

# Programming the SimaPro COM interface

June 2007



## Colophon

**Title :** Programming the SimaPro COM interface

**Written by:** PRé Consultants  
Chris de Gelder

**Report version:** 1.0  
**Date:** June 2007  
**Language:** English  
**Availability:** PDF file

**Copyright:** © 2007 PRé Consultants. All rights reserved.  
PRé Consultants grants the right to print the PDF version of this manual. Parts of the manual may be reproduced only if a clear reference is made that PRé Consultants is the author.  
The manual may not be used for commercial purposes.

**Support:** phone +31 33 4555022  
fax +31 33 4555024  
e-mail [support@pre.nl](mailto:support@pre.nl)  
website [www.pre.nl](http://www.pre.nl)

# Contents

1	INTRODUCTION	1
2	SUPPORTED ENVIRONMENTS AND OPERATING SYSTEMS	1
3	ARCHITECTURE	1
4	TECHNICAL SAMPLES	2
4.1	VBA (EXCEL / WORD)	2
4.2	DELPHI	3
4.3	PHP	4
4.4	C++	4
4.5	.NET SAMPLE	5
5	REFERENCE	7

## 1 Introduction

SimaPro is a versatile LCA tool, but in some occasions you need more than a stand-alone tool. For example, if you want to analyze many data automatically you would like to integrate SimaPro in your business software.

In the SimaPro Developer version a COM-interface is available. This allows the user to control SimaPro from applications such as Excel, .NET applications, Delphi, PHP etc. Practical applications can be:

- Linking with your ERP system
- Analyzing a Bill of Materials from another software tool, such as a CAD/CAM system
- Exporting specific data to Excel
- Create processes from Excel sheets
- Create a website with results from SimaPro
- Create a website where users can enter data into processes in SimaPro.

## 2 Supported environments and operating systems

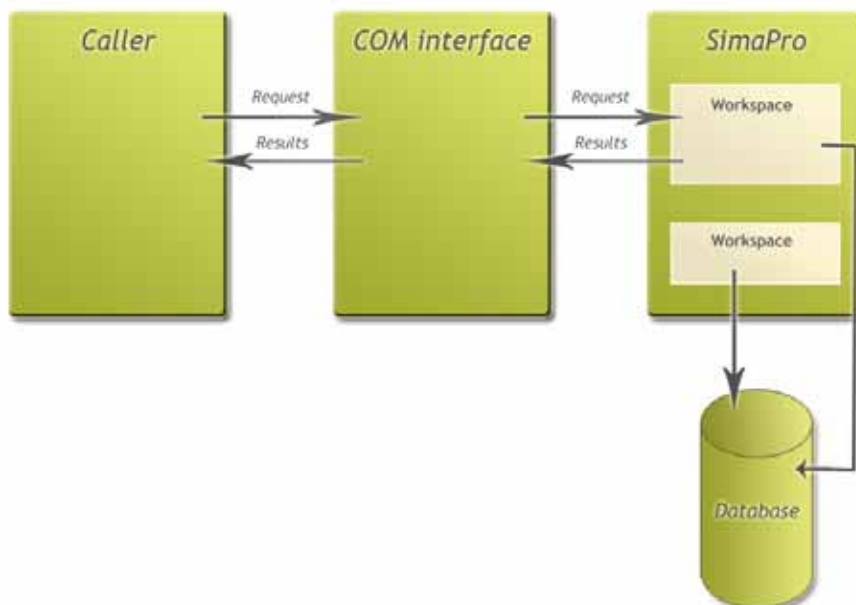
All software that can use the COM interfaces are supported, for example:

- Excel, Word (VBA)
- Delphi
- C++
- Visual Basic
- PHP

Using SimaPro and the COM interface to support a web server requires more on the user rights level. Most easy is to use .NET, but ASP is also possible. COM is typical for Windows so it is only available on Windows Systems.

## 3 Architecture

Every caller has its own workspace within SimaPro. This workspace contains opened database, project and calculation results of the last calculation.



## 4 Technical samples

Below you find some simple examples. Note for all samples: adapt the server name, alias, database and project and process names to your own situation! If you use the COM interface with a single user version, you do not need to enter the server name, for the alias enter the database directory path.

### 4.1 VBA (Excel / Word)

This example creates a substance and 2 processes including parameters. One process gets input from the other.

```
Sub CreateProcess()
    Dim SP As SimaProServer
    Dim PC As Process
    Dim PC2 As Process
    Dim PL As ProcessLine
    Dim Param As ParamLine
    Dim Subs As Substance

    Set SP = New SimaProServer
    SP.Server = "nexusdb@192.168.1.220"
    SP.Alias = "Default"
    SP.Database = "Professional"
    SP.OpenDatabase
    SP.Login "Manager", ""
    SP.OpenProject "Introduction to SimaPro 7", ""

    SP.CreateSubstance "Air", Subs
    Subs.CASNumber = "4-5-13"
    Subs.Name = "Some substance"
    Subs.DefaultUnit = "kg"
    Subs.Update

    SP.CreateProcess ptMaterial, PC
    Set PL = PC.AddLine(ppProducts, -1)
    PL.ObjectName = "Steel 2"
    PL.UnitName = "kg"
    PL.Amount = "2"
    PL.Comment.Add ("My new created process")
    PL.CategoryPath = "Chemicals\inorganic"

    PC.Update

    ' create second material process Case
    SP.CreateProcess ptMaterial, PC2
    Set PL = PC2.AddLine(ppProducts, 0)
    PL.ObjectName = "Case 2"
    PL.UnitName = "kg"
    PL.Amount = "10"

    Set Param = PC2.AddParamLine(ptInputParameter, -1)
    Param.Name = "A"
    Param.Value = "2,3"

    ' add input from Steel
    Set PL = PC2.AddLine(ppMaterialsFuels, -1)
    ' input from steel
    PL.SetProduct "Introduction to SimaPro 7", ptMaterial, "Steel 2"
```

```

PL.Amount = "8"
PL.UnitName = "kg"

Set PL = PC2.AddLine(ppAirborneEmissions, -1)
' input from steel
PL.SetSubstance "Some substance", ""
PL.Amount = "A+1"
PL.UnitName = "kg"

PC2.Update

SP.Logout
SP.CloseDatabase
Set SP = Nothing

End Sub

```

## 4.2 Delphi

### 4.2.1 Calculate a single score

This example calculates the single score of a process, using the Eco-indicator 99 H/A method and returns the result.

```

var
  SimaPro: SimaProServer;
begin
  SimaPro := CoSimaProServer.Create;
  SimaPro.Server := 'nexusdb@192.168.1.220';
  SimaPro.Alias := 'Default';
  SimaPro.Database := 'Professional';
  SimaPro.OpenDatabase;
  SimaPro.Login ('Manager', '');
  SimaPro.OpenProject ('Introduction to SimaPro 7', '');
  if SimaPro.Analyse('BUWAL250', ptMaterial, Electricity Netherlands B250 ',
    'Methods', 'Eco-indicator 99 (H)', 'Europe EI 99 H/A') then
  begin
    memol.lines.add('single score = ');
    memol.lines.add(FloatToStr(SimaPro.AnalyseResult(rtSingleScore, 0).Amount));
  end;
end;

```

### 4.2.2 Create a process

This example creates a process with an input of plastic.

```

var
  SimaPro: SimaProServer;
  PC2: Process;
  PL: ProcessLine;
begin
  SimaPro := CoSimaProServer.Create;
  SimaPro.Server := 'nexusdb@192.168.1.220';
  SimaPro.Alias := 'Default';
  SimaPro.Database := 'Professional';
  SimaPro.OpenDatabase;
  SimaPro.Login ('Manager', '');
  SimaPro.OpenProject ('Introduction to SimaPro 7', '');
  SimaPro.CreateProcess (ptMaterial, PC2);
end;

```

```

PL := PC2.AddLine(ppProducts, 0);
PL.ObjectName := 'Case 2';
PL.UnitName := 'kg';
PL.Amount := '10';
PL := PC2.AddLine(ppMaterialsFuels, -1);
PL.SetProduct ('BUWAL250', ptMaterial, ' PVC B250');
PL.Amount := '8';
PL.UnitName := 'kg';
PC2.Update;
memo1.lines.add('ready');

```

### 4.3 PHP

This example, for PHP in console mode, prints an overview of all processes and product stages in a project to the console.

```

<?php
$SP = new COM("SimaPro.SimaProServer");
$SP->Server = 'nexusdb@192.168.1.111';
$SP->Alias = 'Default';
$SP->Database = 'Professional';
$SP->OpenDatabase;
$SP->Login ('Manager', '');
$SP->OpenProject ('Introduction to SimaPro 7', '');
print $SP->ProductCount + "\n";
for ($I = 1; $I < $SP->ProductCount; $I = $I + 1) {
    print $SP->ProductName($I) . "\n";
}
$sp = null;

?>

```

### 4.4 C++

The C++ is more complex due to the memory allocation requirements. Precondition is a form with a memo called memo1. Otherwise replace the "Memo1->Lines-Add" part with more suitable code.

