|  |  |  |
| --- | --- | --- |
| doc-header-print-portrait | |  |
|  | | Engineering Design Document |
|  | | Subsystem Interfaces of Security Framework |
|  | **Version: 1.01**  **Last Revised:**  **Author: Maksym Sukhovarov** | |

Table of Contents

[1 System Overview 4](#_Toc271212143)

[2 Assumptions and Dependencies 4](#_Toc271212144)

[3 Functional Design 4](#_Toc271212145)

[4 Technical Design 5](#_Toc271212146)

[4.1 Class Diagram 5](#_Toc271212147)

[4.2 Interface Description 6](#_Toc271212148)

[4.3 UI Design 9](#_Toc271212149)

[4.4 High-Level Design / Tasks 9](#_Toc271212150)

[4.5 Database Schema Changes 9](#_Toc271212151)

[5 Error list 9](#_Toc271212152)

[6 Outstanding Issues 10](#_Toc271212153)

Document Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Revision No.** | **Date** | **Description of Change** |
| **Maksym Sukhovarov** | 1.0 | 08/26/2010 | Initial version |
| **Anatoly Lokshin** | 1.0 | 08/26/2010 | Minor Correction |
| **Anatoly Lokshin** | 1.0 | 08/31/2010 | Translation |
| **Maksym Sukhovarov** | 1.0 | 09/01/2010 | SubsystemInternalException and SubsystemInputParamException was added for description interfaces |
| **Maksym Sukhovarov** | 1.1 | 09/02/2010 | IsSubsystemInfo interface and description was added |
| **Maksym Sukhovarov** | 1.1 | 01/15/2011 | Added reference to specification |
|  |  |  |  |

Functional Design Review

| **Name** | **Title** | **Date Reviewed** | **Date Approved** |
| --- | --- | --- | --- |
| **Kyle Quest** | Architect |  |  |
| **Anatoliy Lokshin** | Development lead | 09/03/2010 | 09/03/2010 |
| **Yulia Kuchmai** | QA Representative |  |  |
|  |  |  |  |

Technical Design Review

| **Name** | **Title** | **Date Reviewed** | **Date Approved** |
| --- | --- | --- | --- |
| **Kyle Quest** | Architect |  |  |
|  |  |  |  |
|  |  |  |  |

Engineering Task Worklist Review

| **Name** | **Title** | **Date Reviewed** | **Date Approved** |
| --- | --- | --- | --- |
| **Anatoliy Lokshin** | Development lead |  |  |
|  |  |  |  |
|  |  |  |  |

References

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Author** | **Location** |
| MetraTech Security Framework Specification | Kyle Quest | <svn://qaautoserv/SecurityFramework/branches/SF1.0-Development/Docs/MtSecurityFramework.doc> |
|  |  |  |

Version Configuration

|  |  |  |
| --- | --- | --- |
| **Name** | **Version** | **Additional Comments** |
|  |  |  |
|  |  |  |

# System Overview

This document describes general interfaces for subsystems of MetraTech Security Framework. The document is intended for architects and developers.

# Assumptions and Dependencies

This document override base interfaces for all subsystems with describes on the MtSecurityFrameworkSpec document http://seceng.metratech.com/gf/download/docmanfileversion/20/78/MtSecurityFrameworkSpec.doc

