
## Proprietary Notice

## Revision History

| Rev. | Date | Author | Sections Affected | Description |
|---|---|---|---|---|
| 1 | Mar 12, 2009 | Kai Scheffler | All | Document created. |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

# Table of Contents

**Table of Contents** ii

**1. INTRODUCTION** 1

**1.1 Purpose** 1

**1.2 Scope** 1

**1.3 Applicable Documents** 1

**1.4 Codes and Standards** 1

**1.5 Abbreviations and Definitions** 1

**2. Design** 3

**2.1 Introduction** 3

2.1.1 Description 3

**2.2 Interfaces and Type Definitions** 4

2.2.1 Abstract class Hook 4

2.2.1.1 General 4

2.2.1.2 Definition 4

2.2.1.3 Methods and Properties 5

2.2.1.3.1 Constructor Hook 5

2.2.1.3.2 Property ConfigId 5

2.2.1.3.3 Property RunningCyclic 5

2.2.1.3.4 Property ErrorOutput 5

2.2.1.3.5 Property TestCellId 6

2.2.1.3.6 Property TestId 6

2.2.1.3.7 Property EventId 6

2.2.1.3.8 Property InputNames 6

2.2.1.3.9 Method DefineInputNames 6

2.2.1.3.10 Property OutputNames 7

2.2.1.3.11 Method DefineOutputNames 7
2.2.1.3.12 Property OutputUnits 7
2.2.1.3.13 Method CalculateDirect 8
2.2.1.3.14 Method Flush 8
2.2.1.3.15 Method PostProcessing 9
2.2.2 Class Merger 9
2.2.2.1 General 9
2.2.2.2 Definition 9
2.2.2.3 Methods and Properties 10
2.2.2.3.1 Constructor Merger 10
2.2.2.3.2 Property Hook 10
2.2.2.3.3 Property Subsystem 11
2.2.2.3.4 Property Service 11
2.2.2.3.5 Property OutputValues 11
2.2.2.3.6 Property OutputQualities 11
2.2.2.3.7 Method Calculate 11
2.2.2.3.8 Method PostProcessing 12
2.2.3 Class Collector 13
2.2.3.1 General 13
2.2.3.2 Definition 13
2.2.3.3 Methods and Properties 13
2.2.3.3.1 Constructor Collector 13
2.2.3.3.2 Property Mergers 14
2.2.3.3.3 Property Assemblies 14
2.2.3.3.4 Property InputNamesAllHooks 14
2.2.3.3.5 Property OutputNamesAllHooks 15
2.2.3.3.6 Property OutputUnitsAllHooks 15
2.2.3.3.7 Method Initialise 15
2.2.3.3.8 Method Calculate 15

2.2.3.3.9 Method PostProcessing 16
2.2.3.3.10 Method Flush 16
2.2.4 Abstract class EditConfig 17
2.2.4.1 General 17
2.2.4.2 Definition 17
2.2.4.3 Methods and Properties 17
2.2.4.3.1 Constructor EditConfig 17
2.2.4.3.2 Method Show 17
2.2.4.3.3 Method CreateObject 17

# 1. INTRODUCTION

## 1.1 Purpose

1.1.1 The proDAS External Hook Framework enables customers to develop programs which can be hooked into the proDAS system. The framework does not only support development of external programs, it also provides a runtime environment which allows executing these programs during test runs using online (measured) and offline (archived) data.

## 1.2 Scope

1.2.1 This document is intended for programmers of the client components using the interfaces specified herein.

## 1.3 Applicable Documents

| Number | Title |
|---|---|
| ES78001.2620 | Functional Requirements Document for proDAS |
| ES78045.4349 | Engineering Specification for External Hook Framework |
| DB78045.4351 | Design Brief for External Hook Framework |
| ICD78045.4059 | Interface Control Document for Native Database Assembly .NET |
| ICD78031.2661 | Interface Control Document for Configuration Server |

