] the actual time during which something takes place

In a Use case diagram we capture the functional requirement of the system and it’s interaction between the actor the system.

Each elevator has a set of <n> buttons, one for each floor and position. These illuminate when pressed a panel and cause the elevator to visit the corresponding floor. Waiting people is indicating by a blue light inside cabin. The illumination is cancelled when the elevator visits the right floor. The system has 2 main sensors and 2 actors:

Sensors:

- Door Push
- Button Press

Actors:

- Floor Indicator
- Speaker Information

This is implemented by the following 3 code blocks:

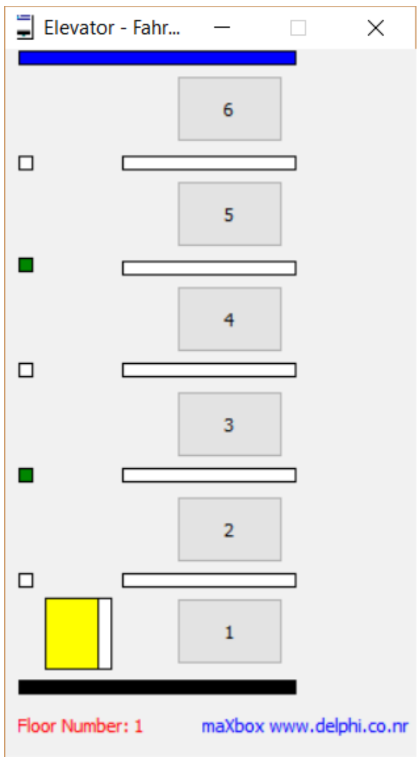
```
procedure openDoor;
var open: integer;
begin
  for open:= 1 to {frm_lift.}shp_lift.height-3 do begin
    shp_door.height:= shp_door.height-1;  //-1
    shp_door.brush.color:= clGreen;
    sleep(30); //acts as a delay
    shp_door.Update;
  end;
 oldcolor.brush.color:= clyellow;
end;

procedure Tfrm_liftupTimer(Sender: TObject);
begin
  shp_lift.top:= shp_lift.top-4;
  shp_door.top:= shp_door.Top-4;
  if shp_lift.top = tmr_liftup.tag then begin
     tmr_liftup.enabled:= false;
     openDoor;
     buttononoff(true);
     liftfloorState(tmr_liftup.tag, lbl2)
  end;
end;
```

Actors:

- Floor Indicator, Light and Speaker Panel

```
begin lbl2.caption:= '3';
   voice2('third floor')
   ChangelightColor(shape4);
 end;
```

By “Real-Time UML”, I mean the application of the UML standard to the development of real-time and embedded systems, and focusing attention on those aspects of UML especially relevant to the areas of special concern for such physical systems like heat or speed.



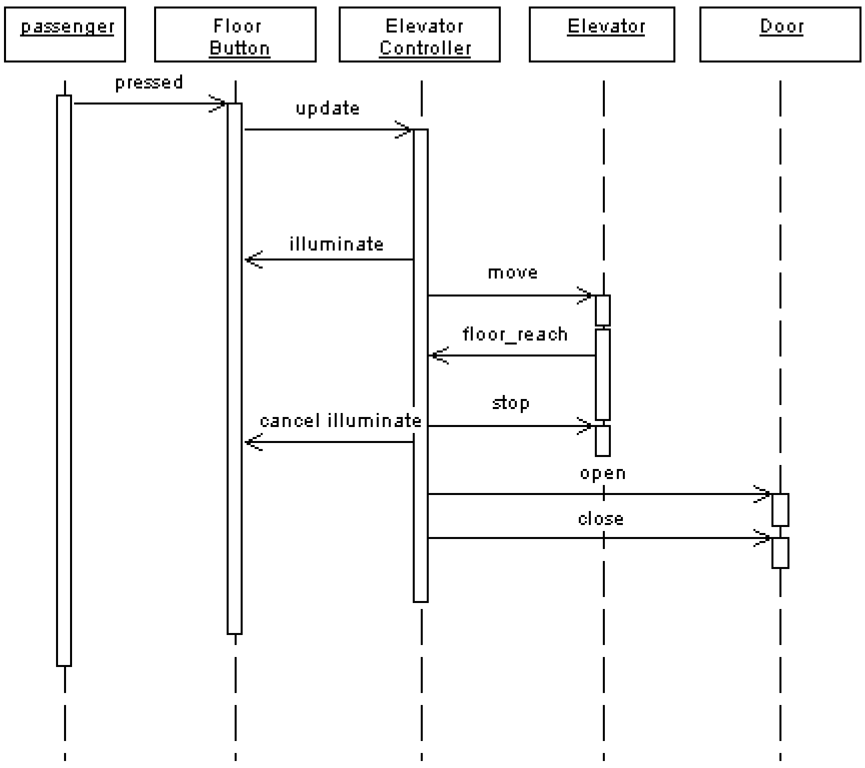
The system may ask an elevator to go up, go down or stop by a panel. The system may ask the elevator to open its door. The system will receive a notification when the door is closed. This simulates the activity of letting people on and off at each floor. The door closes automatically after a predefined amount of time. The touch panel above also shows position (up and down) and the waiting people of the elevator at real time!

