THOR v3.0 – Traitement Hybride pour l’Optimisation du Rapport

*Documentation Utilisateur*

Table des matières

[1 Présentation 2](#_Toc52807438)

[1.1 Constat 2](#_Toc52807439)

[1.2 Besoin 2](#_Toc52807440)

[2 Mise en place 2](#_Toc52807441)

[2.1 Prérequis 2](#_Toc52807442)

[2.1.1 Windows 2](#_Toc52807443)

[2.1.2 Linux 2](#_Toc52807444)

[2.1.3 Modules python a installé avec PIP 2](#_Toc52807445)

[2.2 Exécution 3](#_Toc52807446)

[2.2.1 Windows 3](#_Toc52807447)

[2.2.2 Linux 5](#_Toc52807448)

[3 Configuration de THOR 5](#_Toc52807449)

[3.1 Niveau 1 5](#_Toc52807450)

[3.2 Niveau 2 6](#_Toc52807451)

[3.3 Le niveau 3 6](#_Toc52807452)

[3.3.1 Niveau 4 6](#_Toc52807453)

[4 Utilisation 8](#_Toc52807454)

[4.1 Les tableaux Excel 8](#_Toc52807455)

[4.2 Les échelles 8](#_Toc52807456)

[4.3 Le modèle de rapport Word 8](#_Toc52807457)

[4.3.1 La mise en page 8](#_Toc52807458)

[4.3.2 La gestion des tableaux 8](#_Toc52807459)

[4.4 L’interface graphique 8](#_Toc52807460)

[4.5 La génération du rapport et la journalisation 10](#_Toc52807461)

# Présentation

## Constat

Lors de la réalisation d’une étude de risque avec la méthode EBIOS RM, l’utilisation d’une application dédiée facilite la production des différents livrables au format  Excel et images mais ne permet pas de réaliser un modèle de rapport personnalisé.

## Besoin

Notre retour d’expérience montre qu’il est difficile d’échanger avec un client en utilisant des fichiers Excel, il est plus aisé d’utiliser un fichier Word contenant des éléments de contexte associés à chaque tableau et permettant d’utiliser le mode révision pour le suivi des modifications ainsi que les commentaires. Il a donc fallu trouver une solution permettant d’associer les fichiers Excel issus de l’application et un modèle de rapport Word.

THOR est un script, réalisé en python 3, prenant en entrée un document Word ainsi qu’un fichier de configuration (en JSON) permettant d’associer chaque tableau du document Word à un fichier Excel.

THOR va alors copier chaque tableau Excel dans le document Word en conservant la mise en forme du tableau Excel. Les entêtes sont ignorés afin de pouvoir dissocier les entêtes du tableau Word de celles du tableau Excel issues de l’application et qui ne sont pas personnalisables. De ce fait, le nombre et l’ordre des colonnes entre les tableaux Word et Excel doivent correspondre.

# Mise en place

## Prérequis

THOR étant un script python il est cross-plateforme et fonctionne donc aussi bien sous Windows que sous Linux. Il nécessite en revanche l’installation de certains modules python. En fonction des l’installation de base dont vous disposez, certaines dépendances peuvent ne pas être listées *infra*.

### Windows

* PYTHON 3.8 ou supérieur avec le module TCL et pylauncher activés lors de l’installation

### Linux

* PYTHON 3.8 ou supérieur
* PYTHON3-TK
* PYTHON-PIP
* Interface graphique

### Modules python a installé avec PIP

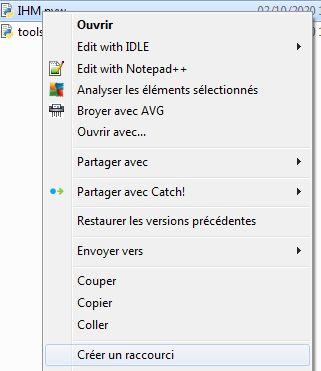
* PYTHON-DOCX => Gestion des fichiers DOCX
* XLRD => gestion des fichiers XLS
* COLORAMA = gestion des couleurs
* PYYAML
* DOCXTPL => gestion de templates word

