PILAR

Library Extensions

24.8.2009

1 Introduction

Libraries establish a reference framework for risk projects. Users will focus on their components, and still different systems may be compared.

There is a clear distinction between *the library manager*, and the tool user. The library is closed. The projects use the library.

A library provides:

- a collection of asset classes (extensible)
- a collection of security dimensions
- a collection of qualitative levels
- a collection of encoded valuation criteria (extensible)
- a collection of threats (extensible)
- a collection of safeguards (extensible)
- typical security procedures

A library also provides:

- standardised criteria for valuing assets (extensible)
- typical threats per asset class (modifiable)
- typical threat likelihood values (modifiable)
- typical degradation levels (modifiable)
- knowledge on the relative importance of safeguards

This document show how to:

- incorporate new classes of assets
- incorporate new criteria for valuation
- incorporate new threats

New elements are regarded as refinements with respect to old ones, in such a way that

- documentation is improved
- the analysis can be tailored to a precise description

... but the general features of the tools are not lose.

2 Naming and location

Extension files have the extension

.lle

and are loaded from the library directory, typically

Program Files / PILAR_version / bib_en /

When there are several files, the files are sorted by filename before loading.

3 New classes of assets

Extension files for new classes of assets are written in XML notation using the following syntax

```
 1  file ::=
 2    <library-extension>
 3      { classes }0+
 4    </library-extension>
 5
 6  classes ::=
 7    <classes [ under="..." ] >
 8      { class }0+
 9    </classes>
10
11  class ::=
12    <class code="..."> NAME </class>
```

Each new class has a code and a name. The name is given in line 12. There are two options for the code

- if there is an attribute “under” in line 7, the attribute “under” and the attribute “code” in line 12 are concatenated to build the full code of the new class
- if there is no attribute “under”, the code is the one given in line 12

In the first case, the class identified by “under” must be already defined.

In any case, the code of the class is the full path, and the new code is located under the class that has a prefix code. Prefixes are created as needed.

It is easier to understand with an example that creates several classes:

```
<library-extension>

  <classes>
    <class code="a.b">class b under a</class>
    <class code="a.c">class c under a</class>
  </classes>

  <classes under="L">
    <class code="b">class b under L (locations)</class>
    <class code="c">class c under L (locations)</class>
  </classes>

</library-extension>
```

4 New criteria for valuation

Extension files for new criteria are written in XML notation using the following syntax

```
 1  file ::=
 2    <library-extension>
 3      { criteria }0+
 4    </library-extension>
 5
 6  criteria ::=
 7    <criteria under="..." >
 8      { reason }0+
 9    </criteria>
10
11  reason ::=
12    <reason code="..."> NAME </reason>
```

Each new criterion has a code (given in line 12), and a name (given in line 12). The new criterion is allocated under the criterion given in line 7, that must exist.

The code in line 12 may start with a dot (“.”), then the full code is the concatenation of the attribute “under” and the attribute “code”.

The following example

- defines 5 personalized levels, allocated under standard PILAR major headings
- defines 5 criteria for confidentiality, under each of the personalized levels
- for instance, there is a criterion coded [my-2.C]; that is my confidentiality criteria for level 2

```
<library-extension>

  <criteria under="7">
    <reason code="my-5">My Organisation - Level 5</reason>
  </criteria>
  <criteria under="5">
    <reason code="my-4">My Organisation - Level 4</reason>
  </criteria>
  <criteria under="3">
    <reason code="my-3">My Organisation - Level 3</reason>
  </criteria>
  <criteria under="2">
    <reason code="my-2">My Organisation - Level 2</reason>
  </criteria>
  <criteria under="0">
    <reason code="my-1">My Organisation - Level 1</reason>
  </criteria>

  <criteria under="my-5">
    <reason code=".C">
      CONFIDENTIAL:
      Information regarded as highly important and critical.
      Only a few number of people is allowed to know it.
```

```
    </reason>
  </criteria>

  <criteria under="my-4">
    <reason code=".C">
      RESTRICTED:
      Its revelation to unauthorised people
      would cause severe economic problems, or bad publicity.
    </reason>
  </criteria>

  <criteria under="my-3">
    <reason code=".C">
      RESTRICTED:
      Its revelation to unauthorised people
      would cause economic problems, bad publicity,
      or lead to a breach of a legal obligation..
    </reason>
  </criteria>

  <criteria under="my-2">
    <reason code=".C">
      INTERNAL:
      Information for internal use only:
      it is not expected to be outside the organisation.
    </reason>
  </criteria>

  <criteria under="my-1">
    <reason code=".C">
      PUBLIC:
      Information any one can access to (e.g. web site).
    </reason>
  </criteria>

</library-extension>
```

5 New threats

Extension files for new threats are written in XML notation using the following syntax

```
 1  file ::=
 2    <library-extension>
 3      { threats }0+
 4    </library-extension>
 5
 6  threats ::=
 7    <threats under="..." >
 8      { threat }0+
 9    </threats>
10
11  threat ::=
12    <threat code="..."> NAME </threat>
```

Each new threat has a code (given in line 12), and a name (given in line 12). The new threat is allocated under the threat given in line 7, that must exist.

The code in line 12 may start with a dot (“.”), then the full code is the concatenation of the attribute “under” and the attribute “code”.

The following example

- defines several threats under the standard [N.*];
  the codes are [N.*.1], ...
- defines several threats under the standard [A.26];
  the codes are [beat], ...

```
<library-extension>

  <threats under="N.*">
    <threat code=".1">Storms</threat>
    <threat code=".2">Thunderstorms and Lightning</threat>
    <threat code=".3">Hurricanes</threat>
    <threat code=".4">Earthquakes</threat>
    <threat code=".5">Tornadoes</threat>
    <threat code=".6">Cyclones</threat>
    <threat code=".7">Landslide and mudslide</threat>
    <threat code=".8">Meteorites</threat>
    <threat code=".9">Tsunamis</threat>
    <threat code=".10">Winter storms and extreme cold</threat>
    <threat code=".11">Extreme heat</threat>
    <threat code=".12">Volcanoes</threat>
  </threats>

  <threats under="A.26">
    <threat code="brute">Beating / blowing</threat>
    <threat code="bomb">Bomb</threat>
    <threat code="terror">Terrorism</threat>
  </threats>

</library-extension>
```

6 XML notation

The XML syntax is presented using a variant of BNF notation, namely:

notation	meaning
`{ x }0+`	stands for zero or more occurrences of “x”
`[ x ]`	stands for zero or one occurrence of “x”; that is, “x” is optional