## 1.4 Codes and Standards

N/A

## 1.5 Abbreviations and Definitions

| Term | Definition |
|---|---|
| .NET | The .NET Framework is an integral Windows component that supports building and running of applications. |
| CVT | Current Value Table |

| Term | Definition |
|---|---|
| EHF | External Hook Framework – a proDAS component that supports building and running of External Hook Programs (EHP). |
| EHL | External Hook Library –. |
| EHP | External Hook Program |
| ICD | Interface Control Document |

# 2. DESIGN

## 2.1 Introduction

### 2.1.1 Description

2.1.1.1 The External Hook Framework has two main goals:

1) Enabling proDAS to hook up external programs during runtime, and
2) Supporting developers in creating libraries that can be hooked up by proDAS.

2.1.1.2 The External Hook Framework provides an abstract .NET class which allows .NET clients to use the framework.

2.1.1.3 The interfaces discussed in the following are all defined by the External Hook Framework. The EHF does not provide a full implementation for all of it. Since the EHF depends on these interfaces each External Hook Program must implement its own behaviour. This design results in a high flexibility of the External Hook Framework, but the developer will have to take care that the requirements of the EHF will be fulfilled.

2.1.1.4 There are three External Hook Programs Cyclic.exe, Fullset.exe and Recalculation.exe.

2.1.1.5 These Programs can be configured to dynamically load one or more External Hook Libraries. Each library will receive input values from the External Hook Programs.

2.1.1.6 An External Hook Library is an assembly which main class is derived from the class *Hook* from the External Hook Framework. An EHL implements calculations based on input values and provides the result as output values.

2.1.1.7 Instead of implementing the calculations within the library, it is also possible to pass the input data to an external program which performs the calculations and returns the results back to the library.

2.1.1.8 An External Hook Library could also perform measurements by connecting to a measurement device. In this case there is no need to obtain input values from the External Hook Programs. Instead, only output values are provided.

## 2.2 Interfaces and Type Definitions

### 2.2.1 Abstract class *Hook*

#### 2.2.1.1 General

2.2.1.1.1 This class is provided by the External Hook Framework. It is the main interface between the ExtHook libraries and the ExtHook programs.

2.2.1.1.2 For an ExtHook library, this is the only class to be derived and implemented.

#### 2.2.1.2 Definition

```
public abstract class Hook
{
   public Hook()

   public static int ConfigId { get; set; }

   public static bool RunningCyclic { get; set; }

   public static bool ErrorOutput { get; set; }

   public static int TestCellId { get; set; }

   public static int TestId { get; set; }

   public static int EventId { get; set; }

   protected System.Collections.ArrayList inputNames = null;

   public System.Collections.ArrayList InputNames { get; }

   public virtual void DefineInputNames()

   protected System.Collections.ArrayList outputNames = null;

   public System.Collections.ArrayList OutputNames { get; }

   public virtual void DefineOutputNames()

   protected System.Collections.ArrayList outputUnits = null;

   public System.Collections.ArrayList OutputUnits { get; }

   public abstract bool CalculateDirect(
      double[] inValues,
      string[] inQualities,
      out double[] outValues,
      out string[] outQualities)
```

```
    public virtual void Flush()

    public abstract void PostProcessing()
}
```

#### 2.2.1.3 Methods and Properties

##### 2.2.1.3.1 Constructor *Hook*

```
public Hook()
```

2.2.1.3.1.1 This is the default constructor.

##### 2.2.1.3.2 Property *ConfigId*

```
public static int ConfigId { get; set; }
```

2.2.1.3.2.1 This property allows an External Hook Program to specify a test configuration ID which the External Hook Library can use.

2.2.1.3.2.2 This property usually is set by an External Hook Program during initialization. By default, this property is set to *-1*.

##### 2.2.1.3.3 Property *RunningCyclic*

```
public static bool RunningCyclic { get; set; }
```

2.2.1.3.3.1 This property allows an External Hook Program to specify whether it is running cyclic or not. Per definition, cyclic programs work on the CVT and static programs work on the fullset database.

2.2.1.3.3.2 ExtHook libraries will use this property if the calculation process depends on it.

2.2.1.3.3.3 This property usually is set by an ExtHook program during initialization. By default, this property is set to *false*.

##### 2.2.1.3.4 Property *ErrorOutput*

```
public static bool ErrorOutput { get; set; }
```

2.2.1.3.4.1 This property indicates whether the error output of the calculation process of the ExtHook libraries should be traced using the global trace object.

2.2.1.3.4.2 This property should be set by an ExtHook program during initialization. By default, this property is set to *false*.

#### 2.2.1.3.5 Property *TestCellId*

```
public static int TestCellId { get; set; }
```

2.2.1.3.5.1 This property allows an ExtHook program to specify a test cell ID which the ExtHook library can use.

2.2.1.3.5.2 This property usually is set by an ExtHook program during initialization. By default, this property is set to *-1*.

#### 2.2.1.3.6 Property *TestId*

```
public static int TestId { get; set; }
```

2.2.1.3.6.1 This property allows an ExtHook program to specify a test ID which the ExtHook library can use.

2.2.1.3.6.2 This property usually is set by an ExtHook program during initialization. By default, this property is set to *-1*.

#### 2.2.1.3.7 Property *EventId*

```
public static int EventId { get; set; }
```

2.2.1.3.7.1 This property allows an ExtHook program to specify an event ID which the ExtHook library can use. This property is not used by ExtHook programs running cyclic.

2.2.1.3.7.2 This property usually is set by a ExtHook program running static when a new scan is detected. By default, this property is set to *-1*.

#### 2.2.1.3.8 Property *InputNames*

```
public System.Collections.ArrayList InputNames { get; }
```

2.2.1.3.8.1 This property allows an ExtHook program to retrieve the defined input names from an ExtHook library. The ExtHook program can then check whether the input names exist.

2.2.1.3.8.2 This property accesses the field *inputNames*, which is filled by the ExtHook library during the method *DefineInputNames*. By default, the field *inputNames* is *null*.

#### 2.2.1.3.9 Method *DefineInputNames*

```
public virtual void DefineInputNames()
```

2.2.1.3.9.1 This method is called by an ExtHook program for each ExtHook library during initialization.

2.2.1.3.9.2 In this method an ExtHook library should initialize and populate the field *inputNames*. During a calculation the ExtHook library will then receive an input value for each defined input name. If there are no input values required, it is strongly recommended that the field inputNames is instantiated anyway but without any items to avoid null reference exceptions.

#### 2.2.1.3.10 Property *OutputNames*

```
public System.Collections.ArrayList OutputNames { get; }
```

2.2.1.3.10.1 This property allows an ExtHook program to retrieve the defined output names from an ExtHook library. The ExtHook program can then check whether the output names exist.

2.2.1.3.10.2 This property accesses the field *outputNames*, which is filled by the ExtHook library during the method *DefineOutputNames*. By default, the field *outputNames* is *null*.

#### 2.2.1.3.11 Method *DefineOutputNames*

```
public virtual void DefineOutputNames()
```

2.2.1.3.11.1 This method is called by an ExtHook program for each ExtHook library during initialization.

2.2.1.3.11.2 In this method an ExtHook library should initialize and populate the field *outputNames*. During a calculation the ExtHook library should then provide a value for each of these output names. If there are no output values required, it is strongly recommended that the field outputNames is instantiated anyway but without any items to avoid null reference exceptions.

#### 2.2.1.3.12 Property *OutputUnits*

```
public System.Collections.ArrayList OutputUnits { get; }
```

2.2.1.3.12.1 This property allows an ExtHook program to retrieve the defined output units from an ExtHook library.

2.2.1.3.12.2 This property accesses the field *outputUnits*, which is filled by the ExtHook library during the method *DefineOutputNames*. By default, the field *outputUnits* is *null*.