This example below calculates the single score of a process, using the Eco-indicator 99 H/A method and returns the result.

```

BSTR Server = ::SysAllocString( L"nexusdb@192.168.1.220" );
BSTR Alias = ::SysAllocString( L"Default" );
BSTR Database = ::SysAllocString( L"Professional" );
BSTR User = ::SysAllocString( L"Manager" );
BSTR Project = ::SysAllocString( L"Introduction to SimaPro 7" );
BSTR ProcessProject = ::SysAllocString(L"BUWAL250");
BSTR Process = ::SysAllocString( L"PVC B250" );
BSTR MethodLib = ::SysAllocString( L"Methods" );
BSTR Method = ::SysAllocString( L"Eco-indicator 99 (H)" );
BSTR NWSet = ::SysAllocString( L"Europe EI 99 H/A" );

TCOMISimaProServer SimaPro = CoSimaProServer::Create();
SimaPro->Server = Server;
SimaPro->Alias = Alias;
SimaPro->Database = Database;
SimaPro->OpenDatabase();
SimaPro->Login (User, L"");
SimaPro->OpenProject (Project, L"");
if (SimaPro->Analyse(ProcessProject, ptMaterial, Process,

```

```

        MethodLib, Method, NWSet))
    {
        Memo1->Lines->Add("Single score = ");
        Memo1->Lines->Add(FloatToStr(SimaPro->AnalyseResult(rtSingleScore,
0)->Amount));
    }

    ::SysFreeString(Server);
    ::SysFreeString(Alias);
    ::SysFreeString(Database);
    ::SysFreeString(User);
    ::SysFreeString(Project);
    ::SysFreeString(ProcessProject);
    ::SysFreeString(Process);
    ::SysFreeString(MethodLib);
    ::SysFreeString(Method);
    ::SysFreeString(NWSet);

```

## 4.5 .NET Sample

This example will create webpage with a list of all processes in a database and create a new process with a given name.

Create a website or aspx page in Visual Studio with

1 Label  
2 Buttons  
1 Textbox  
1 GridView

And use the following code:

```

using System;
using System.Data;
using System.Collections;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using SimaPro;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        SimaProServer SP = new SimaPro.SimaProServer();
        SimaPro.Process PC;
        SP.Server = "nexusdb@192.168.1.220";
        SP.Alias = "Default";
        SP.Database = "Professional";
        SP.OpenDatabase();
        SP.Login("Manager", "");
    }
}

```

```

    SP.OpenProject("Introduction to SimaPro 7", "");
    SP.CreateProcess(TProcessType.ptMaterial, out PC);
    ProcessLine PL = PC.AddLine(TProcessPart.ppProducts, -1);
    PL.ObjectName = TextBox1.Text;
    PL.UnitName = "kg";
    PL.Amount = "2";
    PL.Comment.Add("My new created process");
    PC.Update();
    SP.Logout();
    SP.CloseDatabase();
    Label1.Text = TextBox1.Text + " is created";
}
protected void Button2_Click(object sender, EventArgs e)
{
    ArrayList AL = new ArrayList();

    SimaProServer SP = new SimaPro.SimaProServer();
    SimaPro.Process PC;
    SP.Server = "nexusdb@192.168.1.220";
    SP.Alias = "Default";
    SP.Database = "Professional";
    SP.OpenDatabase();
    SP.Login("Manager", "");
    SP.OpenProject("Introduction to SimaPro 7", "");
    for(int I = 0; I < SP.ProductCount; I++)
    {
        AL.Add(SP.get_ProductName(I));
    };
    GridView1.DataSource = AL;
    GridView1.DataBind();
    SP.Logout();
    SP.CloseDatabase();
    Label1.Text = TextBox1.Text + " list done";
}
}

```

## 5 Reference

### Amount Property

Object	Property description
ProcessLine	Amount or percentage of product or substance
SimaProAnalyseResult	Amount of units substance or impact assessment score
SimaProNetworkResult	Amount of the product
SimaProTreeResult	Amount of the product

### Cancel Method

Object	Method description
Process	Cancel edit mode and returns to read mode
Substance	Cancel edit mode and returns to read mode

### Comment Property

Object	Property description
ParamLine	Comment of the parameter
Process	Comment of the process
ProcessLine	Comment
Substance	Comment

### Distribution Property

Object	Property description
ParamLine	Distribution of the input parameter
ProcessLine	Distribution of amount

### Edit Method

Object	Method description
Process	Set the object in edit mode
Substance	Set the object in edit mode

### LineNumber Property

Object	Property description
ParamLine	Index in the section of the process to which the ProcessLine object belongs
ProcessLine	Index in the section of the process to which the ProcessLine object belongs

## MainCompartmentName Property

Object	Property description
--------	----------------------

SimaProAnalyseResult	
SimaProServer	Name of a main-compartment

## Maximum Property

Object	Property description
--------	----------------------

ParamLine	Maximum value of the input parameter
ProcessLine	Maximum value of amount

## Minimum Property

Object	Property description
--------	----------------------

ParamLine	Minium value of the input parameter
ProcessLine	Minimum value of amount

## Mode Property

Object	Property description
--------	----------------------

Process	Mode of the object, can be: Read, New or Edit
Substance	Mode of the object, can be: Read, New or Edit

## Name Property

Object	Property description
--------	----------------------

ParamLine	Name of the parameter
Substance	Name of the substance (e.g. 'Carbon dioxide')

## ParamLine

Class ParamLine

Properties

Represents one parameter in SimaPro

---

---

## ParamLine properties

ParamLine Legend

- Comment
- Distribution
- Expression
- Hide
- ▶ LineNumber
- Maximum
- Minimum
- Name
- ▶ ParameterType
- ▶ Process
- StandardDeviation
- Value

### ParamLine.Comment

Property Comment As IStrings  
Comment of the parameter  
Member of ParamLine

#### Example

Add comment to a parameter

```
ParamLine1.Comment.add('estimated value')
```

---

### ParamLine.Distribution

Property Distribution As TDistribution  
Distribution of the input parameter  
Member of ParamLine

#### Example

Set the distribution to normal

```
Param1.Distribution := distNormal
```

---

### ParamLine.Expression

Property Expression As String  
Expression of the calculated parameter  
Member of ParamLine

#### Example

B is A + 1

```
Param1.Name = 'B'  
param1.ParameterType = ptCalculatedParameter  
Param1.Expression = 'A+1'
```

---

## ParamLine.Hide

Property Hide As Boolean  
Only local visible in process  
Member of ParamLine

---

## ParamLine.LineNumber

Property LineNumber As Long  
Index in the section of the process to which the ProcessLine object belongs  
Member of ParamLine  
Read-Only

---

## ParamLine.Maximum

Property Maximum As Double  
Maximum value of the input parameter  
Member of ParamLine

---

## ParamLine.Minimum

Property Minimum As Double  
Minimum value of the input parameter  
Member of ParamLine

---

## ParamLine.Name

Property Name As String  
Name of the parameter  
Member of ParamLine

**Example**  
B is A + 1

```
Param1.Name = 'B'  
param1.ParameterType = ptCalculatedParameter  
Param1.Expression = 'A+1'
```

## ParamLine.ParameterType

Property ParameterType As TParameterType

Section in the process to which the ProcessLine object belongs

Member of ParamLine

Read-Only

### Example

B is A + 1

```
Param1.Name = 'B'  
param1.ParameterType = ptCalculatedParameter  
Param1.Expression = 'A+1'
```

---

## ParamLine.Process

Property Process As Process

Process to which the ParamLine object belongs

Member of ParamLine

Read-Only

Can be used to trace the process properties.

---

## ParamLine.StandardDeviation

Property StandardDeviation As Double

Standard deviation of the input parameter

Member of ParamLine

---

## ParamLine.Value

Property Value As Double

Value of the input parameter

Member of ParamLine

### Example

Set A to 13

```
if Process.FindParameter('a', Param) then  
    Param.Edit  
else  
    Param = Process.AddParamLine(ptInputParameter, -1)  
    Param.Name = 'a'  
Param.Value = 13  
Param.Update
```

## Process

Class Process

Properties    Methods

## Process methods

Process    Legend

- AddLine
- AddParamLine
- Cancel
- DeleteLine
- DeleteParamLine
- Edit
- FindParameter
- Update

## Process properties

Process    Legend

- Comment
- ▶ Line
- ▶ LineCount
- ▶ Mode
- ▶ ParamLine
- ▶ ParamLineCount
- ▶ ProcessType
- ▶ ProjectName
- Status

## Process.AddLine

Function AddLine(ByVal Part As TProcessPart, ByVal LineNumber As Long) As ProcessLine

Add a line and returns a ProcessLine object

Member of Process

Parameters	Description
<b>Part</b>	Part within process (ppProducts, ppMaterialsFuels etc)
<b>LineNumber</b>	Linenummer (zero based) -1 means: at the end

**Return value**

Returns ProcessLine

**Example**

Add emission and product to a process

```
Line := MyProcess.AddLine(ppEmissionsWater, -1)
Line.SetSubstance('Carbon dioxide', 'Air')
Line.Amount := '34';
Line.UnitName := 'g';
Line := MyProcess.AddLine(ppProduct, -1);
Line.ProductName := 'My product name',
Line.Amount := '1';
Line.UnitName := 'kg';
```

```
Line.WasteType := 'Steel';
```

See also the example at `SimaProServer.CreateProcess`

---

## Process.AddParamLine

Function `AddParamLine(ByVal ParameterType As TParameterType, ByVal LineNumber As Long) As ParamLine`

Add a line and returns a `ParamLine` object

Member of `Process`

Parameters	Description
<b>ParameterType</b>	Type
<b>LineNumber</b>	LineNumber, -1 adds at end

**Return value**

Returns `ParamLine`

### Example

Set A to 13

```
if NOT Process.FindParameter('a', Param) then
begin
    Param = Process.AddParamLine(ptInputParameter, -1);
    Param.Name = 'a' ;
end;
Param.Value = 13
```

---

## Process.Cancel

Sub `Cancel()`

Cancel edit mode and returns to read mode

Member of `Process`

### Example

Undo all edits

```
Process.name := 'New name';
Process.cancel
```

---

## Process.Comment

Property `Comment` As `IStrings`

Comment of the process

Member of `Process`

### Example

Set the comment

```
process.comment.add('changed by Joe');
```

---

## Process.DeleteLine

Sub DeleteLine(ByVal Part As TProcessPart, ByVal LineNumber As Long)

Delete a line

Member of Process

Parameters	Description
<b>Part</b>	Which part of the process
<b>LineNumber</b>	Which line within part (Zero based)

### Example

Delete first product

```
Process.DeleteLine(ppProducts, 0)
```

---

## Process.DeleteParamLine

Sub DeleteParamLine(ByVal ParameterType As TParameterType, ByVal LineNumber As Long)

Delete a parameter line

Member of Process

Parameters	Description
<b>ParameterType</b>	Type of parameter
<b>LineNumber</b>	Linenummer (zero based)

### Example

Delete second calculated parameter

```
Pc.deleteParamLine(ptCalculatedParameter, 1)
```

---

## Process.Edit

Sub Edit()

Set the object in edit mode

Member of Process

### Example

Change the waste type of 'Steel NL' to 'Steel'

```
pc := SimaPro.FindProcess(ptmaterial, 'Steel NL');
pc.Edit;
try
  // change waste type of first product
  pc.processline[ppProduct, 0].WasteType := 'Steel';
  pc.Update;
except
  pc.Cancel;
end;
```

---

## Process.FindParameter

Function FindParameter(ByVal Name As String, ByRef ParamLine As ParamLine) As Boolean

Find a line and returns a ParamLine object

Member of Process

Parameters	Description
<b>Name</b>	Name of Parameter
<b>ParamLine</b>	Result

### Return value

Returns Boolean

### Example

Set A to 13

```
if Process.FindParameter('a', Param) then
    Param.Edit
else
    Param = Process.AddParamLine(ptInputParameter, -1)
    Param.Name = 'a'
Param.Value = 13
Param.Update
```

## Process.Line

Property Line(ByVal Part As TProcessPart, ByVal LineNumber As Long) As ProcessLine

Returns a ProcessLine object

Member of Process

Read-Only

Parameters	Description
<b>Part</b>	Part within process (ppProducts, ppMaterialsFuels etc)
<b>LineNumber</b>	Linenummer (zero based)

### Example

Delete the emissions to air of Carbon Dioxide.

```
Pc := SimaPro.FindProcess(ptMaterial, 'steel');
Pc.Edit;
I := 0;
while I < Pc.ProcessLineCount[ppEmissionsAir] do
begin
    if Pc.Line[ppEmissionsAir, I].ObjectName2 = 'Carbon Dioxide, biogenic' then
        Pc.DeleteLine(ppEmissionsAir, I)
    else
        Inc(I);
end;
Pc.Update;
```

## Process.LineCount

Property LineCount(ByVal Part As TProcessPart) As Long

Number of lines in a part

Member of Process

Read-Only

Parameters	Description
------------	-------------

<b>Part</b>	Products etc
-------------	--------------

```

if SimaPro.FindProcess('Sample project', ptMaterial, 'steel', pc) then
begin
  Pc.Edit;
  I := 0;
  while I < PC.LineCount[ppEmissionsAir] do
  begin
    s := pc.Line[ppEmissionsAir, I].SubName;
    if Pc.Line[ppEmissionsAir, I].SubName = 'Carbon Dioxide, biogenic' then
      Pc.Line[ppEmissionsAir, I].Delete
    else
      Inc(I);
  end;
  Pc.Update;
end;

```

## Process.Mode

Property Mode As String

Mode of the object, can be: Read, New or Edit

Member of Process

Read-Only

## Process.ParamLine

Property ParamLine(ByVal ParameterType As TParameterType, ByVal LineNumber As Long) As ParamLine

Returns a ParamLine object

Member of Process

Read-Only

Parameters	Description
------------	-------------

<b>ParameterType</b>	ptInputParameter,
----------------------	-------------------

<b>LineNumber</b>	Linenummer where you want to insert. -1 inserts at end
-------------------	--

## Process.ParamLineCount

Property ParamLineCount(ByVal ParameterType As TParameterType) As Long

Number of parameters

Member of Process

Read-Only

Parameters	Description
ParameterType	ptInputParameter or ptCalculatedParameter

## Process.ProcessType

Property ProcessType As TProcessType

Type of the process

Member of Process

Read-Only

## Process.ProjectName

Property ProjectName As String

Name of the project to which the process belongs

Member of Process

Read-Only

## Process.Status

Property Status As TProcessStatus

Status of the process

Member of Process

### Example

Set status to draft

```
Pc.Status := stDraft;
```

## Process.Update

Sub Update()

Store the data of the object in the database and switch to read mode

Member of Process

## Example

Change the waste type of 'Steel NL' to 'Steel'

```
pc := SimaPro.FindProcess(ptmaterial, 'Steel NL');
pc.Edit;
try
  // change waste type of first product
  pc.processline[ppProduct, 0].WasteType := 'Steel';
pc.Update;
except
  pc.Cancel;
end;
```

---

## ProcessLine

Class ProcessLine

Properties    Methods

A line in a process as seen on screen in SimaPro. Can be either products, inputs or outputs.

---

## ProcessLine methods

ProcessLine    Legend

- SetMaterial
- SetProduct
- SetSubstance

## ProcessLine properties

ProcessLine    Legend

- Allocation
- Amount
- CategoryPath
- Comment
- Distribution
- ▶ LineNumber
- Maximum
- Minimum
- ObjectName
- ▶ ObjectName2
- ▶ Part
- ▶ Process
- ▶ ProcessType
- ▶ ProjectName
- ▶ ProjectName2
- StandardDeviation
- UnitName
- WasteType

## ProcessLine.Allocation

Property Allocation As String  
Allocation percentage of a product  
Member of ProcessLine

---

## ProcessLine.Amount

Property Amount As String  
Amount or percentage of product or substance  
Member of ProcessLine

### Example

Example of usage goes here

```
Line.SetProduct('eco invent unit processes', 'Electricity UCTPE');  
Line.Amount := '10';  
Line.UnitName := 'kWh';
```

See also the example at `SimaProServer.CreateProcess`

---

## ProcessLine.CategoryPath

Property CategoryPath As String  
Category path of a product  
Member of ProcessLine

### Example

Change the category

```
AProcessLine.Category := '\others\chemicals';
```

---

## ProcessLine.Comment

Property Comment As IStrings  
Comment  
Member of ProcessLine

---

## ProcessLine.Distribution

Property Distribution As TDistribution  
Distribution of amount  
Member of ProcessLine

## ProcessLine.LineNumber

Property LineNumber As Long

Index in the section of the process to which the ProcessLine object belongs

Member of ProcessLine

Read-Only

---

## ProcessLine.Maximum

Property Maximum As Double

Maximum value of amount

Member of ProcessLine

---

## ProcessLine.Minimum

Property Minimum As Double

Minimum value of amount

Member of ProcessLine

---

## ProcessLine.ObjectName

Property ObjectName As String

Name of the product

Member of ProcessLine

See the example at `SimaProServer.CreateProcess`

---

## ProcessLine.ObjectName2

Property ObjectName2 As String

Sub-compartment or material name

Member of ProcessLine

Read-Only

## ProcessLine.Part

Property Part As TProcessPart  
Section in the process to which the ProcessLine object belongs  
Member of ProcessLine  
Read-Only

---

## ProcessLine.Process

Property Process As Process  
Process to which the ProcessLine object belongs  
Member of ProcessLine  
Read-Only

---

## ProcessLine.ProcessType

Property ProcessType As TProcessType  
Type of the product  
Member of ProcessLine  
Read-Only

---

## ProcessLine.ProjectName

Property ProjectName As String  
Name of the project to which the product belongs  
Member of ProcessLine  
Read-Only

---

## ProcessLine.ProjectName2

Property ProjectName2 As String  
Name of the project to which the material belongs  
Member of ProcessLine  
Read-Only

## ProcessLine.SetMaterial

Sub SetMaterial(ByVal ProjectName As String, ByVal ProductName As String)

Select a material and link it to the process; only for waste treatment product and specific waste flow  
Member of ProcessLine

Parameters	Description
<b>ProjectName</b>	Name of project (see SimaproServer.Projects)
<b>ProductName</b>	Name of product Only for advanced disposal modelling. Use SetProduct to link inputs from technosphere.

### Example

```
PC.SetMaterial('My project', 'Steel');
```

---

## ProcessLine.SetProduct

Sub SetProduct(ByVal ProjectName As String, ByVal ProcessType As TProcessType, ByVal ProductName As String)

Select a product of process as an input and link it to the process  
Member of ProcessLine

Parameters	Description
<b>ProjectName</b>	Name of project
<b>ProcessType</b>	
<b>ProductName</b>	Name of the product

Link inputs from technosphere.

### Example

Example of usage goes here

```
Line.SetProduct('eco invent unit processes', 'Electricity UCTPE');
Line.Amount := '10';
Line.UnitName := 'kWh';
```

---

## ProcessLine.SetSubstance

Sub SetSubstance(ByVal SubstanceName As String, ByVal SubCompartmentName As String)

Select a substance and link it to the process  
Member of ProcessLine

Parameters	Description
<b>SubstanceName</b>	Name as in the substance list
<b>SubCompartmentName</b>	Name of the subcompartment Use this to define the emissions and raw material use of a process

### Example

```
Line.SetSubstance('Carbon dioxide', 'Air')
Line.Amount := '11.4';
Line.UnitName := 'kg';
Pc.Update;
```

---

---

## ProcessLine.StandardDeviation

Property StandardDeviation As Double  
 Standard deviation of amount  
 Member of ProcessLine

---

## ProcessLine.UnitName

Property UnitName As String  
 Name of the unit of amount  
 Member of ProcessLine

### Example

Example of usage goes here

```
Line.SetProduct('eco invent unit processes', 'Electricity UCTPE');
Line.Amount := '10';
Line.UnitName := 'kWh';
```

See also the example at `SimaProServer.CreateProcess`

---

## ProcessLine.WasteType

Property WasteType As String  
 Waste type of a product or specific waste flow  
 Member of ProcessLine

---

## Process Property

Select one of the available subtopics below to see detailed help on **Process** property

Object	Property description
ParamLine	Process to which the ParamLine object belongs
ProcessLine	Process to which the ProcessLine object belongs

## ProcessType Property

Select one of the available subtopics below to see detailed help on **ProcessType** property

Object	Property description
Process	Type of the process
ProcessLine	Type of the product

## ProductName Property

Select one of the available subtopics below to see detailed help on **ProductName** property

Object	Property description
SimaProNetworkResult	Name of the product
SimaProServer	Name of a product
SimaProTreeResult	Name of the product

## ProjectName Property

Select one of the available subtopics below to see detailed help on **ProjectName** property

Object	Property description
Process	Name of the project to which the process belongs
ProcessLine	Name of the project to which the product belongs

## SimaProAnalyseResult

Class **SimaProAnalyseResult**

Properties

**SimaProAnalyseResult** object

---

## SimaProAnalyseResult properties

**SimaProAnalyseResult** Legend

- Amount
- IndicatorName
- MainCompartmentName
- SubCompartmentName
- UnitName

## SimaProAnalyseResult.Amount

Property **Amount** As Double

Amount of units substance or impact assessment score

Member of **SimaProAnalyseResult**

**Example**

See **SimaProServer.Analyse**

---

## SimaProAnalyseResult.IndicatorName

Property **IndicatorName** As String

Name of substance, impact category etc. In case of single score it is empty

Member of **SimaProAnalyseResult**

**Example**

See **SimaProServer.Analyse**

## SimaProAnalyseResult.MainCompartmentName

Property MainCompartmentName As String  
Only for inventory results  
Member of SimaProAnalyseResult

---

## SimaProAnalyseResult.SubCompartmentName

Property SubCompartmentName As String  
Only for inventory results  
Member of SimaProAnalyseResult

---

## SimaProAnalyseResult.UnitName

Property UnitName As String  
Unit (e.g. kg, m3)  
Member of SimaProAnalyseResult

### Example

See SimaProServer.Analyse

---

## SimaProCalculationError

Class SimaProCalculationError  
Properties  
SimaProCalculationError object

---

## SimaProCalculationError properties

SimaProCalculationError    Legend  
    AdditionalInfo  
    ErrorCode  
    ErrorDescription

## SimaProCalculationError.AdditionalInfo

Property AdditionalInfo As String

Member of SimaProCalculationError

---

## SimaProCalculationError.ErrorCode

Property ErrorCode As Long

Number of the error

Member of SimaProCalculationError

---

## SimaProCalculationError.ErrorDescription

Property ErrorDescription As String

Description of the error

Member of SimaProCalculationError

---

## SimaProNetworkResult

Class SimaProNetworkResult

Properties

SimaProNetworkResult object

---

## SimaProNetworkResult properties

SimaProNetworkResult    Legend

Amount

ChildProductName

ProductName

UnitName

## SimaProNetworkResult.Amount

Property Amount As Double

Amount of the product

Member of SimaProNetworkResult

## SimaProNetworkResult.ChildProductName

Property ChildProductName As String  
Name of the child-product  
Member of SimaProNetworkResult

---

## SimaProNetworkResult.ProductName

Property ProductName As String  
Name of the product  
Member of SimaProNetworkResult

---

## SimaProNetworkResult.UnitName

Property UnitName As String  
Unit of the amount  
Member of SimaProNetworkResult

---

## SimaProServer

Class SimaProServer  
Properties    Methods  
SimaProServer Object

---

## SimaProServer methods

SimaProServer    Legend  
Analyse  
AnalyseResult  
CalculationError  
CloseDatabase  
CloseProject  
CreateProcess  
CreateSubstance  
FindProcess  
FindSubstance  
Login

Logout  
 Network  
 NetworkCalcScore  
 NetworkResult  
 OpenDatabase  
 OpenProject  
 Tree  
 TreeCalcScore  
 TreeResult