## Exécution

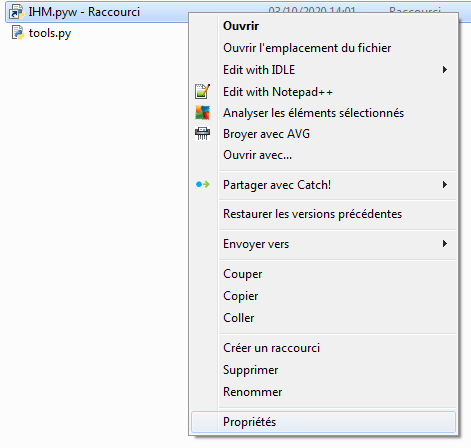
THOR peut être placé n’importe où dans l’arborescence de fichier. En revanche, le répertoire courant est utilisé dans THOR comme répertoire de base dans les fenêtres d’exploration de fichier. Si ce n’est pas nécessaire à son exécution, il peut être utile de créer un raccourci personnalisé placé dans le répertoire du projet (rapport + fichiers Excel) afin de faciliter l’utilisation du script. Une fois créé, le raccourci peut être déplacé dans le répertoire du projet.

### Windows

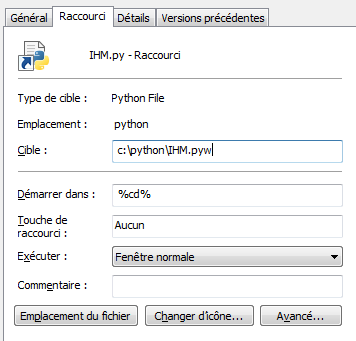
Dans un premier temps, créer un raccourci sur le fichier IHM.Pyw



Ensuite il faudra le personnaliser pour qu’il ouvre THOR dans le répertoire courant et non le répertoire d’installation



Il faut remplacer le répertoire d’exécution en y plaçant la variable d’environnement ***%cd%*** contenant le répertoire courant



### Linux

Sous Linux, il est alors possible de lancer le script depuis n’importe quel répertoire. Afin de se faciliter le travail il est possible de créer un lien symbolique dans le répertoire du projet qui pointe vers le chemin complet du script THOR.

Par example:

*# python3 /home/python/IHM.pyw*

# Configuration de THOR

Toute la configuration de THOR se fait dans le fichier « *THEME.YAML »*.

Ce fichier contient :

* La version de THOR
* L’activation ou non du mode debug
* La déclaration des légendes
* La déclaration des tableaux word
* La personnalisation éventuelle des tableaux word

Son format est en YAML est se construit sur 4 niveaux comme :

version: 2.0

debug: false

modeles:

Scénario\_stratégique: # Ne pas supprimer

type: "generique"

extension: "xls"

…..

echelles:

"Echelles de valeurs":

'Définir les echelles de valeurs':

gravite:

…..

tableaux:

"Atelier 1":

'Définir le Périmètre Métier et Technique':

Valeurs\_Metiers:

…..

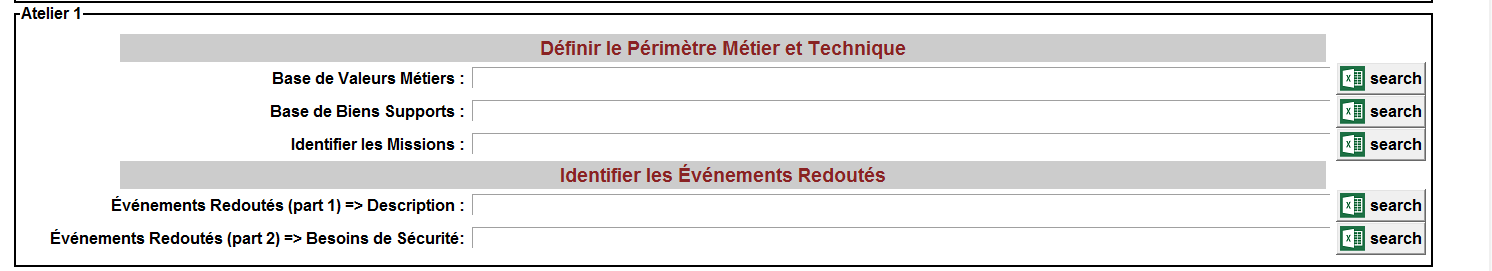
## Niveau 1

Si l’on omet la version et le mode debug, le premier niveau ne contient que 2 parties :