#### 2.2.1.3.13 Method *CalculateDirect*

```
public abstract bool CalculateDirect(
   double[] inValues,
   string[] inQualities,
   out double[] outValues,
   out string[] outQualities)
```

| Argument | Description |
|---|---|
| inValues | An array with the input values. |
| inQualities | An array with the input qualities. |
| outValues | An array returning the computed resulting values. |
| outQualities | An array returning the resulting qualities. |

2.2.1.3.13.1 This method is called by an ExtHook program for each ExtHook library for each calculation cycle. A calculation cycle can be triggered e.g. by a timer (e.g. each second) or by a specific event (e.g. fullset).

2.2.1.3.13.2 In this method an ExtHook library receives the input values and qualities for each defined input name. With these values the method should perform the calculation for the current cycle. This can be done inside the method or inside an external application which is referenced by the ExtHook library.

2.2.1.3.13.3 The parameters *outValues* and *outQualities* are passed with the value *null* and must be initialized before being filled. The number of items should equal the number of output names defined before. If there are no output values required, it is strongly recommended that the fields outValues and outQualities are instantiated anyway but without any items to avoid null reference exceptions.

#### 2.2.1.3.14 Method *Flush*

```
public virtual void Flush()
```

2.2.1.3.14.1 This method is called by an ExtHook program for each ExtHook library when triggered by a user or automatically after certain events, e.g. during initialization and after stopping calculation.

2.2.1.3.14.2 If the ExtHook library uses an external application for calculation, this method must implement the flushing of files of this very application if possible. This is useful in the case of an error, e.g. crash of the external application, to receive the latest version of a used log file.

#### 2.2.1.3.15 Method *PostProcessing*

```
public abstract void PostProcessing()
```

2.2.1.3.15.1 After the calculation cycle is finished for all ExtHook libraries, this method is called by an ExtHook program for each ExtHook library.

2.2.1.3.15.2 In this method an ExtHook library could perform actions which are necessary after a calculation, e.g. sending the data to another destination.

### 2.2.2 Class *Merger*

#### 2.2.2.1 General

2.2.2.1.1 This class is provided by the External Hook Framework. It is a wrapper class for the abstract class *Hook*.

2.2.2.1.2 An ExtHook program should not use the class *Hook* directly but the class *Merger*. Each instance of the class *Merger* handles exactly one instance of the class *Hook*.

2.2.2.1.3 If an ExtHook program manages more than one ExtHook library, the class *Merger* should not be used but the class *Collector*.

2.2.2.1.4 The class *Merger* can be accessed directly and need not to be derived.

2.2.2.1.5 This class must not be used by an ExtHook library.

#### 2.2.2.2 Definition

```
public class Merger
{
   public Merger(string subsystem, string service)

   public Hook Hook { get; set; }

   public string Subsystem { get; }

   public string Service { get; }

   public double[] OutputValues { get; }

   public string[] OutputQualities { get; }
```

```
        public bool Calculate(
            double[] inValuesAllHooks,
            string[] inQualitiesAllHooks,
            ref double[] outValuesAllHooks,
            ref string[] outQualitiesAllHooks,
            System.Collections.Hashtable inputNamesAllHooksIndexHash,
            System.Collections.Hashtable outputNamesAllHooksIndexHash)

        public void PostProcessing()
    }
```

#### 2.2.2.3 Methods and Properties

##### 2.2.2.3.1 Constructor *Merger*

```
public Merger(string subsystem, string service)
```

| Argument | Description |
|---|---|
| subsystem | The subsystem string is used to determine the assembly file name which should be loaded dynamically. In the External Hook Programs Cyclic.exe, Fullset.exe and Recalculation.exe the subsystem string is defined in the respective INI files. |
| service | This string identifies the service which is defined in the proDAS RTE for the corresponding subsystem. |

2.2.2.3.1.1 This is the only constructor to be used because it is the only way to set the subsystem and service string.

2.2.2.3.1.2 In the not recommended case of deriving this class, the new constructor should ensure to call the constructor of the base class with the necessary arguments.

2.2.2.3.1.3 In the constructor, .NET Reflection is used to create an instance of an External Hook Library. This is done by a naming convention which uses the subsystem string to create a DLL file name.