## SimaProServer properties

SimaProServer Legend

Alias  
 ▶ Aliases  
 ▶ CalculationErrorCount  
 ▶ CurrentProject  
 ▶ CurrentUser  
 Database  
 ▶ DatabaseOpen  
 ▶ Databases  
 ▶ LoggedIn  
 ▶ MainCompartmentCount  
 ▶ MainCompartmentName  
 ▶ MethodCount  
 ▶ MethodName  
 ▶ MethodProjectName  
 ▶ NetworkChildNodeCount  
 ▶ NetworkChildNodeIndex  
 ▶ NetworkNodeCount  
 ▶ NetworkProductName  
 ▶ NetworkTopNodeIndex  
 ▶ NWSets  
 ▶ ProductCount  
 ▶ ProductName  
 ▶ ProductProcessType  
 ▶ ProductProcessTypeName  
 ▶ ProductProjectName  
 ▶ ProjectOpen  
 ▶ Projects  
 ▶ QuantityCount  
 ▶ QuantityName  
 ▶ ResultCount  
 ▶ ResultIndicatorName  
 ▶ ResultMainCompartmentName  
 ▶ ResultSubCompartmentName  
 Server  
 ▶ Servers  
 ▶ SubCompartmentCount  
 ▶ SubCompartmentName  
 ▶ SubstanceCASNumber  
 ▶ SubstanceCount  
 ▶ SubstanceDefaultUnit  
 ▶ SubstanceName  
 ▶ TreeChildNodeCount  
 ▶ TreeChildNodeIndex  
 ▶ TreeNodeCount  
 ▶ TreeProductName  
 ▶ TreeTopNodeIndex  
 ▶ UnitCount

- ▶ UnitDefault
- ▶ UnitFactor
- ▶ UnitMetric
- ▶ UnitName
- ▶ WasteTypeCount
- ▶ WasteTypeName

## SimaProServer.Alias

Property Alias As String

Currently used alias

Member of SimaProServer

See the example at SimaProServer.CreateProcess

---

## SimaProServer.Aliases

Property Aliases As IStrings

List of available aliases, set Server first

Member of SimaProServer

Read-Only

---

## SimaProServer.Analyse

Function Analyse(ByVal ProjectName As String, ByVal ProcessType As TProcessType, ByVal ProductName As String, ByVal MethodProjectName As String, ByVal MethodName As String, ByVal NWSetName As String) As Boolean

Perform the analyse function for a process or product stage

Member of SimaProServer

Parameters	Description
<b>ProjectName</b>	Name of project
<b>ProcessType</b>	Type of process
<b>ProductName</b>	Name of Product
<b>MethodProjectName</b>	Project name where method resides
<b>MethodName</b>	name of method
<b>NWSetName</b>	Normalisation Weighting set.

**Return value**

Returns Boolean

**Example**

Show the inventory

```
SP.Analyse('My project', ptMaterial, 'Steel', 'Methods', 'EI99', 'N/A');
// show inventory
for I := 0 to SP.ResultCount(rtInventory) - 1 do
begin
    Res := SP.AnalyseResult(rtInventory, I)
    Print Res.Amount, Res.IndicatorName, Res.UnitName;
end;
```

---

## SimaProServer.AnalyseResult

Function AnalyseResult(ByVal AnalyseResultType As TResultType, ByVal I As Long) As SimaProAnalyseResult

Retrieve the result of the analyse function

Member of SimaProServer

Parameters	Description
<b>AnalyseResultType</b>	rtCharacterisation, rtDamage, rtNormalisation, rtWeighting, rtSingleScore or rtInventory
<b>I</b>	Index

Return value

Returns SimaProAnalyseResult

Example

See SimaProServer.Analyse

---

## SimaProServer.CalculationError

Function CalculationError(ByVal I As Long) As SimaProCalculationError

Calculation error data

Member of SimaProServer

Parameters	Description
<b>I</b>	Index

Return value

Returns SimaProCalculationError

---

## SimaProServer.CalculationErrorCount

Property CalculationErrorCount As Long

Number of calculation errors

Member of SimaProServer

Read-Only

---

## SimaProServer.CloseDatabase

Sub CloseDatabase()

Close the currently open database

Member of SimaProServer

## SimaProServer.CloseProject

Sub CloseProject()

Close the currently open project

Member of SimaProServer

## SimaProServer.CreateProcess

Sub CreateProcess(ByVal ProcessType As TProcessType, ByRef Process As Process)

Creates a new process

Member of SimaProServer

Parameters	Description
<b>ProcessType</b>	ProcessType (ptMaterial, ptEnergy, etc)
<b>Process</b>	Resulting process object

### Example

Create 2 processes and link to each other (VB)

```

Dim SP As SimaProServer
Dim PC As Process
Dim PC2 As Process
Dim PL As ProcessLine

Set SP = New SimaProServer
SP.Server = "nexusdb@192.168.2.113"
SP.Alias = "Default"
SP.Database = "Professional"
SP.OpenDatabase
SP.Login "Manager", ""
SP.OpenProject "A COM DEMO", ""

SP.CreateProcess ptMaterial, PC
Set PL = PC.AddLine(ppProducts, -1)
PL.ObjectName = "Steel"
PL.UnitName = "kg"
PL.Amount = "2"
PL.Comment.Add ("My new created process")
PC.Update

' create second material process Case
SP.CreateProcess ptMaterial, PC2
Set PL = PC2.AddLine(ppProducts, 0)
PL.ObjectName = "Case"
PL.UnitName = "kg"
PL.Amount = "10"

' add input from Steel
Set PL = PC2.AddLine(ppMaterialsFuels, -1)
' input from steel

```

```

PL.SetProduct "A COM DEMO", ptMaterial, "Steel"
PL.Amount = "8"
PL.UnitName = "kg"
PC2.Update

SP.Logout
SP.CloseDatabase
Set SP = Nothing

```

---

## SimaProServer.CreateSubstance

Sub CreateSubstance(ByVal MainCompartment As String, ByRef Substance As Substance)

Create a new substance

Member of SimaProServer

Parameters	Description
<b>MainCompartment</b>	MainCompartment goes here ('Air', 'Water', 'Soil', etc)
<b>Substance</b>	Resulting substance object

### Example

Create a new substance

```

SimaPro.CreateSubstance('Air', Substance)
Substance.Name := 'My new substance'
Substance.UnitName := 'kg'
Substance.Update; // save in database

```

---

## SimaProServer.CurrentProject

Property CurrentProject As String

Name of the currently open project

Member of SimaProServer

Read-Only

---

## SimaProServer.CurrentUser

Property CurrentUser As String

Name of the user that is currently logged in

Member of SimaProServer

Read-Only

---

## SimaProServer.Database

Property Database As String  
 Currently used database, see OpenDatabase  
 Member of SimaProServer

See the example at SimaProServer.CreateProcess

---

## SimaProServer.DatabaseOpen

Property DatabaseOpen As Boolean  
 Indicates if a database is currently open  
 Member of SimaProServer  
 Read-Only

---

## SimaProServer.Databases

Property Databases As IStrings  
 List of available databases, set Server and Alias first  
 Member of SimaProServer  
 Read-Only

---

## SimaProServer.FindProcess

Function FindProcess(ByVal ProjectName As String, ByVal ProcessType As TProcessType, ByVal ProductName As String, ByRef Process As Process) As Boolean  
 Looks for a process in the project and libraries  
 Member of SimaProServer

Parameters	Description
<b>ProjectName</b>	Name of project
<b>ProcessType</b>	Type of process
<b>ProductName</b>	Name
<b>Process</b>	Result object

**Return value**  
 Returns Boolean

### Example

Remove lines from process

```
if SimaPro.FindProcess('Sample project', ptMaterial, 'steel', pc) then
begin
  Pc.Edit;
  I := 0;
  while I < PC.LineCount[ppEmissionsAir] do
```

```

begin
  s := pc.Line[ppEmissionsAir, I].SubName;
  if Pc.Line[ppEmissionsAir, I].SubName = 'Carbon Dioxide, biogenic' then
    Pc.Line[ppEmissionsAir, I].Delete
  else
    Inc(I);
  end;
  Pc.Update;
end;

```

---

## SimaProServer.FindSubstance

Function FindSubstance(ByVal MainCompartmentName As String, ByVal SubstanceName As String, ByRef Substance As Substance) As Boolean

Find a substance in the database

Member of SimaProServer

Parameters	Description
<b>MainCompartmentName</b>	'Water', 'Air' etc
<b>SubstanceName</b>	Required substancename
<b>Substance</b>	Substance object

### Return value

Returns Boolean

### Example

Read CO2 CAS Number

```

PC.FindSubstance('Air', 'Carbon dioxide', Substance);
Print Substance.CASNumber;

```

You can also use

```

PC.SubstanceCasNumber('Air', 'Carbon dioxide')

```

---

## SimaProServer.LoggedIn

Property LoggedIn As Boolean

Indicates if a user is currently logged in

Member of SimaProServer

Read-Only

---

## SimaProServer.Login

Function Login(ByVal UserName As String, ByVal Password As String) As Boolean

Log in the database, not needed for single user if manager-password = empty

Member of SimaProServer

Parameters	Description
<b>UserName</b>	Name of user
<b>Password</b>	Password

### Return value

Returns Boolean

See the example at SimaProServer.CreateProcess

---

## SimaProServer.Login

Function Login() As Boolean  
 Log in to the database  
 Member of SimaProServer

**Return value**  
 Returns Boolean

See also the example at SimaProServer.CreateProcess

---

## SimaProServer.MainCompartmentCount

Property MainCompartmentCount As Long  
 Number of main-compartments  
 Member of SimaProServer  
 Read-Only

### Example

List the maincompartments

```
for I := 0 to SP.MainCompartmentCount - 1 do
  print SP.MainCompartmentName(i)
```

---

## SimaProServer.MainCompartmentName

Property MainCompartmentName(ByVal I As Long) As String  
 Name of a main-compartment  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

---

I	Index
---	-------

### Example

See SimaProServer.MainCompartmentCount

---

## SimaProServer.MethodCount

Property MethodCount As Long  
 Number of impact assessment methods in the currently open project and selected libraries  
 Member of SimaProServer  
 Read-Only

---

## SimaProServer.MethodName

Property MethodName(ByVal I As Long) As String  
 Name of an impact assessment method  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

I	Index Used for listing methods
---	-----------------------------------

## SimaProServer.MethodProjectName

Property MethodProjectName(ByVal I As Long) As String  
 Name of the project of an impact assessment method  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

I	Index
---	-------

## SimaProServer.Network

Function Network(ByVal ProjectName As String, ByVal ProcessType As TProcessType, ByVal ProductName As String, ByVal MethodProjectName As String, ByVal MethodName As String, ByVal NWSetName As String) As Boolean  
 Perform the network function for a process or product stage  
 Member of SimaProServer

Parameters	Description
------------	-------------

<b>ProjectName</b>	Name of project
<b>ProcessType</b>	ProcessType (ptMaterial, ptEnergy, etc)
<b>ProductName</b>	Name of product
<b>MethodProjectName</b>	Project where methods are stored (often 'Methods')
<b>MethodName</b>	name of the Method
<b>NWSetName</b>	Name of normalisation weighting set

**Return value**  
 Returns Boolean

### Example

Calculate a network and fetch the results of the top node

```
if SP.Network('My project', 'ptEnergy', 'Electricity', 'Methods', 'ei99', 'N/A') then
begin
    SP.NetworkCalcScore(rtIndicator, '', '', '');
    for I := 0 to SimaPro.NetworkChildNodeCount[SimaPro.NetWorkTopNodeIndex] - 1 do
    begin
        Res := SP.NetworkResult(nrProductAmount, SimaPro.NetWorkTopNodeIndex, I);
        print res.ProductName, res.Amount, res.UnitName;
    end;
end;
```

## SimaProServer.NetworkCalcScore

Function NetworkCalcScore(ByVal ResultType As TResultType, ByVal Param1 As String, ByVal Param2 As String, ByVal Param3 As String) As Boolean

Calculates the node and flow scores of a network.

Member of **SimaProServer**

Parameters	Description		
<b>ResultType</b>	<b>Param1</b>	<b>Param2</b>	<b>Param3</b>
rtCharacterisation	Impact Category	-	-
rtDamage	Damage Category	-	-
rtNormalisation	Damage Category or Impact Category	-	-
rtWeighting	Damage Category or Impact Category	-	-
rtSingleScore	-	-	-
rtInventory	MainCompartment	Subcompartment	SubstanceName

You can perform multiple NetworkCalcScores calculations on an existing Network.

### Return value

Returns Boolean

### Example

See SimaProServer.Network

## SimaProServer.NetworkChildNodeCount

Property NetworkChildNodeCount(ByVal NodeIndex As Long) As Long

Number of child nodes of a network node

Member of **SimaProServer**

Read-Only

Parameters	Description
<b>NodeIndex</b>	Index of the node

All nodes are indexed. This function return the number of children of a certain node.

### Example

See SimaProServer.Network

## SimaProServer.NetworkChildNodeIndex

Property NetworkChildNodeIndex(ByVal NodeIndex As Long, ByVal FlowIndex As Long) As Long

Index of a child node of a network node

Member of **SimaProServer**

Read-Only

Parameters	Description
<b>NodeIndex</b>	Node
<b>FlowIndex</b>	Flow of that node

Points to a node, for example get the productname with NetworkProductName

---

## SimaProServer.NetworkNodeCount

Property NetworkNodeCount As Long  
 Number of nodes in the network  
 Member of SimaProServer  
 Read-Only

---

## SimaProServer.NetworkProductName

Property NetworkProductName(ByVal NodeIndex As Long) As String  
 Product name of a network node  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

NodeIndex	Node
-----------	------

---

## SimaProServer.NetworkResult

Function NetworkResult(ByVal NodeResultType As TNodeResultType, ByVal NodeIndex As Long, ByVal FlowIndex As Long) As SimaProNetworkResult  
 Retrieve the data of a network node  
 Member of SimaProServer

Parameters	Description
------------	-------------

NodeResultType	nrProductAmount, nrIndicatorContribution, nrIndicatorTotal, nrFlowIndicator
NodeIndex	Node index see NetworkNodeCount
FlowIndex	Flow index (per node) See NetworkChildNodeCount

**Return value**  
 Returns SimaProNetworkResult

**Example**  
 See SimaProServer.Network

---

## SimaProServer.NetworkTopNodeIndex

Property NetworkTopNodeIndex As Long  
 Index of the top node of the network  
 Member of SimaProServer  
 Read-Only

**Example**  
 See SimaProServer.Network

## SimaProServer.NWSets

Property NWSets(ByVal ProjectName As String, ByVal MethodName As String) As IStrings

List of normalisation-weighting sets in a method

Member of SimaProServer

Read-Only

Parameters	Description
<b>ProjectName</b>	Name of Project
<b>MethodName</b>	Name of Method

### Example

Show the first NWSet

```
Print SP.NWSets('methods', 'ecoindicator 99')[0]
```

## SimaProServer.OpenDatabase

Sub OpenDatabase()

Open a database

Member of SimaProServer

Set Server, Alias and Database first

### Example

```
SP.Server := 'local server';
SP.Alias := 'C:\DATA'
SP.Database := 'Professional';
SP.OpenDatabase;
```

See also the example at SimaProServer.CreateProcess

## SimaProServer.OpenProject

Sub OpenProject(ByVal ProjectName As String, ByVal Password as String)

Open a project

Member of SimaProServer

Parameters	Description
<b>ProjectName</b>	Project
<b>Password</b>	Only needed if the project is protected

### Example

Open a project

```
SP.OpenProject('Introduction into LCA');
```

## SimaProServer.ProductCount

Property ProductCount As Long

Number of processes and product-stages in the currently open project and selected libraries

Member of SimaProServer

Read-Only

## SimaProServer.ProductName

Property ProductName(ByVal I As Long) As String

Name of a product

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

I	Index
Used for listing data	

## SimaProServer.ProductProcessType

Property ProductProcessType(ByVal I As Long) As TProcessType

Process type of a product

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

I	Index
---	-------

## SimaProServer.ProductProcessTypeName

Property ProductProcessTypeName(ByVal I As Long) As String

Description of the process type of a product

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

I	Index
---	-------

## SimaProServer.ProductProjectName

Property ProductProjectName(ByVal I As Long) As String  
 Name of the project of a product  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

I	Index
---	-------

## SimaProServer.ProjectOpen

Property ProjectOpen As Boolean  
 Indicates if a project is currently open  
 Member of SimaProServer  
 Read-Only

## SimaProServer.Projects

Property Projects As IStrings  
 List of available projects, open database and log in first  
 Member of SimaProServer  
 Read-Only

## SimaProServer.QuantityCount

Property QuantityCount As Long  
 Number of quantities  
 Member of SimaProServer  
 Read-Only

## SimaProServer.QuantityName

Property QuantityName(ByVal I As Long) As String  
 Name of a quantity  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

I	Index
---	-------

Listing the quantities in the database.