* Modeles : spécifications génériques pour les traitements dynamiques. Dans le cas ci-dessus il s’agit des scenarios stratégiques dont le nombre n’est pas figé. Les options possibles sont le mêmes que pour un tableau.
* Echelles : c’est là que sont déclarées les échelles de base de valeurs
* Tableaux : c’est là que sont déclarer les tableaux du document Word

## Niveau 2 (échelles et tableaux)

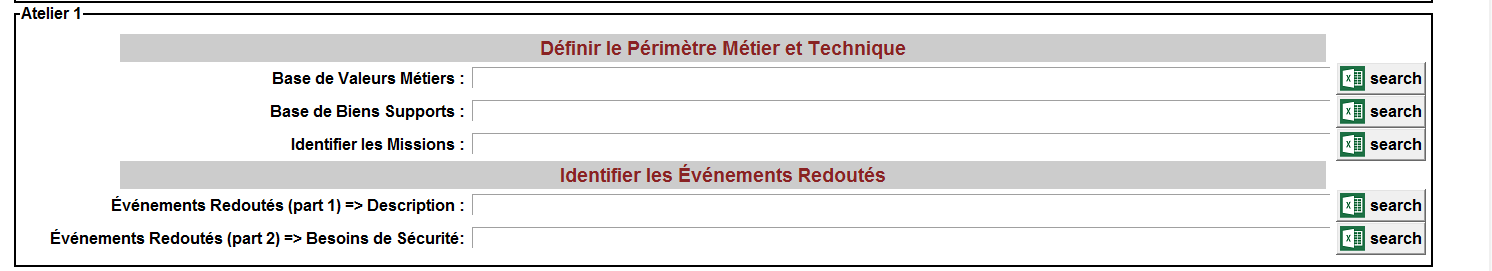
Le niveau 2 ne sert que pour la mise en page de l’interface graphique afin de regrouper les tableaux par atelier. Par exemple pour le niveau « Atelier 1 », le texte se retrouve dans l’interface graphique :



## Le niveau 3 (échelles et tableaux)

Comme pour le niveau 3, le niveau 3 correspond au second regroupement des tableaux qui peut être vu comme un sous-atelier dans l’application « Agile Risk Manager ».

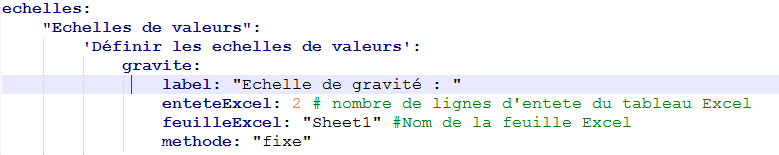
Par exemple pour la valeur 'Définir le Périmètre Métier et Technique' de l’atelier 1 on le retrouve dans l’interface graphique.



### Niveau 4 (échelles et tableaux)

Le niveau 4 correspond à la déclaration d’un tableau ou d’une échelle :

#### Pour une échelle



Dans l’exemple ci-dessus, nous avons déclaré une échelle de base « gravite ». Celle-ci contient les clés suivantes :

**Label** : C’est le label qui sera affiché dans l’interface graphique pour le champ correspondant à ce tableau

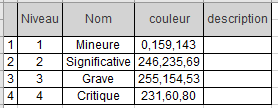
**enteteExcel** : c’est le nombre de ligne que contient l’entête du tableau dans le fichier Excel.