2.2.2.3.1.4 The service string is only necessary for cyclically running External Hook Programs. This is because each External Hook Library is only designed to set proDAS channel values of a specific proDAS subsystem.

##### 2.2.2.3.2 Property *Hook*

```
public Hook Hook { get; set; }
```

2.2.2.3.2.1 This property allows an ExtHook program to access the underlying instance of the class Hook. But usually this is not necessary since all information can be retrieved through the class *Merger*.

#### 2.2.2.3.3 Property *Subsystem*

```
public string Subsystem { get; }
```

2.2.2.3.3.1 This property allows an ExtHook program to retrieve the corresponding subsystem string which was passed to the constructor.

#### 2.2.2.3.4 Property *Service*

```
public string Service { get; }
```

2.2.2.3.4.1 This property allows an ExtHook program to retrieve the corresponding service string which was passed to the constructor.

#### 2.2.2.3.5 Property *OutputValues*

```
public double[] OutputValues { get; }
```

2.2.2.3.5.1 This property allows an ExtHook program to retrieve the output values after a calculation. This array contains the values which were returned from the method *CalculateDirect* of the class *Hook*.

#### 2.2.2.3.6 Property *OutputQualities*

```
public string[] OutputQualities { get; }
```

2.2.2.3.6.1 This property allows an ExtHook program to retrieve the output qualities after a calculation. This array contains the values which were returned from the method *CalculateDirect* of the class *Hook*.

#### 2.2.2.3.7 Method *Calculate*

```
public bool Calculate(
   double[] inValuesAllHooks,
   string[] inQualitiesAllHooks,
   ref double[] outValuesAllHooks,
   ref string[] outQualitiesAllHooks,
   System.Collections.Hashtable inputNamesAllHooksIndexHash,
   System.Collections.Hashtable outputNamesAllHooksIndexHash)
```

| Argument | Description |
|---|---|

| | |
|---|---|
| inValuesAllHooks | This array contains all input values of all ExtHook libraries and is provided by the ExtHook program. |
| inQualititesAllHooks | This array contains all input qualities of all ExtHook libraries and is provided by the ExtHook program. |
| outValuesAllHooks | This array is a container for all output values of all ExtHook libraries. When the method is called, only the output values of the correspondent library are filled. |
| outQualitiesAllHooks | This array is a container for all output qualities of all ExtHook libraries. When the method is called, only the output qualities of the correspondent library are filled. |
| inputNamesAllHooksIndexHash | This hashtable provides the index of each input name where it can be found in the above arrays. |
| outputNamesAllHooksIndexHash | This hashtable provides the index of each output name where it can be found in the above arrays. |

2.2.2.3.7.1 This method is called by the ExtHook program for each calculation cycle. The caller has to provide the input values and qualities, an initialized array for the output values and qualities of the correct size and the hashtables to find the correct indices of input and output names within those arrays.

2.2.2.3.7.2 This method calls the method *CalculateDirect* of the class *Hook* and after that copies the output values and qualities to the correct position of the corresponding global arrays. The global arrays can be understood as containers which hold all values and qualities of all involved ExtHook libraries. The order of the output values and qualities is defined by the order of the output names.

### 2.2.2.3.8 Method *PostProcessing*

```
public void PostProcessing()
```

2.2.2.3.8.1 After the calculation cycle is finished for each ExtHook library, this method is called by an ExtHook program for each ExtHook library.

2.2.2.3.8.2 In this method just the method *PostProcessing* of the class *Hook* is called.

### 2.2.3 Class *Collector*

#### 2.2.3.1 General

2.2.3.1.1 This class is provided by the External Hook Framework. It is a container class for an array of instances of the class *Merger*.

2.2.3.1.2 An ExtHook program should use the class *Collector* if it manages more than one ExtHook library (which itself is managed by the class *Merger* and *Hook*).

2.2.3.1.3 The class *Collector* can be accessed directly and need not to be derived.

2.2.3.1.4 This class must not be used by an ExtHook library.

#### 2.2.3.2 Definition

```
public class Collector
{
   public Collector(string[] subsystems, string[] services)

   public System.Collections.Generic.List<Merger> Mergers { get; }

   public System.Collections.ArrayList Assemblies { get; }

   public System.Collections.ArrayList InputNamesAllHooks { get; }

   public System.Collections.ArrayList OutputNamesAllHooks { get;
}

   public virtual bool Initialise()

   public void Calculate(
      string[] inNamesSource,
      double[] inValuesSource,
      string[] inQualitiesSource,
      out double[] outValuesAllHooks,
      out string[] outQualitiesAllHooks)

   public void PostProcessing()

   public void Flush()
}
```

#### 2.2.3.3 Methods and Properties

##### 2.2.3.3.1 Constructor *Collector*

```
public Collector(string[] subsystems, string[] services)
```

| Argument | Description |
|---|---|

| | |
|---|---|
| subsystems | An array of subsystems. In the External Hook Programs Cyclic.exe, Fullset.exe and Recalculation.exe the subsystem string is defined in the respective INI files. |
| services | An array of services. A single string identifies the service which is defined in the proDAS RTE for the corresponding subsystem. |

2.2.3.3.1.1 The constructor is used to pass the used subsystem and service names to the class.

2.2.3.3.1.2 In the not recommended case of deriving this class, the new constructor should ensure to call the constructor of the base class with the necessary arguments.

2.2.3.3.1.3 The constructor creates an array of mergers where each merger is created by a combination of a subsystem and service string.

#### 2.2.3.3.2 Property *Mergers*

```
public System.Collections.Generic.List<Merger> Mergers { get; }
```

2.2.3.3.2.1 This property allows an ExtHook program to access the underlying list of defined mergers. A merger is a wrapper around the class *Hook* which itself may reference an external application.

2.2.3.3.2.2 Usually an ExtHook program which uses the class *Collector* does not need direct access to the underlying mergers because the class *Collector* already provides the list of all output values of all mergers. But if it is necessary to distinguish between the values of the different mergers, this property is useful.

#### 2.2.3.3.3 Property *Assemblies*

```
public System.Collections.ArrayList Assemblies { get; }
```

2.2.3.3.3.1 This property allows an ExtHook program to access the defined assemblies (ExtHook libraries). This may be necessary to provide the user with information about the used assemblies, e.g. version number.

#### 2.2.3.3.4 Property *InputNamesAllHooks*

```
public System.Collections.ArrayList InputNamesAllHooks { get; }
```

2.2.3.3.4.1 This property allows an ExtHook program to retrieve the defined input names for all defined mergers (ExtHook libraries).

2.2.3.3.4.2 This property accesses the field *inputNamesAllHooks*, which is filled by the method *Initialise*.

#### 2.2.3.3.5 Property *OutputNamesAllHooks*

```
public System.Collections.ArrayList OutputNamesAllHooks { get; }
```

2.2.3.3.5.1 This property allows an ExtHook program to retrieve the defined output names for all defined mergers (External Hook Libraries). The order of the names is defined by the External Hook Libraries.

2.2.3.3.5.2 This property accesses the field *outputNamesAllHooks*, which is filled by the method *Initialise*.

#### 2.2.3.3.6 Property *OutputUnitsAllHooks*

```
public System.Collections.ArrayList OutputUnitsAllHooks { get; }
```

2.2.3.3.6.1 This property allows an ExtHook program to retrieve the defined output units for all defined mergers (External Hook Libraries). The order of the names is defined by the External Hook Libraries.

2.2.3.3.6.2 This property accesses the field *outputUnitsAllHooks*, which is filled by the method *Initialise*.

#### 2.2.3.3.7 Method *Initialise*

```
public virtual bool Initialise()
```

2.2.3.3.7.1 This method is called by the ExtHook program to initialise the instance of the class *Collector*. The call has to be done before the first calculation call takes place.