In order to ensure safety, emergency brakes will be triggered and the car will be forced to stop under any unsafe conditions. When an elevator has no request, it remains at its current floor with its door closed.
During the run a red signal shows precaution and a yellow draws your attention to hear the voice.

```
procedure Movelift (Dest : Integer);
begin
  if {frm_lift.}shp_lift.Top = dest then begin
    shp_lift.brush.color:= clyellow;
    exit;
  end;
  buttononoff(false);
  if shp_lift.Top > Dest
    then begin
        tmr_Liftup.Tag:= Dest;
```

```
         tmr_liftup.Enabled:= true;
         shp_lift.brush.color:= clBlack;
       end
     else
      if shp_Lift.Top < Dest then begin
         tmr_LiftDown.Tag:= Dest;
         tmr_LiftDown.Enabled:= True;
         shp_lift.brush.color:= clBlack;
      end;
   shp_lift.brush.color:= clWhite;
   oldcolor.brush.color:= clred;
end;
```

The traditional assignment approaches for elevator group systems, which are simple and fairly rough, are only fit for low-rise buildings. Hence, they are not suitable in the high-rise buildings with complex traffic requests. This is where real-time scheduling in an operating system for elevator group systems are of interest!



According to Kopetz, a real time computer system is a computer system in which the correctness of the system behaviour depends not only on the logical results of the computations, but also on the physical instant at which these results are produced[2].
KONE for example has now made elevator services truly intelligent. By connecting our elevators to the cloud, by listening closely, by analysing

[2] Hermann Kopetz. Real-Time Systems, UML Design Principles for Distributed Embedded Applications.

their messages, we can tailor the almost perfect maintenance to each individual elevator. So that you can tune into real conversations between machines in a syntax called `ASCIITalk`:

http://machineconversations.kone.com/

The Elevator device generate events for over 22 different set of parameters. In the scope of this demo, let us consider at least to monitor only the `Motor Temperature` in a service view.
When the Motor Temperature value exceeds the 200 degree Celsius mark, a registered `Node-RED` Action is triggered, which in turn, invokes the `WIoTP` to `SEND` the Stop command to the Elevator for example with WMI:

```
isQuery:=
      'SELECT * FROM Win32_Service WHERE State = "Running"';
iset:= WMIExecQuery(isser, isQuery); //Wbem ObjectSet;
```

Then you get a list of your services running! Even more so, this is the recommended standard Microsoft way to do operational things and it is ever more so when it comes to Wine, Win 7 and 10.
WMI (most cases I've seen) is arranged in the form of object tables, whose data (fields or properties) are accessed in a subset of SQL that Microsoft calls WQL. So you don't need a DB or a server to execute those queries.

Code and script can be found at:
http://www.softwareschule.ch/examples/751_Elevator_Simulator3.pas

http://www.softwareschule.ch/examples/766_wmi_management.txt

Of course there's a great number of them, and you can find out just about anything regarding your local system or any system on the network that you can connect to like the running hardware statistic, along with memory usages, processes, and threads or a physical element like temp:

```
 FWbemObjectSet:= FWMIService.Get('Win32_TemperatureProbe');
```

Beware, that selecting all the data in some of the tables will take a very large time and a huge amount of resources, so be careful to limit your statements as you would in dealing with a regular database.
You do also have more than one root name space available, even the query language in the object set is chose able:

```
Const
  WbemComputer       ='localhost'; //wbemFlagForwardOnly= $00000020;
  WbemRootNameSpace ='root\CIMV2';

 FWbemObjectSet:= FWMIService.ExecQuery(Format('Select %s from %s',
                    [WMIProperty, WMIClass]),'WQL',wbemFlagForwardOnly);
```

Each item in a `SWbemObjectSet` is a `SWbemObject` (yet another object (Interface) in the WMI scripting library) that represents a single instance of the requested resource. You use `SWbemObject` and `WMIRowFindNext` to access the methods and properties defined in the managed resource's class definition `WMIClass`.