**feuilleExcel** : c’est le nom de la feuille de calcul dans le fichier Excel

**méthode** : ce champ peut contenir 2 valeurs :

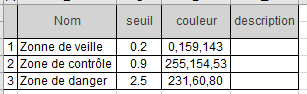
* **fixe** : La valeur est fixe, à une valeur donnée correspond une couleur.

Le format du fichier Excel est figé est doit être comme suit :



Couleur au format RGB

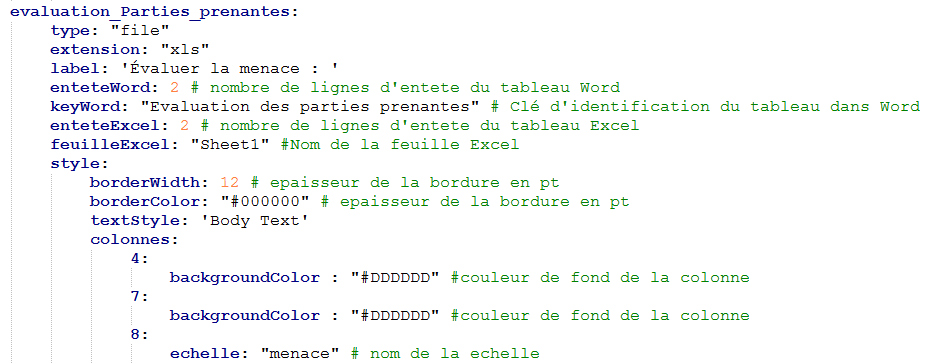
* **calculée** : la valeur est variable, une couleur correspond à une ensemble de valeur entre 2 seuils. Le format du fichier Excel est figé est doit être comme suit :



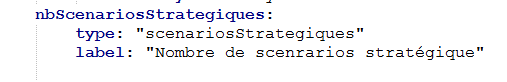
Couleur au format RGB

#### Pour un tableau

Pour un tableau, il y a beaucoup plus de valeurs possibles. Voici un exemple pour le tableau *evaluation\_Parties\_prenantes* déclaré ci-dessous :



**Type :** c’est le type de données attendue :

* **File :** on attend le chemin d’un fichier, associé au champ extension
* **Image :** on attend le chemin d’une image
* **scenariosStrategiques :** type particulier identifiant un entier représentant le nombre de scenarios stratégique pour personnaliser l’interface graphique :

**Extension :** extension de fichier attendu, utilisé pour le type image ou file

**Label** : C’est le label qui sera affiché dans l’interface graphique pour le champ correspondant à ce tableau

**enteteWord**: c’est le nombre de ligne que contient l’entête du tableau dans le rapport Word.

**keyWord** : c’est la clé utilisée dans le rapport Word pour identifier le tableau

**enteteExcel** : c’est le nombre de ligne que contient l’entête du tableau dans le fichier Excel.

**feuilleExcel** : c’est le nom de la feuille de calcul dans le fichier Excel

**style** : toutes les options se rapportant à la mise en forme du tableau dans le rapport Word :

**borderWidth** : taille des bordures des cellules du tableau

**borderColor** : couleur des bordures au format hexadécimal

**textStyle** : style du rapport Word à utiliser pour les cellule du tableau

**colonnes** : optionnel, permet de personnaliser les colonnes du tableau

* 0 : c’est l’indice de la colonne, commence à 0 pour la première colonne
  + **backgroundColor** : couleur de fond de la cellule au format hexadécimal
  + **echelle** : nom de l’échelle de valeur à utiliser pour cette colonne
  + **aligment** : permet de spécifier un alignement pour le texte au sein de la cellule, les valeurs possibles sont : *left, center, right, justify*

# Utilisation

## Les tableaux Excel

Les tableaux Excel sont exportés, par conception de la logique de ce script ils ne doivent pas être modifiés. Il est néanmoins possible de les créer manuellement si besoin. Certaines mises en forme sont conservées lors de l’import sous Word :

* Les caractères en gras
* Les caractères en italique
* Les caractères soulignés
* La couleur des polices.
* Les entêtes sont ignorés, pour autant le nombre de lignes d’entête est fixé dans le fichier de configuration de THOR « THEME.YAML » pour correspondre à notre besoin.