# Functional Design

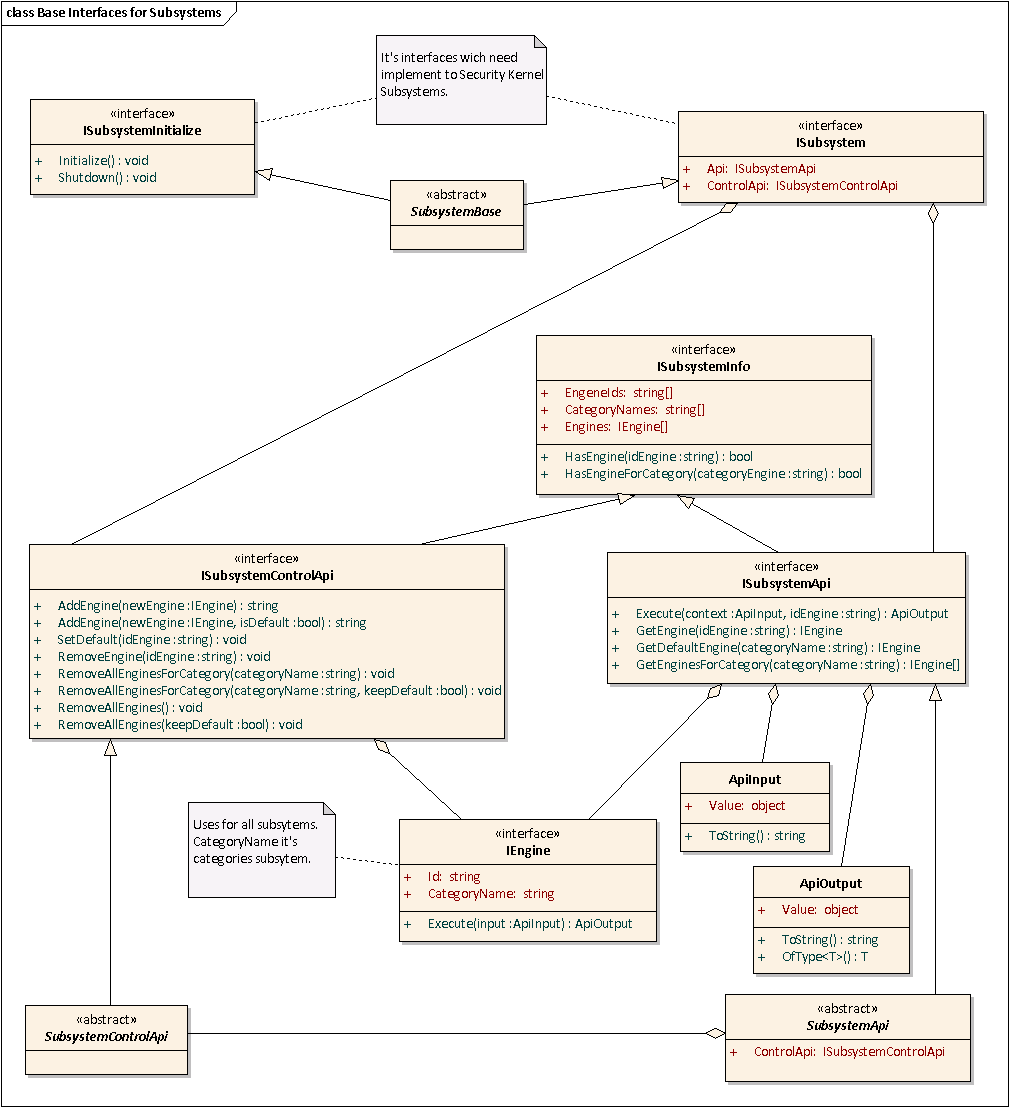
Description of the functional design.

# Technical Design

## Class Diagram

All subsystems can are described with general interfaces. Interfaces generalizing increases architecture flexibility and reduces code size. Also it will be very usefull for the Processor subsystem that will access any other subsystem via the common interface to create engine chaines.

Below is a diagram of the interfaces described by this document.



However not all subsystems can be generalized:

* **Monitoring Subsystem** – it is really a separate system used for monitoring of the Security Framework. This subsystem cannot be used within the Processor and has completely deferent public interface.
* **Random Object Generator Subsystem** – This subsystem cannot be used within the Prosessor subsystem according to the analyzy of the requirements, so it need not to be generalized.
* **Object Reference Mapper Subsytem** – As the **Random Object Generator Subsystem** it cannot be used within **Processor Subsystem.**

## Interface Description

Interfaces are described in the tables below.

**ISubsystemInitialize -** required for subsystems used by the **Processor Subsystem.** It uses for correct initializing and finalizeing of any subsystem.

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| Initialize() | Called at subsystem initializing. Uses the configuration loader to read subsystem’s settings. |
| Shutdown() | Called at subsystem finalizing. It intended for freeing resources used by the subsystem. |

**ISubsystem -** required for subsystems used by the **Processor subsystem.** It provides the interface to subsystem API.