The WMI architecture consists of three primary layers as shown in the picture above from MS (`ms974579.aspx`):

- Managed resources
- WMI infrastructure
- Consumers and producers

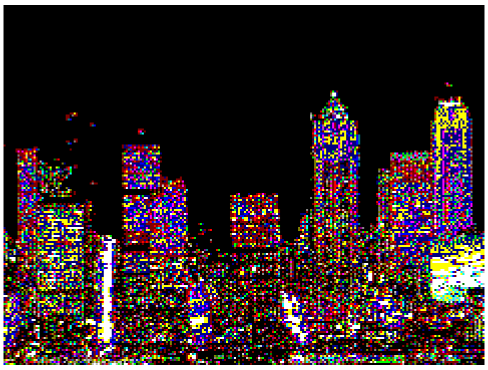
Let's do a second example in the same script, we want to list all running processes like a task manager does:

```
isQuery:= 'SELECT * FROM Win32_Process';
iset:= WMIExecQuery(isser, isQuery); //WbemObjectSet;
writeln(botoStr(WMIRowFindFirst(iset, ENum, tempObj)));
it:= 0;
  repeat
    PrintF('Processes run: %s - PID %s',
                 [tempobj.name,vartoStr(tempobj.processid)])
    inc(it)
  until not WMIRowFindNext(ENum, tempObj);
writeln('running Processes : '+itoa(it))
```

So, if the scripting steps to retrieve information from WMI are identical, what are the `Win32_Process`, `Win32_Service`, and `Win32_NTLogEvent` classes to connect to an elevator?
At least there are two ways to install and configure your box tool into a directory you want. The first way is to use the unzip command-line tool or IDE, which is discussed above.
That means no installation needed. Another way is to copy all the files to navigate to a folder you like, and then simply drag and drop another scripts into the /examples directory.

https://maxbox.codeplex.com/

Conclusion:
The increasing complexity of embedded and real-time systems requires a more premeditated and sophisticated design approach for successful implementation. The object-based Unified Modelling Language (UML) can describe the structural and behavioural aspects critical to real-time systems, and has come to the fore as an outstanding medium for effective design.
Real-time data and analytics's help us to accurately predict and collect equipment needs, and ensure that our technicians perform maintenance at the right time.



“Wise speak: Better late than wait.

Feedback @ max@kleiner.com
Literature: Kleiner et al., Patterns konkret, 2003, Software & Support

https://github.com/maxkleiner/maXbox4/releases

https://www.academia.edu/31112544/Work_with_microservice_maXbox_starter48.pdf

Real Time Tutor:

http://www.softwareschule.ch/download/maxbox_starter23.pdf

UML Tutor:

http://www.softwareschule.ch/download/maxbox_starter29.pdf

1.2 References

Examples and Version of this Tutorial 53:
2017-07 Cumulative Update for Windows 10 Version 1703 for x64-based Systems (KB4025342) – maXbox 4.2.5.10

www.softwareschule.ch/examples/751_Elevator_Simulator3.pas

http://www.softwareschule.ch/examples/766_wmi_management.txt

B. Selic and J. Rumbaugh: “Using UML for Modeling Complex Real Time Systems,”
Time Systems,” ObjecTime Limited and Rational Software Corp., March 1998.
(http://www.rational.com)

Real-Time UML: Developing Efficient Objects for Embedded Systems, 2nd Edition
By Bruce Powel Douglass
Published Oct 27, 1999 by Addison-Wesley Professional. Part of the Addison-Wesley Object Technology Series series.
http://www.informit.com/store/real-time-uml-developing-efficient-objects-for-embedded-9780201657845

Examples of WMI Routines:

WMI Scripting Primer:

https://msdn.microsoft.com/en-us/library/ms974579.aspx

http://www.softwareschule.ch/examples/750_RSA_Toolproof4.txt

WMI and Database SQL Programming Starter 12 Tutorial and Tutorial 52

http://www.softwareschule.ch/download/maxbox_starter12.pdf

```
             _od#HMM6&*MMMH::-_
          _dHMMMR??MMM? ""| `"'-?Hb_
       .~HMMMMMMMMHMMM#M?        `*HMb.
     ./?HMMMMMMMMMMM"*"""           &MHb.
    /'|MMMMMMMMMMM'             -   `*MHM\
   /  |MMMMMMHHM''                   .MMMHb
  |   9HMMP   .Hq,                   TMMMMMH
 /     |MM\,H-""&&6\__               `MMMMMMb
|       `""HH#,       \             - MMMMMMM|
|           `HoodHMM###.              `9MMMMMH
|              .MMMMMMMM##\             `*"?HM
|         ..  ,HMMMMMMMMMMMo\.              |M
|             |M'MMMMMMMM'MMMMMHo           |M
|              ?MMM'MAXBOX M'MMM*           |H
|.              `#MMMM'MM'MMMMM'           .M|
 \                `MMMMMMMMMMM*            |P
 `\                MMMMMMMMT"'            ,H
  `\              `MMMMMMH?              ./
    \.            |MMMH#"               ,/
     `\.          |MMP'               ./'
       `~\        `HM:.-    .       ,/'
          "-\_       '_\ .      _.-"
              "-\-#odMM\_,oo==-"
```

Talk to each other across the globe, in a human voice makes the race.

https://maxbox4.wordpress.com/