2.2.3.3.7.2 This method creates the array lists for the fields *inputNamesAllHooks*, *outputNamesAllHooks* and *outputUnitsAllHooks*.

2.2.3.3.7.3 After that the method iterates through all created mergers to call their methods *DefineInputNames* and *DefineOutputNames*. All the resulting input and output names and output units are collected in the array lists described in 2.2.3.3.7.2.

2.2.3.3.7.4 Additionally, hashtables (*inputNamesAllHooksIndexHash* and *outputNamesAllHooksIndexHash*) are created to map each name to an index. These hashtables are used as parameters in the method *Calculate* of the class *Merger*.

#### 2.2.3.3.8 Method *Calculate*

```
public void Calculate(
   string[] inNamesSource,
   double[] inValuesSource,
   string[] inQualitiesSource,
   out double[] outValuesAllHooks,
   out string[] outQualitiesAllHooks)
```

| Argument | Description |
|---|---|
| inNamesSource | This array contains all input names collected by the ExtHook program, e.g. from CVT or database. |
| inValuesSource | This array contains all input values collected by the ExtHook program, e.g. from CVT or database. |
| inQualitiesSource | This array contains all input qualities collected by the ExtHook program, e.g. from CVT or database. |
| outValuesAllHooks | This array is a container for all output values of all ExtHook libraries and is filled during the calculation. |
| outQualitiesAllHooks | This array is a container for all output qualities of all ExtHook libraries and is filled during the calculation. |

2.2.3.3.8.1 This method is called by the ExtHook program for each calculation cycle. The caller has to provide the input names, values and qualities. The out parameters are passed uninitialized and are filled during the calculation.

2.2.3.3.8.2 This method calls the method *Calculate* of each instance of the class *Merger* which are defined inside the class *Collector*.

#### 2.2.3.3.9 Method *PostProcessing*

```
public void PostProcessing()
```

2.2.3.3.9.1 After the calculation cycle is finished for each ExtHook library, this method is called by the ExtHook program.

2.2.3.3.9.2 The method delegates this call to the method *PostProcessing* of each instance of the class *Merger* which are defined inside the class *Collector*.

#### 2.2.3.3.10 Method *Flush*

```
public void Flush()
```

2.2.3.3.10.1 This method is called by an ExtHook program when triggered by a user or automatically after certain events, e.g. during initialization and after stopping calculation.

2.2.3.3.10.2 This method delegates this call to the method *Flush* of each instance of the class *Merger* which are defined inside the class *Collector*.

### 2.2.4 Abstract class *EditConfig*

#### 2.2.4.1 General

2.2.4.1.1 This class is provided by the External Hook Framework. It is an abstract class that is used to instantiate an edit view to adjust the settings of a subsystem.

2.2.4.1.2 Usually each External Hook Library defines an edit view to provide an easy way for configuration. For this purpose, the library should provide a class which is derived from *EditConfig*.

#### 2.2.4.2 Definition

```
public class EditConfig
{
   public EditConfig()

   public abstract void Show();

   static public EditConfig CreateObject(string subsystem)
}
```

#### 2.2.4.3 Methods and Properties

##### 2.2.4.3.1 Constructor *EditConfig*

```
public EditConfig()
```

2.2.4.3.1.1 This is the default constructor.

##### 2.2.4.3.2 Method *Show*

```
public abstract void Show();
```

2.2.4.3.2.1 This method is defined abstract. A class derived from *EditConfig* within an ExtHook library has to overwrite this method to show the specific edit view of that library.

##### 2.2.4.3.3 Method *CreateObject*

```
static public EditConfig CreateObject(string subsystem)
```

| Argument | Description |
|---|---|
| subsystem | The subsystem string can be retrieved from the ExtHook library file name to be used by removing the substrings ‘ExtHook.’ and ‘.dll’. |

2.2.4.3.3.1 This method is called by the ExtHook program to retrieve a new instance of the edit view of a specific subsystem. If the specific subsystem does not provide an edit view, this method returns a null reference.