**ATTENTION : *Dans le format des tableaux Excel exportés, la première colonne représente l’ID de la ligne. N’étant pas utilisée dans le rapport WORD, la première colonne des fichiers Excel est ignorée. À prendre en compte lors de la création manuelle de fichiers Excel.***

## Les échelles

Les échelles doivent se trouver sous forme de fichier Excel.

**TIP :** Elles sont déclarées dans le fichier de configuration de THOR *« THEME.YAML »*

Dans le fichier THEME.YAML, lorsqu’une échelle est spécifiée pour une colonne d’un tableau, alors si la valeur des cellules de la colonne matche avec celles de l’échelle, la couleur associée de l’échelle sera appliquée à la couleur de fond de la cellule du rapport Word.

## Le modèle de rapport Word

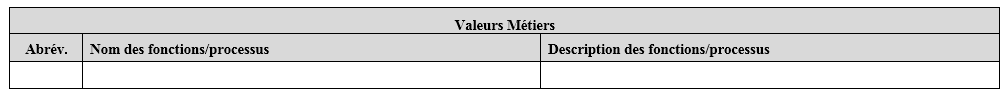
### La mise en page

La mise en page du rapport Word est libre, Tout l’avantage de THOR réside ici, il n’est pas tributaire du contenu ni de la forme du rapport. Il conserve l’intégrité du rapport et ne remplace que le contenu des tableaux identifiés.

**TIP :** *la liste des tableaux est déclarée dans le fichier de configuration de THOR « THEME.YAML »*

### La gestion des tableaux

Afin de pouvoir identifier les tableaux dans le document Word, la première cellule du tableau doit contenir une valeur clé. Afin de ne pas perturber la mise en forme, nous avons fait le choix de considérer la première ligne du tableau comme un titre, la première ligne du tableau ne contient qu’une seule cellule fusionnée sur la totalité du tableau.



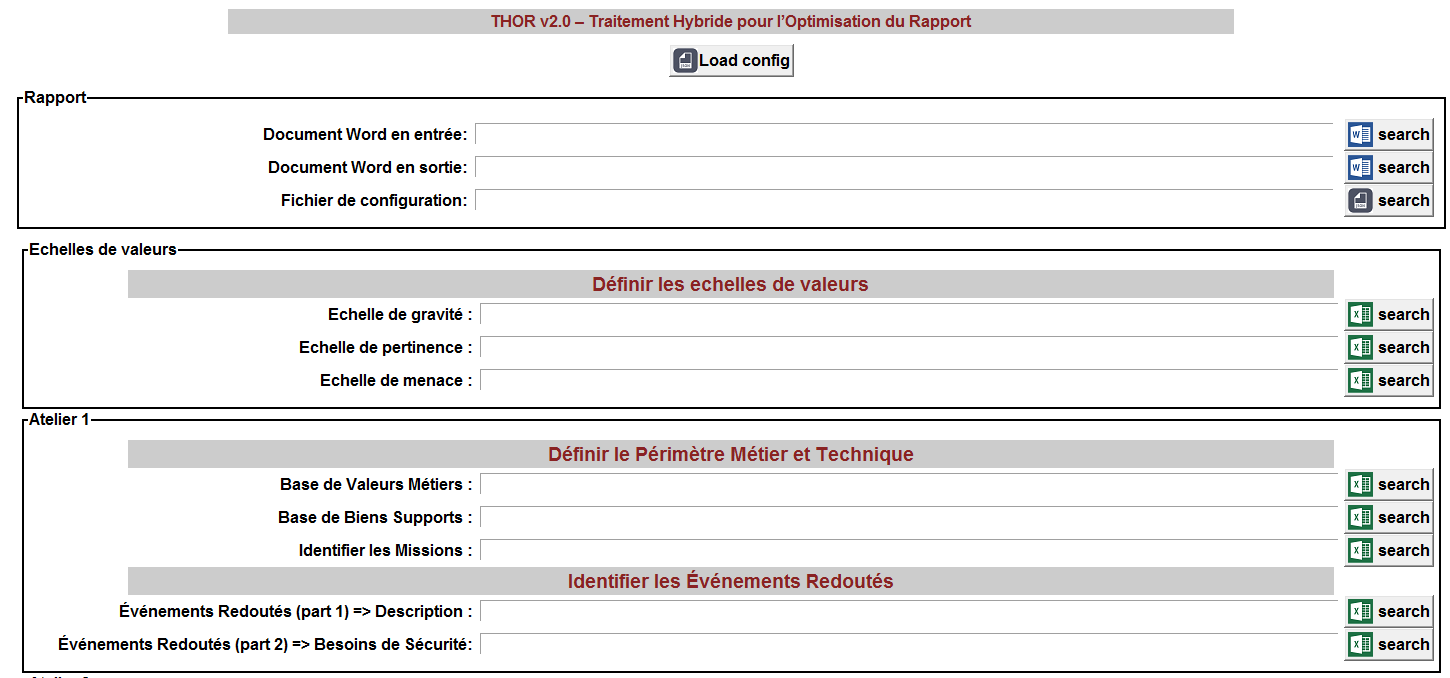
Dans le cas ci-dessus, le titre « Valeur Métiers » est la clé utilisée par le script pour identifier le tableau.

**TIP** : *La liste des tableaux et les clés sont déclarées dans le fichier de configuration de THOR « THEME.YAML ».*

## L’interface graphique

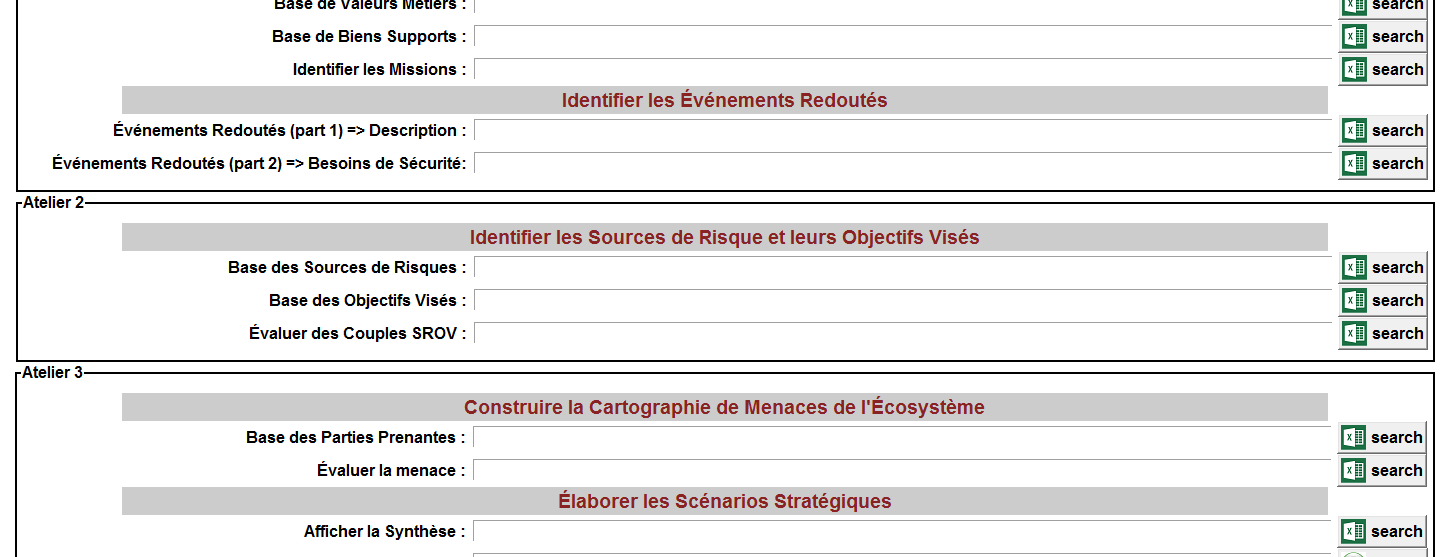
Lorsque vous exécuter le script, une interface graphique s’ouvre, celle-ci contient plusieurs parties réparties en fonction des ateliers. Le bouton « search » en fin de ligne permet d’ouvrir la fenêtre d’exploration de fichier afin de sélectionner un fichier.

**TIP** : *Le but de cette interface graphique est d’associer des fichiers Excel aux tableaux et légendes déclarés dans le fichier THEME.YAML*

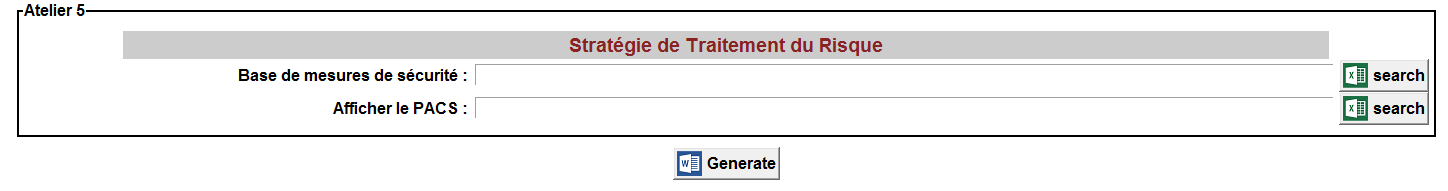


La première partie contient les information d’entrée/sortie du script :

* **Document Word en entrée** : Lien vers le fichier Word originel, lors de la première exécution il est conseillé d’utiliser le modèle puis de reprendre la dernière version du rapport lors des exécutions suivantes.
* **Document Word en sortie**: Lien vers le fichier de sortie. C’est ce fichier qui contiendra le rapport généré.
* **Fichier de configuration**: fichier de sauvegarde de la configuration actuelle pour les exécutions suivantes. Il est possible de charger une configuration existante grâce au bouton « load config » en haut de la fenêtre



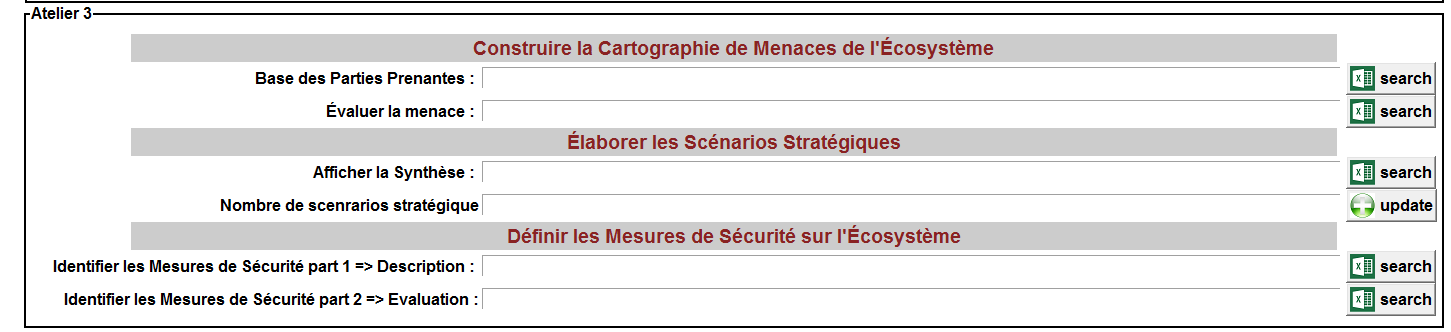
Les parties suivantes contiennent tous les tableaux correspondant aux étapes de la méthode EBIOS RM. Nous avons essayé de conserver ces étapes afin de faciliter l’identification des tableaux correspondant. Celles-ci sont déclarées dans le fichier *THEME.YAML*



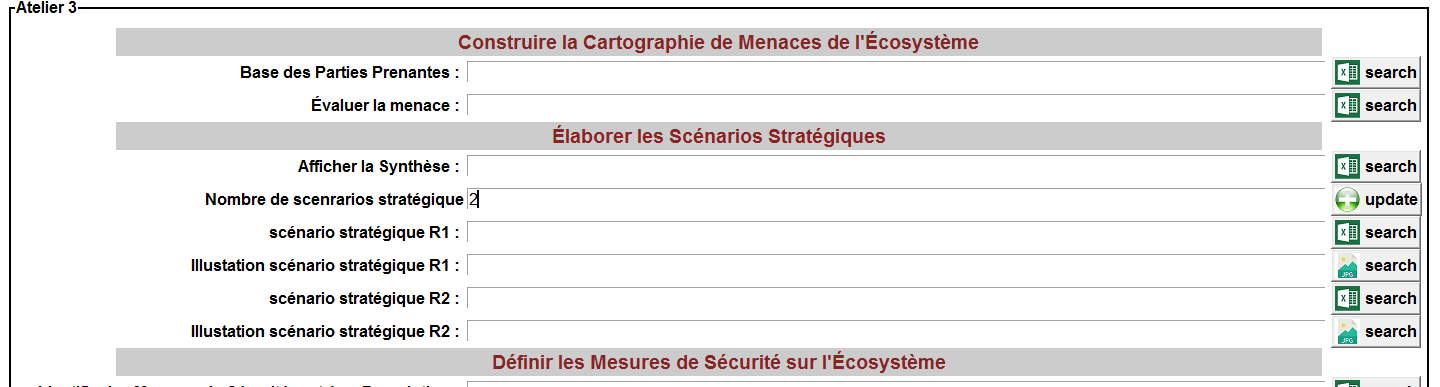
Le bas de la fenêtre contient le bouton « Generate » permettant de lancer la génération du rapport et la sauvegarde de la configuration.

## Les scenarios dynamiques

Le nombre de scénarios stratégiques n’étant pas connu, il est nécessaire de pourvoir ajouter dynamiquement des scenarios. A cette fin, l’atelier 3 contient une ligne nbScenariosStratégiques permettant de définir le nombre de scenarios stratégique. Le bouton « *Update* » correspondant permet de créer les lignes correspondantes. Un scenario stratégique se compose d’un tableau et d’une image. LA mise en forme de ces tableaux se trouve dans les modèles du fichier YAML.

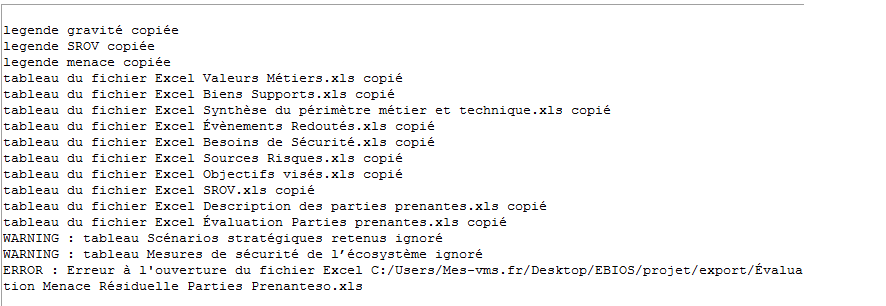


Après mise à jour :



## La génération du rapport et la journalisation

En bas de la fenêtre, se trouve un champ contenant la journalisation des événements. Puiseurs valeurs sont possibles :

* Légende xxx copiée : La légende a été trouvée dans le rapport :
* Tableau xxx copié : le tableau Excel correspondant a été copié avec succé dans le rapport
* WARNING : Tableau xxx ignoré : Lorsqu’un champ de l’interface graphique est vide, le tableau correspondant est ignoré
* ERROR : xxxx : Une erreur est survenue dans le traitement du rapport.
* 

A la fin du traitement, un popup vous affiche un message :

* La génération du rapport est terminée avec succès : Par d’erreur, voir la journalisation pour d’éventuels tableaux ignorés
* La génération du rapport est terminée avec X erreurs : Des erreurs sont survenues, reportez-vous à la journalisation pour identifier les erreurs.