---

## SimaProServer.ResultCount

Property ResultCount(ByVal ResultType As TResultType) As Long

Number of indicators

Member of SimaProServer

Read-Only

Parameters	Description
<b>ResultType</b>	rtCharacterisation etc,

---

## SimaProServer.ResultIndicatorName

Property ResultIndicatorName(ByVal ResultType As TResultType, ByVal I As Long) As String

Name of an indicator

Member of SimaProServer

Read-Only

Parameters	Description
<b>ResultType</b>	rtCharacterisation etc
<b>I</b>	Index

---

## SimaProServer.ResultMainCompartmentName

Property ResultMainCompartmentName(ByVal I As Long) As String

Name of the main-compartment of a substance

Member of SimaProServer

Read-Only

Parameters	Description
<b>I</b>	Index

---

## SimaProServer.ResultSubCompartmentName

Property ResultSubCompartmentName(ByVal I As Long) As String

Name of the sub-compartment of a substance

Member of SimaProServer

Read-Only

Parameters	Description
<b>I</b>	Index

---

## SimaProServer.Server

Property Server As String

Currently used server, e.g. 'local server' or 'myserver@w.p1.local'

Member of SimaProServer

---

## SimaProServer.Servers

Property Servers As IStrings

List of available servers

Member of SimaProServer

Read-Only

---

## SimaProServer.SubCompartmentCount

Property SubCompartmentCount(ByVal MainCompartmentName As String) As Long

Number of sub-compartments

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

<b>MainCompartmentName</b>	Name of main compartment
----------------------------	--------------------------

---

## SimaProServer.SubCompartmentName

Property SubCompartmentName(ByVal MainCompartmentName As String, ByVal I As Long) As String

Name of a sub-compartment

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

<b>MainCompartmentName</b>	name of maincompartment
----------------------------	-------------------------

<b>I</b>	Index
----------	-------

---

## SimaProServer.SubstanceCASNumber

Property SubstanceCASNumber(ByVal MainCompartmentName As String, ByVal I As Long) As String

CAS number of a substance

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

<b>MainCompartmentName</b>	Maincompartment ('Air', 'Water', etc)
----------------------------	---------------------------------------

<b>I</b>	Index
----------	-------

---

Meant for listing the substances. Addressing with index.

### Example

See FindSubstance

## SimaProServer.SubstanceCount

Property SubstanceCount(ByVal MainCompartmentName As String) As Long

Number of substances

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

<b>MainCompartmentName</b>	Maincompartment ('Air', 'Water', etc))
----------------------------	--

Meant for listing the Substances.

### Example

List all substances

```
For I := 0 to SimaPro.SubstanceCount('Air') - 1 do
    print SimaPro.SubstanceName('Air', I);
```

## SimaProServer.SubstanceDefaultUnit

Property SubstanceDefaultUnit(ByVal MainCompartmentName As String, ByVal I As Long) As String

Default unit of a substance

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

<b>MainCompartmentName</b>	Maincompartment ('Air', 'Water', etc))
----------------------------	--

<b>I</b>	Index
----------	-------

Meant for listing the substances. Addressing with index.

## SimaProServer.SubstanceName

Property SubstanceName(ByVal MainCompartmentName As String, ByVal I As Long) As String

Name of a substance

Member of SimaProServer

Read-Only

Parameters	Description
------------	-------------

<b>MainCompartmentName</b>	Maincompartment ('Air', 'Water', etc))
----------------------------	--

<b>I</b>	Index
----------	-------

Meant for listing the substances. Addressing with index.

### Example

List all substances

```
For I := 0 to SimaPro.SubstanceCount('Air') - 1 do
    print SimaPro.SubstanceName('Air', I);
```

## SimaProServer.Tree

Function Tree(ByVal ProjectName As String, ByVal ProcessType As TProcessType, ByVal ProductName As String, ByVal MethodProjectName As String, ByVal MethodName As String, ByVal NWSetName As String) As Boolean

Perform the tree function for a process or product stage

Member of SimaProServer

Parameters	Description
<b>ProjectName</b>	Name of project
<b>ProcessType</b>	ProcessType (ptMaterial, ptEnergy, etc)
<b>ProductName</b>	Name of product
<b>MethodProjectName</b>	Name of project where methods is stored (often 'methods')
<b>MethodName</b>	Name of method
<b>NWSetName</b>	Name of normalisation weighting set

**Return value**

Returns Boolean

## SimaProServer.TreeCalcScore

Function TreeCalcScore(ByVal ResultType As TResultType, ByVal Param1 As String, ByVal Param2 As String, ByVal Param3 As String) As Boolean

Calculates the node and flow scores of a tree

Member of SimaProServer

Parameters	Description			
<b>ResultType</b>	<b>Param1</b>	<b>Param2</b>	<b>Param3</b>	
rtCharacterisation	Impact Category	-	-	
rtDamage	Damage Category	-	-	
rtNormalisation	Damage Category or Impact Category	-	-	
rtWeighting	Damage Category or Impact Category	-	-	
rtSingleScore	-	-	-	
rtInventory	MainCompartment	Subcompartment	SubstanceName	

**Return value**

Returns Boolean

## SimaProServer.TreeChildNodeCount

Property TreeChildNodeCount(ByVal NodeIndex As Long) As Long

Number of child nodes of a tree node

Member of SimaProServer

Read-Only

Parameters	Description
<b>NodeIndex</b>	Node index

---

## SimaProServer.TreeChildNodeIndex

Property TreeChildNodeIndex(ByVal NodeIndex As Long, ByVal FlowIndex As Long) As Long

Index of a child node of a tree node

Member of SimaProServer

Read-Only

Parameters	Description
<b>NodeIndex</b>	Nodeindex
<b>FlowIndex</b>	Flowindex

---

## SimaProServer.TreeNodeCount

Property TreeNodeCount As Long

Number of nodes in the tree

Member of SimaProServer

Read-Only

---

## SimaProServer.TreeProductName

Property TreeProductName(ByVal NodeIndex As Long) As String

Product name of a tree node

Member of SimaProServer

Read-Only

Parameters	Description
<b>NodeIndex</b>	Nodeindex (refers to list of Nodes of network or tree)

---

## SimaProServer.TreeResult

Function TreeResult(ByVal NodeResultType As TNodeResultType, ByVal NodeIndex As Long) As SimaProTreeResult

Retrieve the data of a tree node

Member of SimaProServer

Parameters	Description
<b>NodeResultType</b>	nrProductAmount, nrIndicatorContribution, nrIndicatorTotal, nrFlowIndicator
<b>NodeIndex</b>	Nodeindex (refers to list of Nodes of network or tree)

**Return value**

Returns SimaProTreeResult

---

## SimaProServer.TreeTopNodeIndex

Property TreeTopNodeIndex As Long  
 Index of the top node of the tree  
 Member of SimaProServer  
 Read-Only

---

## SimaProServer.UnitCount

Property UnitCount(ByVal QuantityName As String) As Long  
 Number of units per quantity  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

QuantityName	Quantity
--------------	----------

### Example

Number of 'Mass' units

```
for I := 0 to Sp.unitCount('Mass') - 1 do
  print Sp.UnitName('Mass', I);
```

---

## SimaProServer.UnitDefault

Property UnitDefault(ByVal QuantityName As String) As String  
 Default unit of a quantity (factor = 1)  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

QuantityName	Quantity ('Mass', 'Volume' etc)
--------------	---------------------------------

---

## SimaProServer.UnitFactor

Property UnitFactor(ByVal QuantityName As String, ByVal I As Long) As Double  
 Factor of a unit  
 Member of SimaProServer  
 Read-Only

Parameters	Description
------------	-------------

QuantityName	Quantity ('Mass', 'Volume' etc)
--------------	---------------------------------

I	Index
---	-------

## SimaProServer.UnitMetric

Property UnitMetric(ByVal QuantityName As String, ByVal I As Long) As Boolean

Indicates if a unit is metric

Member of SimaProServer

Read-Only

Parameters	Description
<b>QuantityName</b>	Quantity ('Mass', 'Volume' etc)
<b>I</b>	Index

## SimaProServer.UnitName

Property UnitName(ByVal QuantityName As String, ByVal I As Long) As String

Name of a unit

Member of SimaProServer

Read-Only

Parameters	Description
<b>QuantityName</b>	Quantity ('Mass', 'Volume' etc)
<b>I</b>	Index

### Example

Number of 'Mass' units

```
for I := 0 to Sp.unitCount('Mass') - 1 do
    print Sp.UnitName('Mass', I);
```

## SimaProServer.WasteTypeCount

Property WasteTypeCount As Long

Number of waste-types

Member of SimaProServer

Read-Only

## SimaProServer.WasteTypeName

Property WasteTypeName(ByVal I As Long) As String

Name of a waste-type

Member of SimaProServer

Read-Only

Parameters	Description
<b>I</b>	Index

## SimaProTreeResult

Class SimaProTreeResult

Properties

SimaProTreeResult object

---

## SimaProTreeResult properties

SimaProTreeResult    Legend

Amount

ProductName

UnitName

Valid

### SimaProTreeResult.Amount

Property Amount As Double

Amount of the product

Member of SimaProTreeResult

---

### SimaProTreeResult.ProductName

Property ProductName As String

Name of the product

Member of SimaProTreeResult

---

### SimaProTreeResult.UnitName

Property UnitName As String

Unit of the amount

Member of SimaProTreeResult

---

### SimaProTreeResult.Valid

Property Valid As Boolean

Indicates if the node is part of the tree

Member of SimaProTreeResult

---

## StandardDeviation Property

Select one of the available subtopics below to see detailed help on **StandardDeviation** property

Object	Property description
ParamLine	Standard deviation of the input parameter
ProcessLine	Standard deviation of amount

## SubCompartmentName Property

Select one of the available subtopics below to see detailed help on **SubCompartmentName** property

Object	Property description
SimaProAnalyseResult	
SimaProServer	Name of a sub-compartment

## Substance

Class **Substance**

Properties    Methods

Represents a substance in SimaPro. Use to edit, find or use substances.

## Substance methods

Substance    Legend  
 Cancel  
 Edit  
 Update

## Substance properties

Substance    Legend  
 CASNumber  
 Comment  
 DefaultUnit  
 ► MainCompartment  
 ► Mode  
 Name

## Substance.Cancel

Sub Cancel()

Cancel edit mode and returns to read mode

Member of **Substance**

## Substance.CASNumber

Property CASNumber As String  
CAS number  
Member of **Substance**

### Example

```
Substance.Casnumber := '45-32-45'
```

---

## Substance.Comment

Property Comment As IStrings  
Comment  
Member of **Substance**

---

## Substance.DefaultUnit

Property DefaultUnit As String  
Default unit, defines also the quantity  
Member of **Substance**

---

## Substance.Edit

Sub Edit()  
Set the object in edit mode  
Member of **Substance**

---

## Substance.MainCompartment

Property MainCompartment As String  
Main compartment (e.g. 'Airborne emission')  
Member of **Substance**  
Read-Only

---

## Substance.Mode

Property Mode As String

Mode of the object, can be: Read, New or Edit

Member of **Substance**

Read-Only

---

## Substance.Name

Property Name As String

Name of the substance (e.g. 'Carbon dioxide')

Member of **Substance**

---

## Substance.Update

Sub Update()

Store the data of the object in the database and switch to read mode

Member of **Substance**

### Example

Create a new substance

```
SimaPro.CreateSubstance('Air', Substance)
Substance.Name := 'My new substance'
Substance.UnitName := 'kg';
Substance.Update; // save in database
```

---

## TDistribution

Enum TDistribution

Constant	Value	Description
dsUndefined	0	Distribution is not defined
dsLogNormal	1	Lognormal
dsNormal	2	Normal (Gaussian)
dsTriangle	3	Triangle
dsUniform	4	Uniform

---

## TNodeResultType

Enum TNodeResultType

Type of result from a network or tree node

Constant	Value	Description
nrProductAmount	0	Amount of a product
nrIndicatorContribution	1	Contribution of a product to the selected indicator
nrIndicatorTotal	2	Contribution of a product including all sub-processes to the selected indicator
nrFlowIndicator	3	Contribution of a flow to the selected indicator

## TParameterType

Enum TParameterType

Constant	Value	Description
ptInputParameter	0	Parameter is a constant value optional with distribution data
ptCalculatedParameter	1	Parameter is an expression

## TProcessPart

Enum TProcessPart

Parts of a process

Constant	Value	Description
ppProducts	0	Products (outputs)
ppMaterialsFuels	1	Inputs from technosphere (other processes)
ppElectricityHeat	2	Inputs from technosphere (other processes)
ppAvoidedProducts	3	Avoided product
ppWasteToTreatment	4	Waste
ppRawMaterials	5	Use of resources (raw materials)
ppAirborneEmissions	6	Emissions to air
ppWaterborneEmissions	7	Emissions to water
ppFinalWasteFlows	8	Emissions to waste
ppEmissionsToSoil	9	Emissions to soil
ppNonMaterialEmissions	10	Non material emissions
ppSocialIssues	11	Social issues
ppEconomicIssues	12	Economic issues
ppSpecificWaste	13	Outputs to specific waste
ppRemainingWaste	14	Remaining waste
ppSubAssembly	15	Subassembly (product stages only)

ppReferencedAssembly	16	Referenced assembly (product stages only)
ppAssembliesAndMaterials	17	Assemblies or materials (product stages only)
ppProcesses	18	Process (product stages only)
ppWasteScenarios	19	Waste scenario (product stages only)
ppDisposalScenarios	20	Disposal scenario (product stages only)
ppAdditionalLifeCycles	21	Additional life cycle (Life cycle product stage only)
ppDisassemblies	22	Disassembly (product stages only)
ppReuses	23	Reuse (product stages only)
ppWasteOrDisposalScenario	24	Waste or disposal scenario (product stages only)

---

## TProcessStatus

Enum TProcessStatus

Constant	Value	Description
stEmpty	0	No status
stTemporary	1	Temporary process
stDraft	2	Draft, work to be done
stToBeRevised	3	To be revised
stToBeReviewed	4	To be reviewed
stFinished	5	Finished

---

## TProcessType

Enum TProcessType

Constant	Value	Description
ptMaterial	0	Material process
ptEnergy	1	Energy Process
ptTransport	2	Transport process
ptProcessing	3	Processing process
ptUse	4	Use process
ptWasteScenario	5	Waste scenario
ptWasteTreatment	6	Waste treatment
ptAssembly	7	Assembly product stage
ptLifeCycle	8	Life cycle product stage
ptDisposalScenario	9	Disposal scenario
ptDisassembly	10	Disassembly
ptReuse	11	Reuse

## TResultType

Enum TResultType

Type of result

Constant	Value	Description
rtCharacterisation	0	Characterisation score
rtDamage	1	Damage score
rtNormalisation	2	Normalised score
rtWeighting	3	Weighted score
rtSingleScore	4	Single score
rtInventory	5	Inventory results (LCI)

## UnitName Property

Select one of the available subtopics below to see detailed help on **UnitName** property

Object	Property description
ProcessLine	Name of the unit of amount
SimaProAnalyseResult	Unit (e.g. kg, m3)
SimaProNetworkResult	Unit of the amount
SimaProServer	Name of a unit
SimaProTreeResult	Unit of the amount

## Update Method

Select one of the available subtopics below to see detailed help on **Update** method

Object	Method description
Process	Store the data of the object in the database and switch to read mode
Substance	Store the data of the object in the database and switch to read mode

## Index

- AddLine, 13
- AddParamLine, 14
- Alias, 30, 31, 34, 40
- Calculate, 24, 30, 31, 37, 46, 47, 50
- Database, 31, 34, 40
- DeleteLine, 15
- DeleteParamLine, 15
- Distribution**, 53
- FindParameter, 16
- Line, 16
- LineCount, 17
- Mode, 17
- Network, 24, 26, 28, 30, 37, 42, 46, 48, 49, 50, 52, 54, 55, 56
- ParamLine, 9, 17
  - Comment, 10
  - properties, 9
- ParamLineCount, 18
- Process, 13, 24, 26, 28, 30, 37, 42, 46, 48, 49, 50, 52, 54, 55, 56
- ProcessLine, 19
- Product stage, 24, 26, 28, 30, 37, 42, 46, 48, 49, 50, 52, 54, 55, 56
- Project, 32, 33, 40, 42
- Quantity, 42, 48, 52
- SimaProAnalyseResult, 25
- SimaProCalculationError, 26
- SimaProNetworkResult, 27
- SimaProServer, 28
  - QuantityName, 42
- SimaProTreeResult, 50
- Substance, 33, 35, 45, 51
- TNodeResultType**, 54
- TParameterType**, 54
- TProcessPart**, 54
- TProcessStatus**, 55
- TProcessType**, 55
- Tree, 24, 26, 28, 30, 37, 42, 46, 47, 48, 49, 50, 52, 54, 55, 56
- TResultType**, 56
- Uncertainty, 8, 10, 20, 53, 54
- Unit, 24, 26, 28, 42, 48, 49, 50, 52, 56
- User, 32, 33, 40, 42
- Waste, 15, 18, 23, 24