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| Api() | Provides an access to **ISubsystemApi** interface of the subsystem. If access denied throw SubsystemApiAccessException exception. |
| ControlApi() | Provides an access to **ISubsystemControlApi** interface of the subsystem. If access denied throw SubsystemApiAccessException exception. |

**ISubsystemInfo** – provides an access to information about subsystem’s engines.

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| EngineIds: string[] | Gets a array of all subsystem’s engine ids. |
| Categories: string[] | Gets a array of all subsystem’s categories. |
| Engines: IEngine[] | Gets a array of all engines registered for the subsystem. |
| HasEngine(idEngine: string): bool | Determines whether an engine with the specified **IdEngine** was already registered for the subsystem. Returns **true** if it is and **false** otherwise. |
| HasEngineForCategory(categoryName: string): bool | Determines whether some engine was registered for the specified category. Returns **true** if it registered and **false** otherwise |

**ISubsystemApiControl** – provides an access to subsystem configuration. (the configuration specifies whether an access to this interface is allowed)

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| AddEngine(newEngine: IEngine) : string | Adds a new engine for the subsystem. First added engine become default engine for its category. It will be returned by a function ”Get default engine for category”. Returns an engine **Id** on success. |
| AddEngine(newEngine: IEngine  , isDefault: bool) : string | Adds a new engine for the subsystem. **isDefault** argument specifies wether the engine is default for its category. Default engine will be returned by a function ”Get default engine for category”. Returns an engine **Id** on success. |
| SetDefault(idEngine: string) : void | Sets an engine with **idEngene** as default for category. |
| RemoveEngine(idEngine: string) | Removes an engine with the specified **IdEngine**. Throws a **SubsystemInputParamException** exception if a engine with the specified Id not found. |
| RemoveAllEnginesForCategory(categoryName: string) | Removes all engines those belong to the specified category. Throws a **SubsystemInputParamException** exception if a specified category not found for the subsystem. |
| RemoveAllEnginesForCategory(categoryName: string, keepDefault: bool) | If **keepDefault** = **true** – remove all engines except those specified as default, If **keepDefault** = **false** - removes all engines those belong to the specified category. Throws a **SubsystemInputParamException** exception if a specified category not found for the subsystem. |
| RemoveAllEngine() | Removes all endgines from the subsystem. |
| RemoveAllEngine(keepDefault: bool) | If **keepDefault** = **true** – remove all engines from the subsystem except those specified as default, If **keepDefault** = **false** - removes all endgines from the subsystem. |

**ISubsystemApi** – provides an access to configured subsystem engines. (the configuration specifies whether an access to this interface is allowed)

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| Execute(value: ApiInput, idEngine: string) : ApiOutput | Executes an engine with the specified **IdEngine**. The **value** argument is a value to act on it. Returns an **ApiOutut** object on success. Throws a **SubsystemInputParamException** exception if an engine with the specified **IdEngine** not found. Throws a **BadInputDataException** exception when some security problem found. |
| GetEngine(idEngine: string) : IEngine | Gets Engine by **idEngine**. If **idEngine** does not exist throw **SubsystemInputParamException** exception. |
| GetDefaultEngine(categoryName: string) : IEngine | Get default Engine for a category specified by **categoryName**.  If category does not have any throw **SubsystemInputParamException** exception.  If category doesn’t exist throw **SubsystemInputParamException** |
| GetEnginesForCategory(categoryName: string) : IEngine[] | Get all Engines by **categoryName**.  If category doesn’t exist throw **SubsystemInputParamException** |

**IEngine** – base interface for any engines that should be used into **Processor Subsystem.**

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| Id : string | Engine identifier within the subsystem. Any format for unique key |
| CategoryName : string | Gets the name of category the engine belongs to. |
| Execute(input : ApiInput) : ApiOutput | Performs an action on a input value. If the action executed without any errors returns **ApiOutput** and throws an exception otherwise. |

**ApiInput** – class with used for input parameters into engine

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| Value : object | A value represented by the object. |
| ToString() : string | Converts object to string |

**ApiOutput**– class with used for get of output parameters engine

|  |  |
| --- | --- |
| **Properties/Methods** | **Description** |
| Value : object | A value represented by the object. |
| ToString() : string | Converts object to string |
| OfType<T>() : T | Converts the Value to a specified type. |

SystemControlApli – inmlemented ISystemControlApli interface. That class can use into any subsystems.

SystemApli – inmlemented ISystemApli interface and agregate ISystemControlApli to provide access to the angines collection. That class can use into any subsystems.

## UI Design

## High-Level Design / Tasks

## Database Schema Changes

# Error list

The following table shows a listing of errors that can occur.

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Code** | **Error Message** | **Description** | **Area** |
|  |  |  |  |
|  |  |  |  |

# Outstanding Issues

List all open issues regarding this document.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Date raised** | **Description and Resolution** | **Page/ Section** | **Raised by** | **Allocated to** | **Status** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |