

Manuel d'installation et d'exploitation des Plx

Réf. POM3-CS-2021-MIEX-PIX-007

	Nom	Société	Fonction	Date	Visa
Rédigé par :	M .Renon	CS Group	Équipe projet		
Validé par :	C. Mertz	CS Group	Chef de projet		
Pour application :	J. Covès	CS Group	Directeur de projet		

CS GROUP
6 rue Brindejonc des Moulinais
Parc de la Grande Plaine
BP 15872
31506 Toulouse Cedex 5

ED.	RÉV.	DATE	MOTIF
01	00	02/02/18	Création du document
01	01	09/02/18	Serveur SSH
01	02	13/02/18	Activation de l'environnement virtuel sous Windows
01	03	09/04/18	Prise en compte des remarques et légères corrections.
01	04	30/08/18	Précisions pour les plateformes de type Windows sous Cygwin.
01	05	03/09/2018	Prise en compte des remarques
01	06	04/09/2018	Ajout du chapitre PIT à compléter.
01	07	05/06/19	Schapi : Modification des chapitres PI et PIG suite à la livraison du PIG2.1 Modification du chapitre PIPT suite à la livraison de la VM 2019
02	03	06/08/20	Modification pour le PIG3 / GRP 2018 et la livraison d'une VM dédiée basée sur Debian 10
02	04	20/10/20	Ajout des compatibilités avec le PI 3.1 Modification pour le PIPT3 : installation dans un environnement virtuel Python
02	05	15/08/21	Relecture et modification de la partie Pipt
02	06	11/10/21	Relecture Schapi par rapport aux dépendances numpy et pandas
03	00	25/10/2021	§4 PIG : [issue #8] GRP 2020
03	01	18/03/2022	§1.4 - §7 PIM : [issue #2] Nouvelle VM Mascaret [issue #1] Initialisation "rampe" [issue #4] Sauvegarde ligne d'eau via API [issue #5] Casiers et différents noyaux [issue #9] Décalage vertical en m (en entrée et en sortie)
03	02	01/04/2022	§7.1.4 correction exécutable PIM à lancer par la plateforme
03	03	09/06/2022	§1.4 Maj PIG 3.2, PI 3.2 avec libbdimage 1.5.8
03	04	15/12/2022	§2.5 Mise à jour sécurité pour VM Debian Mise à jour PIT sur VM Debian 10 Mascaret
04	00	05/05/2023	Passage à PI 4.0.0, PIM 4.0.0, PIPT 4.0.0
04	01	30/05/2023	Ajout warning pour la clé « [models]directory », cf §3.2.3
04	02	06/06/2023	Passage à PIG 4.0.0 ; Ajout PIFG 4.0.0
04	03	16/06/2023	§2.3.1 : Gestion de la VM SCHAPI Debian 11 pour l'installation des utilitaires Python
04	04	30/06/2023	§6 PIT : passage à PI 4.0.0
04	05	04/07/2023	§8 PIFG

			<ul style="list-style-type: none"> - ajout de la procédure de recompilation §8.1.4 - compléments pour §8.1.1, §8.2.1, §8.2.3, §8.2.4
04	06	20/09/2023	<p>§3.2.3 Compléments sur le fonctionnement de l'outil 'checkini' pour les clés du PIT</p> <p>§3.2.5 Informations tracées dans les logs sur les calculs prévisions effectuées en parallèle</p>

Table des matières

1. Généralités.....	14
1.1 Objet du document.....	14
1.2 Principe général.....	14
1.3 Protocole POM.....	15
1.3.1 Présentation.....	15
1.3.2 Arborescence des fichiers POM.....	15
1.3.3 Test de compatibilité.....	16
1.4 Correspondances.....	16
2. Installation des dépendances.....	19
2.1 Serveur SSH.....	19
2.1.1 Généralités.....	19
2.1.2 CopSSH pour Windows.....	19
2.1.3 Installation de Cygwin pour Windows.....	19
2.1.3.1 Prérequis.....	19
2.1.3.2 Installation.....	20
2.1.3.3 Configuration du serveur ssh.....	21
2.1.3.3.1 Modification de l'environnement.....	21
2.1.3.3.2 Déclaration des utilisateurs et groupes.....	21
2.1.3.3.3 Configurer le serveur SSH.....	22
2.1.3.4 Lancer le serveur SSH.....	23
2.1.3.5 Tester le serveur SSH.....	23
2.1.3.6 Coté POM.....	24
2.2 Installation de Python.....	28
2.2.1 Sous Windows.....	28
2.2.1.1 Miniconda.....	28
2.2.1.2 Paramétrer le proxy.....	28
2.3 Installation des utilitaires Python.....	29
2.3.1 Pour VM modèle SCHAPI Debian 11.....	29
2.3.1.1 Installation de pip.....	29
2.3.1.2 Installation des virtualenvs.....	30
2.3.2 Spécificités pour les VMs modèles Plathynes, Mascaret et GRP.....	31
2.3.2.1 Modification du fichier .bashrc pour les VM PLATHYNES.....	31
2.3.2.2 Modification du fichier .bashrc pour les VM MASCARET.....	31
2.3.2.3 Modification du fichier .profile pour les VM GRP.....	31
2.3.3 Sous Windows.....	32
2.4 Création et activation d'un environnement virtuel.....	32
2.4.1 Sous Linux.....	32
2.4.2 Sous Windows.....	33
2.5 Mise à jour sécurité pour VM Debian.....	33
3. PI.....	35
3.1 Installation.....	35
3.1.1 Sous Linux.....	35
3.1.2 Sous Windows.....	36
3.2 Exploitation.....	37
3.2.1 Fichier « .ini ».....	37

3.2.2	Calculs concurrents.....	39
3.2.3	Outil de vérification des fichiers .ini.....	40
3.2.4	Gestion des fichiers .toml.....	41
3.2.5	Gestion des calculs parallèles en prévision.....	41
3.3	FAQ (Foire Aux Questions).....	43
4.	PIG.....	45
4.1	Installation.....	45
4.1.1	Système d'exploitation.....	45
4.1.1.1	Caractéristiques de la VM.....	45
4.1.1.2	Utilisateurs et groupes.....	45
4.1.1.3	Contenu de la VM.....	45
4.1.2	Pré-requis.....	46
4.1.3	Procédure d'installation.....	46
4.1.4	Configuration à faire sur la POM.....	48
4.2	Exploitation.....	48
4.2.1	Mode de calcul.....	48
4.2.2	Fichier « .ini ».....	50
4.2.3	Ligne de commande.....	51
4.2.4	Calculs concurrents.....	51
4.3	FAQ (Foire Aux Questions).....	52
5.	PIPt.....	53
5.1	Installation du PIPt.....	53
5.1.1	Système d'exploitation.....	53
5.1.2	Pré-requis.....	53
5.1.3	Procédure d'installation.....	53
5.1.4	Configuration à faire sur la POM.....	54
5.1.5	Configuration pour les fichiers « .ini ».....	54
5.2	Installation du solveur PLATHYNES.....	55
5.2.1	Prérequis – Système d'exploitation.....	55
5.2.1.1	Caractéristiques de la VM PLATHYNES.....	56
5.2.1.2	Utilisateurs et groupes.....	56
5.2.3	Solveur PLATHYNES.....	56
5.2.3.1	Contenu de l'archive du noyau.....	56
5.2.3.2	Installation.....	57
5.3	Exploitation.....	58
5.3.1	Dépôts des exports PLATHYNES.....	58
5.3.2	Conseils pour le codage des modèles PLATHYNES.....	59
5.3.3	Arborescence des fichiers.....	59
5.3.4	Mode de calcul.....	60
5.3.5	Fichier « .ini ».....	61
5.3.6	Calculs concurrents.....	62
5.3.7	Ménage automatique.....	62
5.4	FAQ (Foire Aux Questions).....	62
6.	PIT.....	63
6.1	Installation.....	63
6.1.1	Pré-requis.....	63
6.1.2	Procédure d'installation.....	63

6.1.3 Configuration à faire sur la POM.....	65
6.1.4 Mise à jour des variables d'environnement.....	65
6.1.5 Mise à jour des modèles pour Telemac V8P4.....	66
6.2 Exploitation.....	66
6.2.1 Arborescence des fichiers.....	66
6.2.2 Mode de calcul.....	67
6.2.3 Fichier « .ini ».....	68
6.2.4 Calculs concurrents.....	69
6.2.5 Ménage automatique.....	70
7. PIM.....	71
7.1 Installation.....	71
7.1.1 Système d'exploitation.....	71
7.1.1.1 Caractéristiques de la VM.....	71
7.1.1.2 Utilisateurs et groupes.....	71
7.1.1.3 Contenu de la VM.....	71
7.1.2 Pré-requis.....	72
7.1.3 Procédure d'installation.....	72
7.1.4 Configuration à faire sur la POM.....	73
7.2 Exploitation.....	74
7.2.1 Arborescence des fichiers.....	74
7.2.2 Mode de calcul.....	75
7.2.3 Fichier « .ini ».....	76
7.2.4 Calculs concurrents.....	78
7.2.5 Ménage automatique.....	78
7.2.6 Conseils pour le portage des modèles de la VM DAMP.....	79
8. PIFG.....	81
8.1 Installation.....	81
8.1.1 Système d'exploitation.....	81
8.1.2 Pré-requis.....	81
8.1.3 Procédure d'installation.....	81
8.1.4 Procédure de recompilation de l'outil de conversion.....	83
8.1.5 Configuration à faire sur la POM.....	83
8.1.6 Ligne de commande.....	84
8.2 Exploitation.....	84
8.2.1 Arborescence des fichiers.....	84
8.2.2 Mode de calcul.....	84
8.2.3 Fichier « .ini ».....	85
8.2.4 Calculs concurrents.....	85

Liste des tableaux

Tableau 1: correspondances POM – Pix – plateformes de modélisation	18
Tableau 2: fichier « .ini » du PI	39
Tableau 3: fichier « .ini » du PI - « Cleaners »	39

Tableau 4: modes de calcul du PIG	50
Tableau 5: fichier « .ini » du PIG	51
Tableau 6: paquets Debian nécessaires au cœur de calcul Plathynes	57
Tableau 7: mode de calcul du PIPt	61
Tableau 8: fichier « .ini » du PIPt	62
Tableau 9: mode de calcul du PIM	68
Tableau 10: fichier « .ini » du PIT	69
Tableau 11: mode de calcul du PIM	76
Tableau 12: fichier « .ini » du PIM	77
Tableau 13: fichier « .ini » du PIM - « Rampe-sections »	78
Tableau 14 : Exemple de suppression de dossier de crue	79
Tableau 15 : Exemple de mise à jour complète d'un modèle Mascaret	80
Tableau 16: mode de calcul du PIFG	84
Tableau 17: fichier « .ini » du PIFG	85

Liste des figures

Figure 1: Communication POM / PIx	14
-----------------------------------	----

1. Généralités

1.1 Objet du document

Ce document présente les procédures d'installation et d'exploitation des Programmes d'Interface (Plx) avec la POM des plateformes suivantes :

- ✓ GRP (PIG)
- ✓ PLATHYNES (PIPt)
- ✓ Télémac (PIT)
- ✓ Mascaret (PIM)
- ✓ Bouchon générique d'interface (PI)

1.2 Principe général

Le Plx est lancé par la POM sur un jeu de fichiers créé spécialement pour ce lancement.

Le Plx, après traitements, lance la plateforme de modélisation puis récupère les données produites pour les renvoyer à la POM.

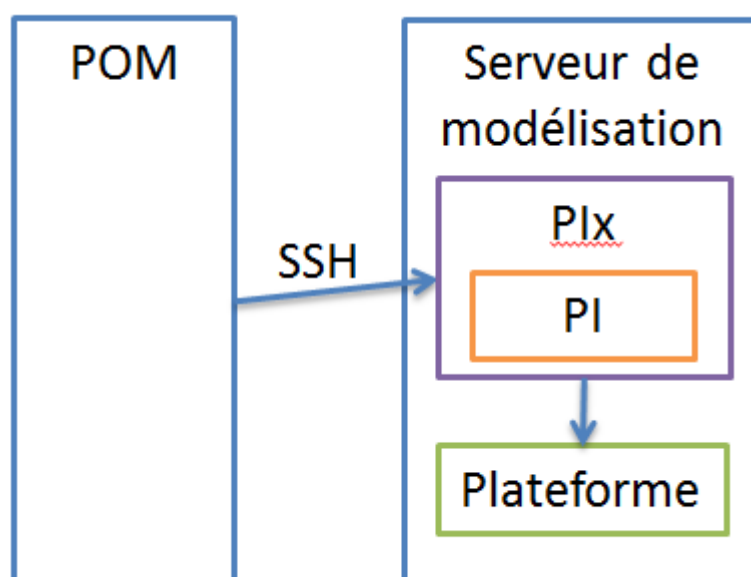


Figure 1: Communication POM / Plx

La POM se connecte sur le serveur de la plateforme de modélisation à l'aide du protocole SSH. Ce protocole est natif pour les environnements Linux mais nécessite l'installation d'un logiciel spécifique pour les environnements Windows (un « serveur SSH », cf. 2.1).

1.3 Protocole POM

1.3.1 Présentation

Les programmes d'interface sont des programmes codés en python 2.7 qui implémentent le protocole d'échange de la POM et qui sont chargés de le transcrire pour leur plateforme de modélisation.

Le protocole POM fait l'objet d'un numéro de version qui permet d'identifier la compatibilité du Plx avec la version de la POM qui tente de le piloter.

Le protocole POM regroupe les éléments et procédures suivantes :

- ✓ La POM copie sur le serveur un ensemble de fichiers structurés, et auto-descriptifs (cf. 1.3.2)
- ✓ Le Plx doit générer lors de son lancement un fichier de suivi (progression.xml) assurant la communication avec la POM
 - avancement
 - réussite
 - échec
 - messages d'information, d'avertissement, d'erreur
- ✓ Le PIX doit générer les fichiers de sortie attendus par la POM

Afin de faciliter ces communications, une librairie générique a été développée en python 2.7 : le PI (Programme d'Interface). Cette librairie permet de surcharger (i.e. réécrire) uniquement les parties du code source dont le fonctionnement est spécifique à la plateforme de modélisation (formats de fichiers, initialisations spécifiques, ...).

Le tableau présenté en 1.4 présente les correspondances de version et les compatibilités. Tous les Plx ne reposent pas sur le PI.

1.3.2 Arborescence des fichiers POM

La POM génère sur le serveur de calcul une arborescence de fichiers dans le répertoire dit « d'échange » de la plateforme (renseigné dans l'interface POM, sur l'objet « plateforme »).

Chaque lancement est cloisonné dans une série de répertoires pour assurer qu'ils ne s'écrasent pas les uns les autres. Pour un lancement, le répertoire racine est le suivant :

```
{ECHANGES}/{CODE_SESSION}/{DATE_PIVOT}/{ID_CONFIG}/
{CODE_MODELE_POM}/{CODE_SCENARIO_PRINCIPAL_POM}
```

Note : dans ce qui suit {RACINE} désignera ce répertoire.

La POM génère dans ce répertoire racine les fichiers et répertoires suivants :

```
{CODE_SCNEARIO_COMPL_1}
  inputs
  outputs
{CODE_SCNEARIO_COMPL_2}
  inputs
  outputs
...
{CODE_SCNEARIO_COMPL_X}
  inputs
  outputs
inputs
outputs
parameters.xml
```

```
progression.xml  
pom.log
```

Note : en noir les répertoires, en bleu les fichiers.

Les fichiers parameters.xml et progression.xml sont définis dans des schémas XML disponibles sur demande.

Le fichier pom.log est la redirection de la sortie standard de l'exécution du Plx, par exemple :

```
c:\pig\__main.py > {RACINE}\pom.log 2&>1
```

On y trouve donc les messages produits par le Plx (en plus du fichier progression.xml).

1.3.3 Test de compatibilité

Depuis la version 2.1, le PI permet de tester la compatibilité du protocole d'échange de la POM. Les Plx basés sur le PI v2.1 (ou postérieur) implémentent donc ce test automatique.

1.4 Correspondances

Le tableau ci-dessous présente les correspondances de compatibilité entre les différents éléments de « l'écosystème POM ».

Plx	VERSION	PLATEFORME DE MODÉLISATION	PROTOCOLE POM	PI
PIG	1.0	GRP 3.1	POM 1.6	Aucun
	2.0	GRP 3.2 GRP 3.3 GRP 2016	Protocole 2.1	PI 2.1 (libhydro 0.6.6)
	2.1	GRP 2016	Protocole 2.1, 2.2	PI 2.2 (libhydro 0.6.6 ; libbdimage 1.0.0)
	3.0	GRP 2018	Protocole 2.2, 2.3	PI 3 (libhydro 0.8 ; libbdimage 1.0.0)
	3.1	GRP 2018	Protocole 2.2, 2.3	PI 3.1 (libhydro 0.8 ; libbdimage 1.0.0)
	3.2	GRP 2018 GRP 2020	Protocole 2.2, 2.3	PI 3.2 (libhydro 0.8 ; libbdimage 1.5.8)
	4,0	GRP 2018 GRP 2020 GRP 2021	Protocole 2.2, 2.3	PI 4.0 (libhydro 0.9.4 ; libbdimage 1.5.8)
PIPt	1.0	PLATHYNES 1.6	POM 1.6	Aucun
	2.0	PLATHYNES 1.6	Protocole 2.1	PI 2.1 (libhydro 0.6.6)
	2.1	PLATHYNES 1.8.7	Protocole 2.1, 2.2	PI 2.2 (libhydro 0.6.6 ; libbdimage 1.0.0)
	2.2	PLATHYNES 1.8.7	Protocole 2.1, 2.2	PI 2.2 (libhydro 0.6.6 ; libbdimage 1.0.0)
	3.0	PLATHYNES 1.9.4	Protocole 2.2, 2.3	PI 3 (libhydro 0.8 ; libbdimage 1.0.0)
	3.1	PLATHYNES 1.9.4	Protocole 2.2, 2.3	PI 3.1 (libhydro 0.8 ; libbdimage 1.0.0)
	4.0	PLATHYNES 1.9.4	Protocole 2.2, 2.3	PI 4.0 (libhydro 0.9.4 ; libbdimage 1.5.8)

PIT	2.0	Telemac XX	Protocole 2.2, 2.3	PI 3 (libhydro 0.8 ; libbdimage 1.0.0)
	3.1	Telemac XX	Protocole 2.2, 2.3	PI 3.1 (libhydro 0.8 ; libbdimage 1.0.0)
	4.0	Telemac XX	Protocole 2.2, 2.3	PI 4.0 (libhydro 0.9.4 ; libbdimage 1.5.8)
PIM	2.0	Mascaret XX	??	PI 1.0
	3.0	Mascaret XX	Protocole 2.2, 2.3	PI 3 (libhydro 0.8 ; libbdimage 1.0.0)
	3.1	Mascaret XX	Protocole 2.2, 2.3	PI 3.1 (libhydro 0.8 ; libbdimage 1.0.0)
	3.2	Mascaret XX	Protocole 2.2, 2.3	PI 3.2 (libhydro 0.8 ; libbdimage 1.5.8)
	4.0	Mascaret XX	Protocole 2.2, 2.3	PI 4.0 (libhydro 0.9.4 ; libbdimage 1.5.8)

Tableau 1: correspondances POM – PIx – plateformes de modélisation

2. Installation des dépendances

Ce chapitre présente les installations des composants nécessaires à la mise en place des Plx.

2.1 Serveur SSH

2.1.1 Généralités

Les interactions entre la POM et les serveurs de calcul passent par le protocole SSH (Secure SHell). Il s'agit d'un protocole de communication sécurisé entre serveurs qui permet notamment :

- ✓ D'échanger des fichiers (téléchargement et téléversement)
- ✓ De lancer des commandes sur le serveur distant (ce que fait la POM quand elle lance un calcul)

Les serveurs de calcul doivent respecter les contraintes suivantes :

- ✓ Disposer d'un serveur SSH (logiciel de communication entre machines selon le protocole SSH) :
 - ce genre de serveur est natif sous Linux et ne nécessite donc pas d'installation. Les serveurs ssh natifs sur les versions récentes de Windows serveur ne sont pas encore gérées par la POM. Sous Windows, il est donc toujours nécessaire d'installer un serveur ssh pour communiquer avec la POM :
 - soit Copssh_4.9.4, léger mais limité à une seule connexion SSH simultanée
 - soit Cygwin, plus lourd, qui ne limite pas le nombre de connexions SSH simultanées
- ✓ Etre synchronisée temporellement avec le serveur POM

2.1.2 CopSSH pour Windows

L'espace documentaire du projet « Alfresco » contient la version 4.9.4 du logiciel Copssh pour disposer d'un serveur SSH (CopSSH) sous Windows , léger mais limité à une seule connexion SSH simultanée dans cette version gratuite :

Documents > Interfaçage avec les modèles > Serveur ssh pour Windows

2.1.3 Installation de Cygwin pour Windows

Le logiciel de serveur SSH « CopSSH » présenté ci-dessus dispose d'une licence gratuite n'autorisant qu'une connexion à la fois. Le présent chapitre présente la procédure d'installation d'un autre logiciel de serveur SSH, Cygwin, qui ne connaît pas cette limitation.

Note : il s'agit en fait d'un émulateur de Linux sous Windows qui offre en particulier le serveur SSH.

L'installation est largement inspirée de la page :

<http://www.commentcamarche.net/faq/2132-reseaux-installation-d-un-serveur-ssh-sous-windows>

2.1.3.1 Prérequis

L'installation de Cygwin peut interagir avec l'installation de CopSSH. Nous recommandons de désinstaller au besoin CopSSH pour installer Cygwin.

L'installation est réalisée depuis internet. L'éventuel proxy doit être paramétré sur le poste.

Il est nécessaire d'être connecté avec un compte administrateur pour réaliser l'installation.

2.1.3.2 Installation

- ✓ Téléchargez l'installeur Cygwin (setup.exe) de

<http://www.cygwin.com/>

Note : l'adresse de téléchargement est https://cygwin.com/setup-x86_64.exe pour les Windows en 64 bits.

- ✓ Lancez-le.

Dans l'installeur :

- ✓ Lancer l'exécutable " setupXXXX.exe " et cliquez sur " suivant "
- ✓ Dans la fenêtre " Choose installation type "
 - choisissez " Install from internet "
 - cliquez sur " suivant "
- ✓ Dans la fenêtre " Choose Installation Directory "
 - choisissez un répertoire d'installation
 - choisissez " all users "
 - cliquez sur " suivant "
- ✓ Dans la fenêtre " Select Local Package directory "
 - choisissez le même répertoire que le répertoire d'installation (les fichiers d'install cygwin seront placés dans un sous-répertoire de setup.exe)
 - cliquez sur " suivant "
- ✓ Dans la fenêtre " Select connection type "
 - Choisissez votre connexion internet et entrez d'éventuels paramètres de proxy si vous en utilisez un
 - cliquez sur " suivant "
- ✓ Dans la fenêtre " Choose Download Site(s) "
 - Choisissez un site de téléchargement proche de chez vous (les serveurs en .fr). Au pire, choisissez des serveurs dans des pays voisins.
 - cliquez sur " suivant "
- ✓ Dans la fenêtre " Select packages "
 - cliquez sur la liste déroulante "View" et sélectionner « full » pour voir tous les paquets disponibles.
 - chercher "openssh: The OpenSSH server and client programs" soit en descendant dans la liste, soit en tapant les première lettres dans le filtre de recherche
 - cliquez sur le mot " Skip " (colonne Current) pour sélectionner ce package. Le mot " Skip " doit alors être remplacé par la version d'openssh (par exemple "7.6p1-1 ")
 - d'autres packages vont automatiquement être sélectionnés
 - cliquez sur " suivant "
- ✓ Le téléchargement commence (environ 17 Mo)
- ✓ Dans la fenêtre " installation status and create icons "

- Choisissez les icônes à créer
- cliquez sur " Finish "

L'installation de Cygwin est terminée, nous allons désormais paramétrer le serveur SSH.

2.1.3.3 Configuration du serveur ssh

Une fois Cygwin installé, il faut paramétrer le serveur SSH.

2.1.3.3.1 Modification de l'environnement

L'objectif de cette étape est de déclarer Cygwin comme exécutable du système. Cela permet de le lancer en ligne de commande depuis n'importe quel répertoire.

- ✓ Cliquez-droit sur le Poste de travail > "Propriétés" > "Avancé" > "Variables d'environnement" > "Variables système".

Note : sur les versions récentes de Windows, il faut ouvrir un explorateur de fichiers (menu démarrer -> ordinateur) puis cliquer droit sur l'icône « poste de travail » (ou « ordinateur ») puis propriétés puis le menu « paramètres système avancés » puis le bouton « variables d'environnement » de la nouvelle fenêtre

- ✓ Cliquer sur "Nouveau" du bloc « Variables systèmes », et entrer la variable : CYGWIN ; valeur : ntsec tty
- ✓ Sélectionnez PATH dans la liste, cliquez sur "modifier" et ajoutez à la fin du chemin le répertoire « bin » du répertoire d'installation de Cygwin précédé d'un « ; ». Par exemple :

;C:\Cygwin\bin

Ou bien

;C:\Programmes\cygwin64\bin

Note : attention de bien positionner le « ; » du début. C'est le séparateur des différents exécutable du « path ».

2.1.3.3.2 Déclaration des utilisateurs et groupes

Cette étape vise à déclarer les comptes utilisateurs autorisés à se connecter en SSH.

- ✓ Ouvrez la fenêtre Cygwin en tant qu'administrateur
 - menu démarrer -> clic droit sur « Cygwin terminal » -> « exécuter en tant qu'administrateur »
- ✓ Créer les utilisateurs (première commande) et les groupes d'utilisateur (seconde commande) en lançant les deux commandes suivantes :

```
mkpasswd -l > /etc/passwd
mkgroup -l > /etc/group
```

Attention : « -l » est bien un L MINUSCULE, et PAS le chiffre 1.

Note : si vous avez un message d'erreur sur « mkpasswd » ou « mkgroup », inutile de poursuivre l'installation : vous devez d'abord résoudre ce problème avant de continuer.

Cela va prendre les « users » et « groupes » locaux de Windows et les créer dans les fichiers correspondants Cygwin.

Ces deux commandes écrasent le contenu des fichiers « /etc/passwd » ou « /etc/group » par le résultat de la commande « mkpasswd » ou « mkgroup ». Ces deux fichiers sont lus par le serveur SSH pour déterminer quels utilisateurs peuvent se connecter au serveur ainsi que leur mot de

se passe Windows (crypté) sur la machine ou sur le réseau. Vous pouvez donc la lancer autant de fois que désiré, les fichiers cibles seront écrasés.

L'objectif est d'avoir les mêmes login et mots de passe SSH que sur Windows.

Si vous souhaitez mettre dans ces fichiers les utilisateurs et groupes locaux et du domaine vous pouvez faire :

```
mkpasswd -l > /etc/passwd
mkpasswd -d >> /etc/passwd
mkgroup -l > /etc/group
mkgroup -d >> /etc/group
```

Note : le symbole « > » écrase le fichier cible, le symbole « >> » complète le fichier cible.

Pour ajouter un user précis, utilisez -u :

```
mkpasswd -u johnny -l > /etc/passwd
```

Contrôlez bien le contenu des fichiers passwd et group.

```
cat /etc/passwd
cat /etc/group
```

Si ces fichiers sont vides, le serveur ssh ne fonctionnera pas.

Si un utilisateur ET SON GROUPE ne sont pas déclarés dans ces 2 fichiers, il ne pourra pas se connecter.

Le fichier « /etc/passwd » (utilisateurs) présente des lignes structurées comme suit :

```
nom_du_compte:mot_de_passe:numero_utilisateur:numero_de_groupe:co
mmentaire:répertoire:programme_de_demarrage
```

Le fichier « /etc/group » présente des lignes structurées comme suit :

```
nom_de_groupe:champ_special:numero_de_groupe:membre1,membre2
```

Pour vérifier si un utilisateur « xpt » peut se connecter faire :

```
cat /etc/passwd | grep xpt
```

S'il n'y a pas de résultat, l'utilisateur n'est pas déclaré. Sinon, lire son numéro de groupe dans le résultat (disons NNNN) puis entrer la commande :

```
cat /etc/group | grep NNNN
```

S'il n'y a pas de résultat, c'est que le groupe n'existe pas.

2.1.3.3 Configurer le serveur SSH

Lancer la commande :

```
ssh-host-config -y
```

Il est possible qu'il demande un mot de passe pour la création d'un user « cyg_server ». C'est le user qui sera utilisé pour faire tourner le service sshd.

Note : si le système ne parvient pas à créer seul l'utilisateur « cyg_server » il faut le créer à la main et relancer la commande ci-dessus.

S'il vous est demandé "CYGWIN=", entrez

```
ntsec tty
```

Cela va créer le service sshd dans Windows. Il apparaîtra sous le nom "CYGWIN sshd" dans la liste des services. Il est en principe configuré pour démarrer automatiquement, mais il n'est pas encore démarré.

2.1.3.4 Lancer le serveur SSH

Utilisez la commande

```
net start sshd
```

ou bien la commande

```
cygrunsrv -S sshd
```

Notez que le service démarrera automatiquement au prochain redémarrage de Windows, vous n'aurez donc plus à taper cette commande.

Si le service ne démarre pas, regardez le contenu du fichier C:\cygwin\var\log\sshd.log

Selon les installations, il est possible que vous ayez à faire `chown system /etc/ssh*` et `chown system /var/empty` pour que le service démarre correctement.

Il se peut également que l'utilisateur endossé par le service (cf. ci-dessus `cyg_server`) n'ait pas le droit de lancer un service, car il a été créé « à la main ». Dans ce cas, il faut lancer le service une fois manuellement :

- ✓ Ouvrir la fenêtre des services : Menu démarrer -> panneau de configuration -> système et sécurité -> outils d'administration -> services
- ✓ Rechercher le service CYGWIN sshd et faire clic droit -> propriétés
- ✓ Dans l'onglet « connexion », choisir « Ce compte » et cliquer sur parcourir pour sélectionner le compte `cyg_server` :
 - Taper `cyg_server` dans le champ texte
 - Cliquer sur « vérifier les noms »
- ✓ Saisir le mot de passe de l'utilisateur
- ✓ Cliquer sur OK
- ✓ Dans la fenêtre de commande Cygwin, taper :


```
net start sshd
```

2.1.3.5 Tester le serveur SSH

Pour tester si le serveur est démarré entrer dans la fenêtre de commande Cygwin du serveur SSH :

```
ssh monlogin@localhost
```

où « monlogin » est le login d'un des utilisateurs déclaré dans le fichier des utilisateurs.

A la première connexion, le client ssh va probablement vous demander de confirmer la clé de sécurité.

Ensuite, après l'entrée du mot de passe, vous devez obtenir un shell de commande Linux. Vous pouvez voir la connexion en tapant : `echo $SSH_CONNECTION` (Port 22 = votre serveur ssh).

Taper la commande ci-dessous pour se déconnecter :

```
exit
```

Il se peut que l'erreur suivante apparaisse :

```
/bin/bash: Operation not permitted
```

Dans ce cas, il faut attribuer un droit supplémentaire à l'utilisateur cyg_server qui fait tourner le service SSH. Pour cela :

- ✓ Menu démarrer -> panneau de configuration -> système et sécurité -> outils d'administration -> Stratégie de sécurité locale
- ✓ Dans cette nouvelle fenêtre, ouvrir dans l'arborescence de gauche « Stratégies locales » -> « attribution des droits d'utilisateur »
- ✓ Parmi les droits affichés, faire « clic droit -> propriétés » sur « remplacer un jeton de niveau processus »
- ✓ Dans la nouvelle fenêtre, cliquer sur « ajouter un utilisateur » puis
 - Saisir cyg_server
 - Cliquer sur vérifier les noms
 - OK
- ✓ Appliquer et OK
- ✓ Relancer le service
- ✓ Retester la connexion SSH.

ATTENTION : certains pare-feu Windows empêchent la connexion SSH. Il faut donc désactiver le pare-feu pour le port 22.

2.1.3.6 Côté POM

Côté POM, vous pouvez désormais déclarer un serveur de calcul et une plateforme associée au serveur SSH nouvellement créé.

Pour les serveurs un bouton permet de tester la connexion SSH. Pour tester les plateformes de modélisation, il faut d'abord les avoir installées sur les serveurs ainsi que le programme d'interface dédié (voir plus loin dans le document).

Note : lorsque les lignes de commandes sont trop longues il est possible de créer côté serveur de modélisation un fichier exécutable (« .bat » sous Windows, « .sh » sous Linux) à lancer par la POM, qui contient ces commandes.

- ✓ Pour les serveurs Linux

Pour paramétrer les plateformes de modélisation côté POM, on utilise des chemins Linux (avec des / pour séparateur).

Exemples :

#

✓ Pour les serveurs Windows

Plateforme de modélisation ?

0 Information - Server : OK
 0 Information - Connexion : OK
 0 Information - Commande : echo LD_LIBRARY_PATH=/usr/local/gcc464/lib64:/usr/local/mpich2-1.2.1p1-gcc464/lib /home/mascaret/.venv/pim/bin/python /home/mascaret/utilities/pim/pim /home/mascaret/ECHANGE/20180828_022147_pom_params.xml [PARAMS SCENARIO] 1>/home/mascaret/ECHANGE/20180828_022147_pom_params.xml.log 2>&1 &
 0 Information - Exécution : OK
 0 Information - Rapatriement log : OK
 0 Information - Suppression du log distant : OK
 0 Information - Déconnexion : OK
 0 Information - Contenu du fichier log "/home/mascaret/ECHANGE/20180828_022147_pom_params.xml.log" :
 0 Information - LD_LIBRARY_PATH=/usr/local/gcc464/lib64:/usr/local/mpich2-1.2.1p1-gcc464/lib /home/mascaret/.venv/pim/bin/python /home/mascaret/utilities/pim/pim /home/mascaret/ECHANGE/20180828_022147_pom_params.xml [PARAMS SCENARIO]
 0 Information - Test réussi !

Nom	Type de plateforme	Mascaret
Server de calcul	DAMP2_3b+Pi	

Exécutable

Chemin complet de l'exécutable à lancer	LD_LIBRARY_PATH=/usr/local/gcc464/lib64:/usr/local/mpich2-1.2.1p1-gcc464/lib /home/mascaret/.venv/pim/bin/python /home/mascaret/utilities/pim/pim	Paramètres de commande ?
Chemin du répertoire des fichiers d'échange	/home/mascaret/ECHANGE ?	Supporte les lancements groupés de modèles Non
Conserver les fichiers en fin de calcul	Non	Commande complète LD_LIBRARY_PATH=/usr/local/gcc464/lib64:/usr/local/mpich2-1.2.1p1-gcc464/lib /home/mascaret/.venv/pim/bin/python /home/mascaret/utilities/pim/pim /home/mascaret/ECHANGE/path/to/parameters.xml [PARAMS SCENARIO] ?

Modifier retour aux plateformes Lancer un test de commande Tester la commande

Le paramétrage de la plateforme de modélisation côté POM se fait de la même façon que l'on utilise Copssh ou Cygwin comme serveur SSH.

Avec Cygwin, pour paramétrer la connexion SSH aux serveurs côté POM, il est préférable de renseigner le login et le mot de passe du compte de l'utilisateur Windows qui lance le service cygwin (sans quoi il peut apparaître des problèmes de droits d'accès aux fichiers).

Pour paramétrer les plateformes de modélisation côté POM, on utilise des chemins Windows (avec des \ pour séparateur). Pour les paramètres de commande, si des chemins sont nécessaires, il faut utiliser des chemins Windows (avec des \ pour séparateur) et les mettre entre '.

Il ne doit y avoir aucun chemin "cygwin" dans les paramètres de définition pour une plateforme sur serveur Windows¹.

¹ Car on utilise cmd

Exemple :

```

rmation - Server : OK
rmation - Connexion : OK
rmation - Génération d'un fichier de paramétrage vide ...
rmation - Génération du fichier de paramétrage : OK
rmation - Envoi du fichier de paramétrage : OK
rmation - Commande : cmd /c 'C:\Anaconda2\envs\pig2.1\python.exe C:\Anaconda2\envs\pig2.1\Lib\site-packages\pig\__main__.py' 'C:\GRP_TR\v2016\Echanges\2019_090414_pom_params.xml' [PARAMS SCENARIO] 1>/cygdrive/c/GRP_TR/v2016/Echanges/20190419_090414_pom_params.xml.log 2>&1 &
rmation - Exécution : OK
rmation - Rapatriement log : OK
rmation - Suppression du log distant : OK
rmation - Suppression du fichier de paramétrage distant : OK
rmation - Déconnexion : OK
rmation - Contenu du fichier log 'C:\GRP_TR\v2016\Echanges\20190419_090414_pom_params.xml.log' :
rmation - usage: __main__.py [-h] [--verbose] [--force] PARAMETERS_FILE [INI_FILE]
rmation - __main__.py: error: unrecognized arguments: SCENARIO]
rmation - Test réussi !

```

Informations

GRP-pig2.1 Type de plateforme Grp

Paramètres de commande ?

Commande complète ?

Lancer un test de commande

Chemin complet :

C:\Anaconda2\envs\pig2.1\python.exe C:\Anaconda2\envs\pig2.1\Lib\site-packages\pig__main__.py

Paramètres de commande (a priori vide)

'c:\toto'

Chemin d'échanges :

C:\GRP_TR\Echanges

La commande générée est :

```
cmd /c 'C:\Anaconda2\envs\pig2.1\python.exe C:\Anaconda2\envs\pig2.1\Lib\site-packages\pig\__main__.py' 'c:\toto' 'C:\GRP_TR\Echanges\20180830_125203_pom_params.xml' [PARAMS SCENARIO] 1>/cygdrive/c/GRP_TR/Echanges/20180830_125203_pom_params.xml.log 2>&1 &
```

Explications détaillées :

Chemin complet : il est mis entre ' par la POM donc on peut et on doit utiliser des chemins Windows avec des \.

Paramètres de commande : ils ne sont pas mis entre ' par la POM, donc si dedans il y a un argument qui est un chemin (qui doit être Windows), cet argument doit être mis entre ' dans la plateforme et les \ laissé tels quels.

Chemin d'échanges : il est mis entre ' par la POM donc on peut et on doit utiliser des chemins Windows avec des \.

Si les paramètres supplémentaires de la ligne de commande [PARAMS SCENARIO] qui est défini au niveau de chaque scénario principal comportent des chemins Windows (avec des \ pour séparateur), ils doivent être mis entre '.

2.2 Installation de Python

Le PI fonctionne en Python, qu'il faut installer s'il ne l'est pas déjà. Nous conseillons d'utiliser une installation packagée, avec des modules de base pré installés.

L'idée est d'utiliser le concept d'environnement virtuel Python cf §2.4 - Création et activation d'un environnement virtuel, pour cloisonner les différentes installations des Plx (de natures différentes ou de versions différentes).

2.2.1 Sous Windows

L'installation est réalisée à l'aide de l'outil Miniconda qui facilite les manipulations.

2.2.1.1 Miniconda

Conda (ou Miniconda) est un gestionnaire de paquets compilés prenant en charge les dépendances.

Conda intègre la librairie 'virtualenv'.

Voir : <http://www.lfd.uci.edu/~gohlke/pythonlibs/>, <https://en/latest/installing/>

- Télécharger et installer l'installateur miniconda

Voir : <http://conda.pydata.org/miniconda.html>

2.2.1.2 Paramétrer le proxy

L'installateur conda doit pouvoir accéder à internet en http pour télécharger les paquets à installer, ce qui peut nécessiter le paramétrage d'un proxy (à vérifier le cas échéant avec votre administrateur réseau).

La méthode conseillée est l'utilisation d'un fichier « .condarc ».

Voir : <https://docs.python.org/2/using/windows.html>

Voir : <https://conda.io/docs/config.html>

Voir : <https://conda.io/docs/config.html#the-conda-configuration-file-condarc>

- Créer un fichier « .condarc » dans le répertoire « home » de l'utilisateur courant

Exemple de contenu du fichier :

```
# Proxy settings: http://[username]:[password]@[server]:[port]
proxy_servers:
    http: http://[username]:[password]@[server]:[port]
```

Note : il faut remplacer les « [] » par les valeurs correspondantes.

Pour les utilisateurs du ministère :

```
# Proxy settings: http://[username]:[password]@[server]:[port]
proxy_servers:
    http: http://pfrie-std.proxy.e2.rie.gouv.fr:8080
    https: http://pfrie-std.proxy.e2.rie.gouv.fr:8080
```

A défaut, le proxy internet peut être déclaré dans la console DOS avec la commande « set », avant d'exécuter des commandes d'installation Python :

```
set http_proxy=http://[username]:[password]@[server]:[port]
```

Note : il est déconseillé d'utiliser des variables d'environnement globales qui peuvent impacter d'autres applications.

Pour les utilisateurs du ministère :

```
set https_proxy=pfrie-std.proxy.e2.rie.gouv.fr:8080
```

2.3 Installation des utilitaires Python

Python offre la possibilité de cloisonner complètement une partie de ses bibliothèques afin de ne pas perturber la configuration générale de Python. Un **environnement virtuel Python** permet de faire cohabiter plusieurs Plx basés sur des versions différentes de PI.

L'installation des Plx repose sur cette bonne pratique.

Pour les VM Linux l'installation préalable de l'installateur de paquets python PIP est décrite ci-dessous.

2.3.1 Pour VM modèle SCHAPI Debian 11

2.3.1.1 Installation de pip

Il faut préparer la VM pour qu'elle ait une commande **python** :

```
$ cd /usr/bin
$ sudo ln -s python3.9 python
```

Puis effectuer l'installation de **pip** :

```
$ cd ~
$ sudo apt update
$ sudo apt install python3-pip
```

Vérifier la version de **pip** (les commandes **pip** et **pip3** sont identiques) :

```
$ pip --version
pip 20.3.4 from /usr/lib/python3/dist-packages/pip (python3.9)

$ pip3 --version
pip 20.3.4 from /usr/lib/python3/dist-packages/pip (python3.9)
```

!! Si ce n'est pas le cas (si pip vis la version 2.7), il faudra utiliser pip3 à la place de pip dans la suite des commandes.

Si besoin, effectuer la mise à jour de **pip** :

```
$ sudo pip install --upgrade pip
$ pip --version
```

```
pip 23.1.2 from /usr/local/lib/python3.9/dist-packages/pip
(python3.9)
```

2.3.1.2 Installation des virtualenvs

Il faut installer les paquets suivants :

```
$ sudo pip install virtualenv
$ sudo pip install virtualenvwrapper
```

Puis préparer le dossier qui recevra les environnements virtuels Python (commande à faire avec l'utilisateur applicatif souhaité) :

```
$ mkdir $HOME/virtualenvs
```

Il faut ajouter les lignes suivantes dans le nouveau fichier **\$HOME/.bash_profile** :

```
# virtualenvwrapper
export WORKON_HOME=$HOME/virtualenvs

source /usr/local/bin/virtualenvwrapper.sh
```

Pour finaliser l'installation des environnements virtuels, il faut se déconnecter puis se reconnecter :

virtualenvwrapper doit afficher des informations de création des fichiers de base ci-dessous

```
Debian GNU/Linux 11 pom3-bdd tty1
Hint: Num Lock on

pom3-bdd login: schapi
Password:
Linux pom3-bdd 5.10.0-10-amd64 #1 SMP Debian 5.10.84-1 (2021-12-08) x86_64
#####
Serveur = modele-debian11
Virtualisation = Vm
OS = Debian11
Adresse IP = 10.31.178.145
Fonction = modele debian11
#####
Last login: Fri Jun 16 19:25:31 CEST 2023 on tty1
You have new mail.
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/premkproject
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/postmkproject
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/initialize
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/premkvirtualenv
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/postmkvirtualenv
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/prermvirtualenv
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/postrmvirtualenv
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/predeactivate
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/postdeactivate
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/preactivate
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/postactivate
virtualenvwrapper.user_scripts creating /home/schapi/virtualenvs/get_env_details
19:26:13-schapi@pom3-bdd:~\
$
```

!! Si ces lignes ne sont pas affichées, il convient de taper les 2 lignes du `.bashprofile` avant de pouvoir créer (commande `mkvirtualenv`) ou se connecter (commande `workon`) à un environnement virtuel python.

2.3.2 Spécificités pour les VMs modèles Plathynes, Mascaret et GRP

2.3.2.1 Modification du fichier `.bashrc` pour les VM PLATHYNES

Ajouter les lignes suivantes au fichier `/home/plathynes_pom/.bash_profile`

```
#Virtualenvwrapper

export WORKON_HOME=$HOME/PIPt/virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

Puis se déconnecter et se reconnecter afin de tester la disponibilité des commandes `mkvirtualenv` et `workon` :

```
$ ssh plathynes_pom@XX.XX.XX.XX
$ mkvirtualenv --version
virtualenv 20.0.27 from
/usr/local/lib/python2.7/dist-packages/virtualenv/__init__.pyc
```

2.3.2.2 Modification du fichier `.bashrc` pour les VM MASCARET

Ajouter les lignes suivantes au fichier `/home/mascaret/.bash_profile`

```
#Virtualenvwrapper

export WORKON_HOME=$HOME/pom/pim/install/virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

Puis se déconnecter et se reconnecter afin de tester la disponibilité des commandes `mkvirtualenv` et `workon` :

```
$ ssh mascaret@XX.XX.XX.XX
$ mkvirtualenv --version
virtualenv 20.0.27 from
/usr/local/lib/python2.7/dist-packages/virtualenv/__init__.pyc
```

2.3.2.3 Modification du fichier `.profile` pour les VM GRP

Ajouter les lignes suivantes au fichier `/home/grp_pom/.profile`

```
#Virtualenvwrapper
```

```
export WORKON_HOME=$HOME/.virtualenvs
source /home/grp_pom/.local/bin/virtualenvwrapper.sh
```

Puis se déconnecter et se reconnecter afin de tester la disponibilité des commandes **mkvirtualenv** et **workon** :

```
$ ssh grp_pom@xx.xx.xx.xx
$ mkvirtualenv --version
virtualenv 20.0.27 from
/usr/local/lib/python2.7/dist-packages/virtualenv/__init__.pyc
```

2.3.3 Sous Windows

Avec anaconda (voir <https://conda.io/docs/using/envs.html>) ou miniconda, taper dans une fenêtre de commande DOS :

```
conda create --name XXXXX pip python=2.7
```

où

✓ XXXXX est le nom de l'environnement virtuel à créer

L'environnement virtuel Python est créé (il s'appelle XXXXX). Désormais, toutes les installations doivent se faire dans cet environnement, qui doit au préalable être activé. Les dépendances doivent être installées en priorité avec anaconda, et à défaut avec pip (l'installateur de python) pour les modules en pur python non disponibles avec anaconda.

2.4 Création et activation d'un environnement virtuel

Avant toute étape d'installation de configuration ou de paramétrage dans l'environnement virtuel, il faut l'activer.

Toutes les commandes tapées par suite le seront dans cet environnement virtuel.

2.4.1 Sous Linux

Création de l'environnement virtuel avec la commande :

```
$ mkvirtualenv -p python3.7 XXXX
```

où XXXX est le nom de l'environnement virtuel à créer.

Pour activer l'environnement virtuel préalablement créé taper la commande :

```
$ workon XXXX
```

Pour désactiver l'environnement virtuel taper la commande :

```
$ deactivate
```

Pour supprimer un environnement virtuel taper la commande :

```
$ rmvirtualenv XXXX
```

Attention, il n'y a pas de demande de confirmation avant la suppression.

Pour lister les environnements virtuels créés sur votre machine

```
$ lsvirtualenv
```

2.4.2 Sous Windows

Ouvrir une fenêtre de commande DOS et taper les commandes ci-dessous.

```
activate XXXXX
```

où XXXX est le nom de l'environnement virtuel à activer.

Selon les installations de Windows et les versions de Conda, il se peut que cela ne fonctionne pas. Dans ce cas, il faut lancer le script « activate.bat » de l'environnement virtuel :

```
cd {CHEMIN_VIRTUAL_ENV}\venv\Scripts  
activate.bat
```

où {CHEMIN_VIRTUAL_ENV} est le nom complet du répertoire d'installation de l'environnement virtuel à activer.

2.5 Mise à jour sécurité pour VM Debian

Pour effectuer les mises à jour de sécurité sur une VM Debian, il faut faire en **root** :

```
vi /etc/apt/sources.list  
  
deb http://security.debian.org/debian-security stable-security main  
  
# commenter toutes les autres lignes
```

Mettre à jour le référentiel des dépôts

```
apt update
```

Effectuer les mises à jour de sécurité

```
apt upgrade
```

Si la mise à jour ne se fait pas, et qu'il y a un message pour indiquer que des dépendances ne sont pas satisfaites. Par exemple :

```
apt upgrade  
  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
You might want to run 'apt --fix-broken install' to correct  
these.  
The following packages have unmet dependencies:  
libgcc-8-dev : Depends: libquadmath0 (>= 8.3.0-6) but 6.3.0-  
18+deb9u1 is installed  
libgfortran3 : Depends: gcc-6-base (= 6.3.0-18+deb9u1) but it is  
not installable
```

```
libquadmath0 : Depends: gcc-6-base (= 6.3.0-18+deb9u1) but it is
not installable
E: Unmet dependencies. Try 'apt --fix-broken install' with no
packages (or specify a solution).
```

Comme indiqué dans le message de retour, il faut effectuer les corrections avec la commande suivante :

```
apt --fix-broken install
```

Puis relancer la commande de mise à jour :

```
apt upgrade
```

3. PI

Le PI seul n'a qu'un intérêt limité puisqu'il n'embarque qu'un bouchon destiné aux tests, par contre il est utilisé par les différents programmes d'interface PIG, PIpt, PIM et PIT. Attention à respecter la compatibilité entre les versions de ces programmes et avec la version de la POM cf. <https://partage.din.developpement-durable.gouv.fr/ethercalc/POM-Plx>.

Voilà la marche à suivre pour installer le PI.

Attention : l'installation de Python et la création d'un environnement virtuel XXXX est un prérequis à cette installation (cf. §2). Pour faire cohabiter plusieurs installations, il est préférable de créer un environnement dédié à chaque installation.

Télécharger les programmes à installer (a priori PomInterface et un Plx dédié au modèle à piloter) sur Alfresco POM

- sous Documents> Interfaçage avec les modèles>Librairie Pom Interface (PI)
- sous Documents> Interfaçage avec les modèles > Plateformes nationales (GRP, Plathynes, Mascaret, Telemac)

À partir de PI2.2, libhydro et libbdimage sont incluses dans le PI. La procédure à suivre est donc la suivante :

- activer l'environnement virtuel. cf. 2.4 Création et activation d'un environnement virtuel
- sous windows, installer les dépendances de libhydro et pip

```
conda install pandas==0.18.1 xml pip
```

- installer le PI. cf. 3.1 Installation
- installer le Plx. cf. chapitres 4, 5, 6

NB : Pour renvoyer ce qui sort dans le terminal vers un fichier install_pi_{horodatage}.log (sous /tmp) ; il faut ajouter à la fin de la commande :

(commande) > /tmp/install_pi_\$(date +%Y%m%d%H%M%S).log 2>&1

En cas de problème cela permet d'envoyer ce fichier à la personne à qui on demande de l'aide.

3.1 Installation

3.1.1 Sous Linux

Au préalable, activer l'environnement virtuel, télécharger l'archive du PI et se placer dans le répertoire où l'archive a été téléchargée.

Le téléchargement de paquets est nécessaire à l'installation du PI, la VM doit donc être configurée par le SPC pour pouvoir accéder à internet.

Lancer l'installation du PI

```
$ pip install pominterface-4.0.0-py3-none-any.whl
```

Si c'est nécessaire, la commande à lancer pour utiliser un accès Internet avec proxy :

```
$ pip install pominterface-4.0.0-py3-none-any.whl --
proxy=https://mon.proxy:ProxyIp
```

Ou en utilisant le proxy du ministère

```
$ pip install pominterface-4.0.0-py3-none-any.whl --
proxy=http://pfrie-std.proxy.e2.rie.gouv.fr:8080
```

Pour lister les paquets installés dans l'environnement python taper :

```
$ pip list
```

3.1.2 Sous Windows

Pour installer le PI, il faut télécharger le ZIP d'installation, le dézipper et lancer la commande d'installation.

Pour cela :

✓ Télécharger le package d'installation PI dans la bonne version

- soit depuis le site collaboratif du projet POM (<https://travail-collaboratif.din.developpement-durable.gouv.fr/share/page/site/dgprsrnhshapom/dashboard>), dans l'espace documentaire ==> Interfaçage avec les modèles ==> Librairie Pom Interface (PI)

✓ Extraire cette archive dans un répertoire temporaire, par exemple :

```
C:\GRP_TR2016\InstallPIG\pominterface-2.1.0
```

✓ Ouvrir une fenêtre de commande DOS et taper les commandes ci-dessous

- Activation de l'environnement virtuel du PI
activate XXXXX

où XXXXX est le nom de l'environnement virtuel du PI.

- Installer les dépendances de libhydro et pip (si ce n'est pas déjà fait)

```
conda install pandas==0.18 lxml pip
```

- Se placer dans le répertoire contenant le PI

```
cd C:\GRP_TR2016\InstallPIG
```

✓ Installer le PI

```
pip install pominterface-2.2.6.tar.gz --no-deps --upgrade >|
/tmp/install_pi_$(date +%Iminutes).log 2>&1
```

Attention : si un message d'erreur concernant la libhydro apparaît, c'est qu'il faut l'installer manuellement comme indiqué ci-avant.

Il se peut qu'un message d'erreur non bloquant concerne les versions de « numpy » soit affiché.

3.2 Exploitation

3.2.1 Fichier « .ini »

Le fichier « .ini » du PI contient des clefs de paramétrage communes aux Plx qui sont basés sur le PI.

Attention de bien formater ces .ini avec un encodage UTF-8 (sans BOM).

Ces clefs sont dans les sections :

```
[general]
[cleaning]
```

Les clefs générales sont les suivantes :

SECTION	CLEF	OBLIGATOIRE	FORMAT	DESCRIPTION
[general]	timeout	Oui	entier	Durée de vie en secondes du fichier « pid » de lock et du calcul du modèle. Lorsque cette durée est dépassée, le modèle est mis en erreur.
	modelname	Oui	chaîne	Nom du modèle (pour affichage des messages).
	rootdirectory	Oui	chaîne	Nom complet du répertoire racine de l'exécutable du modèle à lancer.
	commandlineworkdirectory	Oui	chaîne	Nom du répertoire dans lequel le PI doit se placer pour lancer le modèle.
	commandline	Oui	chaîne	Commande à lancer dans le répertoire racine ci-dessus. La ligne de commande générées est « rootdirectory/commandline »
	lancementsimultanes	Non	booléen	« True » si le modèle peut être lancé en parallèle, « False » sinon. Si « False » le PI vérifie qu'il n'est pas déjà en train de tourner.
	versionprotocolepom	Non	chaîne	Numéro de version du protocole POM pour vérification de la compatibilité.
	modelenvfile	Non	chaîne	Nom complet d'un fichier python « .py » à exécuter avant lancement pour positionner des variables d'environnement.
	conversiontable	Non	chaîne	Série de couples séparés par des « ; ». Chaque couple contient deux codes « code POM » « code modèle », séparés par des « ».

				Lors de la lecture des fichiers d'entrées, les données des entités de code POM sont remplacés par les codes modèles. Si un code POM est associé à plusieurs codes modèles, les données sont dupliquées.
	include	Non	chaîne	Nom du fichier « .ini » contenant les paramètres de base, que ce fichier « ini » vient écraser. Cela permet d'avoir un fichier « .ini » avec les paramètres généraux, et un ensemble de fichiers « .ini » pour modifier quelques paramètres.
[cleaning]	cleaners	Non	chaîne	Liste des noms des clefs définissant les répertoires à nettoyer, séparés par des « , ». Chaque nom doit correspondre à une section définie dans le fichier « .ini ».
Models	directory	Oui	chaîne	Chemin complet du répertoire des modèles. Ce répertoire doit exister. La valeur par défaut est définie par chaque Plx.
	names	Oui	chaîne	Code des modèles à lancer, séparés par des « ». Chaque code doit correspondre à un répertoire existant du répertoire des modèles.
multi_previ	max_nb_calculs_previ	Non	entier	Nombre maximum de calculs à lancer par le Plx. La valeur est comprise entre 1 et 1000. La valeur par défaut est « 2 ».
	max_cpu_load_percent	Non	entier	Maximum de la charge CPU pour pouvoir lancer un calcul. Valeur en % La valeur est comprise entre 10 et 100. La valeur par défaut est « 80 ».
	min_free_mem_mo	Non	entier	Minimum de RAM disponible pour pouvoir lancer un calcul. Valeur en Mo. La valeur est comprise entre 1 et 1000.

				La valeur par défaut est « 100 ».
	max_attente_sec	Non	entier	Nombre maximum de secondes d'attentes sans lancement de calcul. La valeur est comprise entre 1 et 10000. La valeur par défaut est 1800, soit 30 minutes

Tableau 2: fichier « .ini » du PI

Chaque répertoire à nettoyer fait l'objet d'une section du fichier « .ini », dont le nom doit être indiqué dans la clef « cleaning → cleaners ». Chaque section de « nettoyage » peut prendre les clefs suivantes :

SECTION	CLEF	OBLIGATOIRE	FORMAT	DESCRIPTION
[XXXX] (où XXXX est le nom de la section renseignée dans la clef cleaners)	directory	Non	Chaîne	Nom complet du répertoire à traiter.
	filter	Non	Chaîne	Filtre à appliquer sur les fichiers du répertoire. S'il est renseigné, seuls les fichiers correspondant à ce filtre seront supprimés.
	age	Non	Entier	Age en jours des fichiers à supprimer. S'il est renseigné, seuls les fichiers plus âgés seront supprimés.
	mode	Non	Chaînes	Peut valoir : « prerun » (nettoyage effectué avant l'exécution du modèle), « postrun » (nettoyage effectué après l'exécution du modèle), « pre-postrun » (nettoyage effectué avant et après l'exécution du modèle)

Tableau 3: fichier « .ini » du PI - « Cleaners »

3.2.2 Calculs concurrents

Un mécanisme du PI permet de s'assurer, si besoin, qu'il n'est pas déjà en train de s'exécuter (problématique dite des « calculs concurrents »).

Le fichier « .ini » (cf. 3.2.1) contient la clef « lancementsimultanes » qui permet d'indiquer si le PI peut se lancer s'exécuter à nouveau s'il est déjà en train de tourner.

Dans le cas d'une interdiction, le PI gère un fichier de verrou « pi.pid » dans le répertoire temporaire du serveur de calcul. Au lancement du PI, on vérifie si ce fichier existe et contient des numéros de processus (les PIDS) qui ne sont pas le sien.

Si oui, le PI lève une erreur et s'arrête.

Si non, le PI crée ce fichier et y positionne son numéro de processus et sa date de lancement. En fin d'exécution (y compris en erreur), le numéro du processus est retiré (et le fichier supprimé si besoin).

Attention :

- ✓ cette mécanique par défaut peut être modifiée par les Plx qui en auraient besoin (c'est le cas du PIPT par exemple),
- ✓ le nom du fichier de verrou « pi.pid » peut changer d'un Plx à l'autre.

3.2.3 Outil de vérification des fichiers .ini

A partir de la version 4.0.0, le PI inclut un outil de vérification et de mise à jour des fichiers .ini :

- ✓ Suppression des clés obsolètes
 - par exemple '[general]modelname'
- ✓ Transformation des clés
 - par exemple '[pix]xxdirectory' --> '[models]directory'
 - et '[pix]xxmodele' --> '[models]names'
 - mise à jour des chemins des exécutable pour les VM telemac.

Dès que le PI est installé dans un environnement virtuel, l'outil est accessible par la commande :

checkini

L'outil prend en paramètre un ou plusieurs chemins de fichiers ini.

Avant de modifier chaque fichier, il en effectue une sauvegarde, avec un suffixe « _bak »

L'outil « checkini » ne transforme que les clés existantes dans le fichier « .ini ».

Attention : Si la clé '[pix]xxdirectory' n'est pas définie, l'outil « checkini » ne générera pas la nouvelle clé « [models]directory », alors qu'elle est obligatoire.

C'est à l'utilisateur de l'ajouter manuellement.

Attention : pour le PIT, checkini va essayer de mettre à jour la clé '[general]commandline' ; Pour cela, il va effectuer une recherche du fichier 'telemac2d.py' dans le dossier '/home/mascaret'. S'il trouve ce fichier, il va modifier la clé '[general]commandline' avec le chemin complet de ce fichier. Sinon, il affiche un warning.

Attention : pour le PIT, checkini va essayer de créer la nouvelle clé '[pit]commandline_selafin'. Pour cela, il va effectuer une recherche du fichier 'run_selafin.py' dans le dossier '/home/mascaret'. S'il trouve ce fichier, il va créer la clé '[pit]commandline_selafin' avec le chemin complet de ce fichier. Sinon, il affiche un warning.

3.2.4 Gestion des fichiers .toml

A partir de la version 4.0.0, le PI peut lire les fichiers de paramétrage au format '.toml' (en plus du format '.ini' historique)

Les sections, clé et valeurs sont les mêmes que pour les '.ini' (cf § 3.2.1). Seule la syntaxe est différente.

Pour l'utilisateur, cela est totalement transparent : il peut remplacer un fichier '.ini' par un fichier '.toml' dans les commandes de lancement de Plx.

Exemple :

```
/chemin/vers/pix parameters.xml fichier.ini
```

ou

```
/chemin/vers/pix parameters.xml fichier.toml
```

3.2.5 Gestion des calculs parallèles en prévision

A partir de la version 4.0.0, le PI exécute les calculs prévision en parallèle, en fonction des réglages définis dans le « .ini » (cf XXX).

Dans le fichier « pom.log » produit par chaque calcul POM, l'utilisateur va retrouver les éléments suivants :

- ✓ une ligne de début de calcul parallèle :

```
INFO [ 86] === EXECUTION PARALLELE
```

- ✓ plusieurs lignes d'information du nombre de calculs encours :

```
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 1 | 0 |  
0 / 1 (+0.0s)
```

- ✓ d'éventuelles lignes de warning :

```
INFO [ 86] [WARN-SUPERVISEUR-PIPt] Impossible de lancer une  
commande car : la charge cpu est 100% (> 80%) ;
```

- ✓ deux lignes de fin de calcul parallèle :

```
INFO [ 86] [INFO-SUPERVISEUR-PIPt] Execution parallele terminee  
INFO [ 86] === FINALISATION EXECUTION PARALLELE
```

Les lignes d'information et de warning contiennent des balises dédiées (resp [INFO-SUPERVISEUR-xxx] et [WARN-SUPERVISEUR-xxx]) qui permettent une recherche facile . Les 'xxxx' correspondent au nom du Plx en cours d'exécution, par exemple 'PIM', 'PIT' ou 'PIPt'. Cela permet de classer les messages par type de modèle de calcul.

Note : La recherche d'éventuels warning peut être utilisée pour les outils de supervision afin d'alerter le franchissement d'une limite sur la VM d'un modèle. Ainsi, l'apparition de ces warnings peut signifier que la capacité de la VM n'est pas assez suffisante, et dans ce cas il est conseillé d'effectuer des actions correctives sur la VM, par exemple augmentation de la mémoire ou ajout d'un cœur de calcul supplémentaire.

Voici un exemple de commande permettant d'afficher les messages de warning :

```
find /chemin/vers/echanges/ -name "pom.log" | xargs grep "WARN-SUPERVISEUR-[PIM|PIT|PIPt]"
```

Les lignes d'informations contiennent des valeurs qui indiquent la progression des calculs parallèles :

Le texte d'information est de la forme :

```
calculs prevision : <nb1> | <nb2> | <nb3> / <nb4> (<tps>s)
```

avec les valeurs pour l'instance du Plx en cours d'exécution :

nb1 : nombre de calculs en attente

nb2 : nombre de calculs en cours

nb3 : nombre de calculs terminés

nb4 : nombre total de calculs

tps : temps en secondes depuis le début du calcul parallèle

Voici un exemple de progression avec 4 calculs demandés, mais 1 seul possible en parallèle :

```
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 4 | 0 | 0 / 4 (+0.0s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 3 | 1 | 0 / 4 (+0.1s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 3 | 0 | 1 / 4 (+8.7s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 2 | 1 | 1 / 4 (+8.8s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 2 | 0 | 2 / 4 (+24.3s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 1 | 1 | 2 / 4 (+24.5s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 1 | 0 | 3 / 4 (+27.4s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 0 | 1 | 3 / 4 (+27.5s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 0 | 0 | 4 / 4 (+31.8s)
```

Voici un autre exemple, avec 4 calculs, mais 2 possibles en parallèle :

```
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 4 | 0 | 0 / 4 (+0.0s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 3 | 1 | 0 / 4 (+0.1s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 2 | 2 | 0 / 4 (+0.2s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 2 | 1 | 1 / 4 (+8.8s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 1 | 2 | 1 / 4 (+8.9s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 1 | 1 | 2 / 4 (+12.0s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 0 | 2 | 2 / 4 (+12.1s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 0 | 1 | 3 / 4 (+16.5s)
INFO [ 86] [INFO-SUPERVISEUR] calculs prevision : 0 | 0 | 4 / 4 (+17.2s)
```

Les messages de warnings sont de la forme :

```
[WARN-SUPERVISEUR-pix] Impossible de lancer une commande car :
<txt>
```

avec <txt> pouvant être un ou plusieurs message parmi :

✓ « il y a déjà {X} calculs en cours (>= {Y}) »

avec {X} : le nombre de calculs en cours d'exécution

avec {Y} : le nombre maximum de calculs autorisés, défini dans le fichier «.ini » ou «.toml »

✓ « la charge cpu est {X}% (> {Y}%) »

avec {X} : le pourcentage CPU en cours

avec {Y} : le pourcentage maximum autorisé, défini dans le fichier «.ini » ou «.toml »

✓ « la mémoire disponible est {X} Mo (< {Y} Mo) »

avec {X} : le nombre de Mo libres

avec {Y} : le nombre minimum de Mo libres, défini dans le fichier «.ini » ou «.toml »

Voici un exemple complet de 4 calculs demandés, mais avec un seul calcul possible en parallèle :

```
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 4 | 0 | 0 / 4 (+0.0s)
INFO [ 86] Execution [calcul1]
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 3 | 1 | 0 / 4 (+0.1s)
INFO [ 86] [WARN-SUPERVISEUR-PIPt] Impossible de lancer une commande car : la
charge cpu est 100% (> 80%) ;
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 3 | 0 | 1 / 4 (+8.7s)
INFO [ 86] Execution [calcul2]
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 2 | 1 | 1 / 4 (+8.8s)
INFO [ 86] [WARN-SUPERVISEUR-PIPt] Impossible de lancer une commande car : la
charge cpu est 100% (> 80%) ;
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 2 | 0 | 2 / 4 (+24.3s)
INFO [ 86] Execution [calcul3]
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 1 | 1 | 2 / 4 (+24.5s)
INFO [ 86] [WARN-SUPERVISEUR-PIPt] Impossible de lancer une commande car : la
charge cpu est 100% (> 80%) ;
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 1 | 0 | 3 / 4 (+27.4s)
INFO [ 86] Execution [calcul4]
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 0 | 1 | 3 / 4 (+27.5s)
INFO [ 86] [INFO-SUPERVISEUR-PIPt] calculs prevision : 0 | 0 | 4 / 4 (+31.8s)
```

3.3 FAQ (Foire Aux Questions)

Comment paramétrer les répertoires de nettoyage ?

Il faut les déclarer un par un, sous forme de section dans le fichier «.ini », avec les clefs indiquées au chapitre 3.2.1, puis les déclarer sous la clef « [cleaning] → cleaners » (séparés par des « , »).
Exemple :

```
[cleaning]
cleaners=entrees,sorties,echanges

[entrees]
directory=C:\GRP_2016\Modeles\BV\Temps_Reel\Entrees
filter=*.txt

[sorties]
directory=C:\GRP_2016\Modeles\BV\Temps_Reel\Sorties

[echanges]
directory=C:\GRP_2016\Echanges\SESSION_TR
age=1
```

Comment simplifier les différents fichiers «.ini » pilotés depuis la POM ?

Il suffit de créer un fichier «.ini » complet (appelons le « base.ini »), qui renseigne toutes les clefs obligatoires. Une fois ceci fait, on peut créer un autre fichier «.ini » (appelons le « bassin1.ini »), dans le même répertoire, qui vient écraser certaines propriétés de « base.ini ». C'est le fichier « bassin1.ini » qui est renseigné côté POM. Exemple :

```
[general]
include=base.ini

[pipt]
modeles=00sPLA_SCS_3ssBV
```

Attention de bien formater ces .ini avec un encodage UTF-8 (sans BOM).

4. PIG

4.1 Installation

Ce chapitre ne traite que de l'installation du PIG dans ses versions postérieures ou égales à 4.0. PIG 4 supporte les version GRP : 2018, 2020 et 2021.

4.1.1 Système d'exploitation

Le fonctionnement du PIG / GRP est testé et validé avec la distribution Linux Debian dans sa version 10. Une VM spécifique est mise à disposition des SPC mais une installation directe de Debian 10 peut également être réalisée par les SPC. Toutefois l'utilisation de la VM diffusée est fortement préconisée.

La VM mise à disposition contient les pré-requis nécessaire au fonctionnement de PIG / GRP :

- Python 3.7
- le logiciel R dans sa version 3.5.2-1 : <https://cran.r-project.org/>

4.1.1.1 Caractéristiques de la VM

- 1 CPU
- 2048 Mo de mémoire
- 16 Go de disque dur

4.1.1.2 Utilisateurs et groupes

Deux groupes et utilisateurs sont disponibles :

- Le groupe **grp_pom** contient l'utilisateur d'exécution du PIG et de GRP, il est également le propriétaire des composants installés.
- Le groupe **exploit**

4.1.1.3 Contenu de la VM

```
$ ssh grp\_pom@xx.xx.xx.xx

$ ls
Desktop  Documents  Downloads  modeles  Music  Pictures  PIG  Public
Templates  uploads      Videos
```

Les dossiers utilisés par le PIG et GRP sont :

- **/home/grp_pom/modeles** : contient les modèles installés
 - **/home/grp_pom/modeles/GRP** : contient la version/les versions de GRP
- **/home/grp_pom/PIG** :
 - **/home/grp_pom/PIG/echanges** : répertoire utilisé par la POM pour fournir les données nécessaires à chaque calculs par GRP

- **/home/grp_pom/PIG/modeles** : ce répertoire contient la configuration des modèles utilisés par le PIG et GRP
- **/home/grp_pom/uploads** : le répertoire de dépôt des archives à utiliser pour installer le PI, le PIG et GRP

4.1.2 Pré-requis

Le PIG nécessite :

- la configuration du compte **grp_pom**, cf 2.3.2.3 - Modification du fichier `.profile` pour les VM GRP
- la création d'un environnement virtuel, cf 2.4 - Création et activation d'un environnement virtuel

Ces pré-requis sont à réaliser avec le compte **grp_pom**.

Le téléchargement de paquets est nécessaire à l'installation du PI, la VM doit donc être configurée par le SPC pour pouvoir accéder à internet.

4.1.3 Procédure d'installation

La procédure est très semblable à celle du PI sous Linux (cf. 3.1.1 Sous Linux) : télécharger l'archive d'installation, et lancer la commande d'installation.

Pour cela :

- ✓ Télécharger l'archive d'installation
 - depuis le site collaboratif du projet POM (<https://travail-collaboratif.din.developpement-durable.gouv.fr/share/page/site/dgprsmhshapipom/dashboard>), dans l'espace documentaire ==> Interfaçage avec les modèles ==> Plateformes nationales ==> La POM pilote GRP
 - déposer les fichiers **pominterface-4.0.XX.whl** et **pig-4.0.XX.whl** dans le répertoire :

```
/home/grp_pom/uploads
```

- ✓ Se connecter sur la VM avec le compte **grp_pom** activer l'environnement virtuel

Nous supposons dans la suite que le nom de l'environnement virtuel est « venv-pig-py3 »

```
$ ssh grp_pom@xx.xx.xx.xx
$ workon venv-pig-py3
(venv-pig-py3) $
```

- ✓ Installer le PIG

```
(venv-pig-py3) $ cd ~/uploads
(venv-pig-py3) $ pip install pominterface-4.0.XX.whl [proxy]
(venv-pig-py3) $ pip install pig-4.0.XX.whl [proxy]
```

Le paramètre `[proxy]` est à définir en fonction de la configuration du réseau local. Ce paramètre peut être vide ou de la forme :

```
(venv-pig-py3) $ pip install pig-4.0.XX.whl
--proxy=http://monproxy:ProxyPort
```

Exemple avec le proxy du ministère :

```
(venv-pig-py3) $ pip install pig-4.0.XX.whl --proxy=http://pfrie-std.proxy.e2.rie.gouv.fr:8080
```

Ne pas tenir compte des messages d'avertissement suivants :

WARNING: You are using pip version 20.1.1; however, version 23.1.2 is available.

- ✓ Contrôler les packages installés dans l'environnement virtuel

```
(venv-pig-py3) $ pip list
Package          Version
-----
lxml              4.9.2
numpy             1.21.6
pandas            1.3.5
pig             4.0.0rc0
pip              20.1.1
pominterface    4.0.0rc0
python-dateutil  2.8.2
pytz              2023.3
setuptools        57.0.0
shapely           2.0.1
six               1.16.0
tomli             2.0.1
wheel             0.36.2
```

- ✓ Vérifier l'installation du PIG et des outils du PI

```
(venv-pig-py3) $ pig --version
PIG 4.0.0RC0, PI 4.0.0RC1

(venv-pig-py3) $ checkini --help
INFO:pominterface.tools.checkini:Outil de vérification des .ini
(version PI : 4.0.0RC1)
usage: checkini [-h] ini_file [ini_file ...]

checkini executable

positional arguments:
  ini_file      Provide the ini file path to be checked

optional arguments:
  -h, --help    show this help message and exit
```

Il faut ensuite personnaliser le paramétrage du PIG :

en copiant sur le répertoire des modèles GRP

/home/grp_pom/modeles/GRP/GRP2020

en modifiant le fichier model_pig.ini, en particulier, les chemins relatifs à l'installation de la base temps réel de GRP

Attention de bien formater ces .ini avec un encodage UTF-8 (sans BOM).

Comme suggéré au chapitre 3.3 FAQ (Foire Aux Questions) Comment simplifier les différents fichiers « .ini » pilotés depuis la POM ?), il est possible de créer :

- un fichier **pig-base.ini** commun à tous les PIG
- un fichier **modelpig.ini** à dupliquer en **pig_Bvxxx.ini** (1 par modèle POM) :
 - puis d'ajouter la ligne suivante au début au fichier **pig_Bvxxx.ini**

```
include='pig-base.ini'
```

- et de ne mettre dans le fichier **pig_Bvxxx.ini** que les paramètres différents du fichier **pig-base.ini**

4.1.4 Configuration à faire sur la POM

La **plateforme** côté POM doit être configurée ainsi :

- ✓ Chemin complet de l'exécutable à lancer : **/home/grp_pom/.virtualenvs/XXX/bin/pig**
- ✓ Chemin du répertoire des fichiers d'échange : **/home/grp_pom/PIG/echanges**
- ✓ Paramètres de commande : **\$f /path/to/the/modelpig.ini \$e**
- ✓ Supporte les lancements groupés de modèles : **non**

Les paramètres de la commande sont décrit en détail en 4.2.3 - Ligne de commande.

Le **serveur de calcul** côté POM doit être configuré ainsi :

- ✓ Identifiant : **grp_pom**
- ✓ Mot de passe : **password**
- ✓ Type de serveurs : **Serveur de calcul**
- ✓ Système d'exploitation : **Linux**
- ✓ Port : **22**

4.2 Exploitation

4.2.1 Mode de calcul

Le mode de calcul du PIG peut être renseigné dans le champ « mode de calcul » du scénario principal de la POM. Il contient des mots clefs, séparés par des espaces, et dont la valeur est renseignée comme suit :

```
MOT_CLEF=VALEUR
```

Tous les mots clefs sont facultatifs. S'ils ne sont pas renseignés, ils sont pris égaux à leur valeur par défaut.

La liste des mots clefs est la suivante :

MOT CLEF	VALEUR PAR DÉFAUT (VALEURS POSSIBLE)	DESCRIPTION
----------	---	-------------

type	DET DET INC D+INC AbaquesSeules	Type de résultat attendu <ul style="list-style-type: none"> ✓ DET : résultats déterministes uniquement (par défaut si non renseigné) ✓ INC : incertitudes uniquement ✓ D+INC : résultats déterministes et incertitudes ✓ AbaquesSeules : calcul des abaques uniquement
abaques	NON NON MANUEL AUTO	Résultats avec abaques <ul style="list-style-type: none"> ✓ NON : pas de lancement des abaques (par défaut si non renseigné) ✓ MANUEL : l'initialisation pour le calcul des abaques est effectuée si le paramètre « initialisation du modèle » est indiqué à « true » dans le fichier parameters.xml. <p>Si l'initialisation est effectuée avec succès, lancement du calcul des abaques .</p> ✓ AUTO : l'initialisation pour le calcul des abaques est effectuée. <p>Si l'initialisation est effectuée avec succès, lancement du calcul des abaques</p>
codes	Codes séparés par des « »	Codes des sites hydro à piloter.
ScenarioGRP		Code scénario à attribuer dans GRP pour ce scénario POM. Ce doit être un entier de 4 chiffres maximum. Ce paramètre doit être renseigné sur tous les scénarios POM (y compris le principal) ou aucun. Toutes les valeurs de ce paramètre (sur les différents scénarios POM) doivent être différents.
pdT		Liste des pas de temps des sites hydro de sortie : <ul style="list-style-type: none"> ✓ La valeur est une liste de couples (code du site de sortie, valeur du pas de temps) ✓ Le caractère « » est utilisé pour séparer les couples ✓ Le caractère « ; » est utilisé pour séparer le code du site hydro de sortie de la valeur du pas de temps ✓ Les valeurs de pas de temps sont au format « nnJnnHnnM », où nn correspond à l'occurrence, J, H, M correspondent respectivement à Jour, Heure, Minute. Ce format utilisé est exactement le même que celui défini pour GRP 2018, cf. [EVO_GRP_2018] §1.2.3 Paramétrer les pas de temps. <p>Exemple : pdT=I3422010;00J01H00M I3442310;00J00H30M</p>
Assimilation	OUI	SEULEMENT POUR GRP >= 2020 ; Elle est

	OUI	ignorée avec GRP2018.
	NON	Calcul GRP avec ou sans assimilation

Tableau 4: modes de calcul du PIG

4.2.2 Fichier « .ini »

Le fichier « .ini » contient les clefs du PI (cf. 3.2.1) et les clefs présentées ci-après.

Attention de bien formater ces .ini avec un encodage UTF-8 (sans BOM).

SECTION	CLEF	OBLIGATOIRE	FORMAT	DESCRIPTION
grp	grpversion	Oui	Chaîne	« 2018 » ou « 2020 » ou « 2021 »
	parameterdirectory	Oui	Chaîne	Nom du répertoire de paramétrage GRP.
	listebassinfile	Oui	Chaîne	Nom du fichier des bassins de GRP.
	configprevifile	Oui	Chaîne	Nom du fichier « config_prevision.ini » de GRP.
	inputdirectory	Oui	Chaîne	Nom du répertoire des entrées
	resultdirectory	Oui	Chaîne	Nom du répertoire des sorties
	resultfilename	Oui	Chaîne	Nom du fichier des résultats (Previsions.txt)
	facteurintensite	Oui	Entier	Facteur d'intensité des pluies reconstituées.
	bdbassindir	Oui	Chaîne	Nom du répertoire de la BD bassins
	resultdirectorypdf	Oui	Chaîne	Nom du répertoire contenant les fiches PDF (le plus souvent Sorties\Fiches_Control)
	incertituderesultdirectory	Oui	Chaîne	Répertoire des fichiers d'incertitude
Abaques	rootdirectoryabaques	Oui	Chaîne	Chemin complet du répertoire racine des abaques.
	configfileabaques	Oui	Chaîne	Nom du fichier de paramétrage des abaques
	commandlineabaquesgrp	Oui	Chaîne	Nom de l'exécutable GRP (relativement à la racine des abaques)
	resultdirectoryabaques	Oui	Chaîne	Nom du répertoires des résultats des abaques (relativement à la racine des abaques)
	siteshydroabaques	Oui	Chaîne	Codes des sites hydro par défaut à lancer, s'ils ne sont pas renseignés côté POM.
	initbpressourcecode mask	Oui	Chaîne	Masque du codes de la métadonnée BP utilisée pour initialiser les abaques.

Tableau 5: fichier « .ini » du PIG

4.2.3 Ligne de commande

Le lancement en ligne de commande s'effectue comme suit :

```
/home/grp_pom/.virtualenvs/XXX/bin/pig {FICHIER_PARAMETERS}
[ {FICHIER_INI} ] [ --force ] [ --logs ]
```

où :

- ✓ XXX est le nom de l'environnement virtuel du PIG
- ✓ {FICHIER_PARAMETERS} est le chemin complet du fichier parameters.xml fourni par la POM
- ✓ {FICHIER_INI} (facultatif) est le nom du fichier « .ini » à utiliser pour ce calcul. Il doit exister dans le même répertoire que le « __main__.py ». S'il n'est pas renseigné, le fichier par défaut « modelpig.ini » est utilisé.
- ✓ --force (facultatif) permet de forcer un lancement sans tenir compte des tests de calculs concurrents.
- ✓ --logs (facultatif) permet de forcer la génération des fichiers de log pour debug.

4.2.4 Calculs concurrents

GRP ne peut pas s'exécuter plusieurs fois en parallèle. Le PIG s'assure donc qu'il n'est pas déjà lancé avant de s'exécuter.

Il utilise pour cela la mécanique générique du PI (cf. 3.2.2).

4.3 FAQ (Foire Aux Questions)

Le PIG utilise-t-il le PI ?

Oui à partir du PIG v2.0.

Le PIG prend-il en charge les fichiers .PRE ?

Non à partir du PIG v2.0.

Le PIG peut-il être lancé plusieurs fois en parallèle ?

Non car GRP ne le permet pas. Le PIG s'en assure lors de son exécution.

5. PIPT

- ✓ Ce chapitre décrit l'installation du PIPT à partir de sa version 4 dans une VM DEBIAN 9.7 ou DEBIAN 10.

5.1 Installation du PIPT

5.1.1 Système d'exploitation

Le fonctionnement du PIPT est testé et validé avec une distribution Linux Debian dans une version 9 au minimum.

La VM doit disposer de Python 3.7.

Pour l'installation des dépendances, il est nécessaire que la VM ait un accès Internet.

5.1.2 Pré-requis

Le PIPT nécessite :

- la création d'un environnement virtuel 'venv-pipt-py3' (cf 2.3 - Installation des utilitaires Python).

Ces pré-requis sont à réaliser avec le compte **plathynes_pom**.

Les autres pré-requis sont déjà effectués sur la VM livrée. C'est pourquoi nous conseillons fortement d'utiliser cette VM.

•

5.1.3 Procédure d'installation

- ✓ déposer les fichiers **pominterface-4.0.XX.whl** et **pipt-4.0.XX.whl** dans le répertoire :

```
/home/exploit/uploads
```

- ✓ Se connecter sur la VM avec le compte **exploit** (car ce compte permet d'accéder à internet) et activer l'environnement virtuel

Nous supposons dans la suite que le nom de l'environnement virtuel est « venv-pipt-py3 »

```
$ ssh exploit@xx.xx.xx.xx
$ workon venv-pipt-py3
(venv-pipt-py3) $
```

- ✓ Installer le PIPT

```
(venv-pipt-py3) $ cd ~/uploads
(venv-pipt-py3) $ pip install pominterface-4.0.XX.whl [proxy]
(venv-pipt-py3) $ pip install pipt-4.0.XX.whl [proxy]
```

Le paramètre [proxy] est à définir en fonction de la configuration du réseau local. Ce paramètre peut être vide ou de la forme :

```
(venv-pipt-py3) $ pip install pim-4.0.XX.whl
--proxy=http://monproxy:ProxyPort
```

Exemple avec le proxy du ministère :

```
(venv-pipt-py3) $ pip install pim-4.0.XX.whl
--proxy=http://pfrie-std.proxy.e2.rie.gouv.fr:8080
```

Ne pas tenir compte des messages d'avertissement suivants :

WARNING: You are using pip version 20.1.1; however, version 20.2.1 is available.

- Contrôler les packages installés dans l'environnement virtuel

```
(venv-pipt-py3) $ pip list
Package            Version
-----
lxml                4.9.2
numpy               1.21.6
pandas              1.3.5
pip                 20.1.1
pipt               4.0.0rc0
pominterface      4.0.0rc0
python-dateutil     2.8.2
pytz                2023.3
setuptools          57.0.0
six                 1.16.0
tomli                2.0.1
wheel               0.36.2
```

Création de l'arborescence

Lors de l'installation du pipit, l'arborescence nécessaire au fonctionnement de plathynes réel n'est pas automatiquement créée, il faut donc créer manuellement sur la VM l'arborescence suivante :

```
/home/plathynes_pom
| - PIPT
|   | - echanges
|   | - MODELES
|   | - WORKDIR
```

5.1.4 Configuration à faire sur la POM

La **plateforme** côté POM doit être configurée ainsi :

- ✓ Chemin complet de l'exécutable à lancer :

/home/plathynes_pom/pipt/virtualenvs/venv-pipt-py3/bin/pipt

5.1.5 Configuration pour les fichiers « .ini »

Pour gérer les variables d'environnement spéciales nécessaires au solveur Plathynes, il faut utiliser les sections [setenv] et [appendenv] du fichier « .ini ».

Exemple :

```
[setenv]
MARINE_DIR = /home/plathynes_pom/modeles/plathynes/CODE/
PLATHYNES_DIR = /home/plathynes_pom/modeles/plathynes/CODE/
LD_LIBRARY_PATH =
/home/plathynes_pom/modeles/plathynes/CODE/lib/:
/home/plathynes_pom/modeles/plathynes/CODE/bin

[appendenv]
PYTHONPATH = /home/plathynes_pom/modeles/plathynes/CODE/:
/home/plathynes_pom/modeles/plathynes/CODE/src:
/home/plathynes_pom/modeles/plathynes/CODE/lib
```

5.2 Installation du solveur PLATHYNES

Afin de simplifier les installations nécessaires au fonctionnement de PLATHYNES temps-réel un script a été développé pour l'installation et les mises à jour du noyau de calcul PLATHYNES.

5.2.1 Prérequis – Système d'exploitation

Le fonctionnement de PLATHYNES temps-réel est testé et validé avec la distribution Linux Debian dans sa version 9.7 et 10. Une VM spécifique est mise à disposition des SPC mais une installation directe de DEBIAN 9.7 ou DEBIAN 10 peut également être réalisée par les SPC.

*Attention, avant de lancer l'installation du noyau Plathynes, il faut mettre à jour le script **install_solveur_vX.Y.Z.sh** avec les informations du serveur d'installation.*

```
# le répertoire d'installation du coeur de calcul Plathynes
PLATHYNES_PATH=/home/xxxx/modeles/plathynes

# le user d'exécution du coeur de calcul
USERPLATHYNES=xxxx
```

S'il s'agit d'une première installation, l'installation est à faire en « root », car elle va créer l'utilisateur « plathynes_pom » et installer les packages « gfortran » et « quadmath ».

Pour les installations suivantes, cette installation est à faire avec le compte « exploit ».

```
# tar xf install_solveur_X.Y.Z.tar.gz
# cd install_solveur_X.Y.Z
# ls
deploiement_plathynes  gfortran  install_solveur_v1.8.4.sh

# ./install_solveur_vX.Y.Z.sh
```

5.2.1.1 Caractéristiques de la VM PLATHYNES

- 1 CPU,
- 2048 Mo de mémoire
- 16 Go de disque dur

Ces caractéristiques sont minimales, la mémoire et l'espace disque pourront être augmentés en fonction du nombre de modèles PLATHYNES à piloter.

5.2.1.2 Utilisateurs et groupes

Deux groupes et utilisateurs sont nécessaires pour l'utilisation temps-réel :

- Le groupe **exploit (ou un autre nom choisi à l'installation de DEBIAN)** est le propriétaire des composants installés.
- Le groupe **plathynes_pom** contient l'utilisateur d'exécution du cœur de calcul Plathynes. Il sera créé lors de l'installation du solveur.

Fournitures

La mise en place de PLATHYNES temps-réel nécessite :

- L'archive du noyau de calcul PLATHYNES **install_solveur_X.X.X.tar.gz**

5.2.2

5.2.3 Solveur PLATHYNES

5.2.3.1 Contenu de l'archive du noyau

L'archive du cœur de calcul Plathynes contient :

- ✓ un script d'installation,
- ✓ les librairies nécessaires au système,
- ✓ les binaires,
- ✓ les sources Plathynes.

Les binaires du cœur de calcul de Plathynes sont :

- ✓ **prepro**
- ✓ **solver**

Les librairies nécessaires sont fournies sous forme de paquets compatibles avec la version Debian 9.7, il s'agit de :

- ✓ la bibliothèque **libquadmath0** qui fournit des fonctions mathématiques quadruple précision, elle est utile pour le compilateur Fortran GNU
- ✓ Le **runtime** du compilateur Fortran GNU

libquadmath0	libquadmath0_6.3.0-18+deb9u1_amd64.deb

Fortran	libgfortran3_6.3.0-18+deb9u1_amd64.deb
----------------	--

Tableau 6: paquets Debian nécessaires au cœur de calcul Plathynes

Les sources sont fournies car le PIPT requiert des fichiers de propriété présents dans les sources du solveur.

5.2.3.2 Installation

Copier **install_solveur_X.Y.Z.tar.gz** dans /home/exploit/ puis en tant que root effectuer les commandes suivantes :

```
# tar xf install_solveur_X.Y.Z.tar.gz
# cd install_solveur_X.Y.Z
# ls
deploiement_plathynes  gfortran  install_solveur_v1.8.4.sh

# ./install_solveur_vX.Y.Z.sh

## Debut de l'installation
## Creation du user plathynes_pom
Entrez le nouveau mot de passe UNIX :
Retapez le nouveau mot de passe UNIX :
passwd: password updated successfully
## Installation de GFORTRAN
Installation de libgfortran3
Sélection du paquet libgfortran3:amd64 précédemment
désélectionné.
(Lecture de la base de données... 27936 fichiers et répertoires
déjà installés.)
Préparation du dépaquetage de .../libgfortran3_6.3.0-
18+deb9u1_amd64.deb ...
Dépaquetage de libgfortran3:amd64 (6.3.0-18+deb9u1) ...
Sélection du paquet libquadmath0:amd64 précédemment
désélectionné.
Préparation du dépaquetage de .../libquadmath0_6.3.0-
18+deb9u1_amd64.deb ...
Dépaquetage de libquadmath0:amd64 (6.3.0-18+deb9u1) ...
Paramétrage de libquadmath0:amd64 (6.3.0-18+deb9u1) ...
Paramétrage de libgfortran3:amd64 (6.3.0-18+deb9u1) ...
Traitement des actions différées (« triggers ») pour libc-bin
(2.24-11+deb9u3) ...
Le Package libquadmath0 est déjà installé
## Creation de l'arborescence
```

Au cours de l'installation, il faut définir et confirmer le mot de passe de l'utilisateur **plathynes_pom**,

Le script **install_solveur_vX.Y.Z.sh** contient des variables d'environnement qui permettent de configurer son comportement. Par défaut, il n'est pas nécessaire de modifier ces valeurs.

```
# choix d'installer ou non le coeur de calcul Plathynes
INSTALL_SOLVEUR=true

# la version du coeur de calcul installé
VERSION_SOLVEUR="1.8.7"

# le positionnement d'un lien courant pour le coeur de calcul
installé
SET_LINK_CURRENT_SOLVEUR=true

# le répertoire d'installation du coeur de calcul Plathynes
PLATHYNES_PATH=/home/plathynes_pom/modeles/plathynes

# le user d'exécution du coeur de calcul
USERPLATHYNES=plathynes_pom
```

5.3 Exploitation

5.3.1 Dépôts des exports PLATHYNES

Le dépôt des exports se fait à l'aide d'un script spécifique **install_session_plathynes.sh** avec l'utilisateur **plathynes_pom**.

Copier le script dans **/home/plathynes_pom**

Créer un répertoire « uploads » dans **/home/plathynes_pom** et y copier le fichier .tar.gz de l'export plathynes étude

Revenir dans **/home/plathynes_pom** et lancer le script **install_session_plathynes.sh** **SESSION** **CODE**

avec **SESSION** : chemin complet vers l'archive session au format .tar.gz (normalement **/home/plathynes_pom/uploads**)

et **CODE** : le code du modèle PLATHYNES (voir 5.3.2)

exemple :

```
bash install_session_plathynes.sh
/home/exploit/uploads/session.tar.gz 00sPLA0070
```

Le script extrait l'archive et dépose les fichiers exportés dans :

/home/plathynes_pom/PIPt/MODELES/[CODE MODELE]/SESSION

Chaque répertoire de modèle est structuré comme suit :

```
{RACINE_EXPORT}
| - {CODE_MODELE_1}
|   | - SESSION
|   |   | - {NOM_BASSIN}.mwc
|   |   | - parameters (répertoire)
|   |   | - session.xml
| - {CODE_MODELE_2}
|   | - SESSION
|   |   | - {NOM_BASSIN}.mwc
```

		- parameters (répertoire)
		- session.xml

5.3.2 Conseils pour le codage des modèles PLATHYNES

Il est conseillé de définir des codes modèles PLATHYNES qui reprennent les codes modèles POM auquel ils sont associés.

Ainsi, sur les quatre derniers caractères libres des codes modèles, trois pourraient être utilisés pour identifier le modèle POM et le dernier pour identifier le modèle PLATHYNES.

Exemple :

le modèle POM 00sPLA012 pourraient permettre de lancer au maximum 10 modèles PLATHYNES nommés 00sPLA0120, 00sPLA0121, 00sPLA0122, ..., 00sPLA0129.

5.3.3 Arborescence des fichiers

Le PIPT gère une arborescence de fichiers de travail qui permet d'éviter la concurrence entre deux calculs.

Cette arborescence est organisée comme suit :

```

PIPT_WORKDIR
|- Archives
|   |- {CODE_MODEL_PLATHYNES}
|   |-
|   |   |- {DATE_PIVOT}__{MODE_CALCUL_POM}__{SCENARIO_POM}__{CODE_SESSION_POM}__{D
ET|ENS_XXX}{__ASSIM}.zip
|   |   |- {CODE_SESSION_POM}
|   |   |   |- {DATE_PIVOT}
|   |   |   |   |- {MODE_CALCUL_POM}
|   |   |   |   |   |- {CODE_SCENARIO_POM}
|   |   |   |   |   |   |- {CODE_MODELE_PLATHYNES}
|   |   |   |   |   |   |   |- analyse
|   |   |   |   |   |   |   |   |- {NOM_BASSIN}.mwc
|   |   |   |   |   |   |   |   |- {NOM_BASSIN}.net
|   |   |   |   |   |   |   |   |- parameters
|   |   |   |   |   |   |   |   |- ...
|   |   |   |   |   |   |   |   |- analyse.pid
|   |   |   |   |   |   |   |   |- analyse.sim
|   |   |   |   |   |   |   |   |- ev_Analyse
|   |   |   |   |   |   |   |   |   |- data
|   |   |   |   |   |   |   |   |   |   |- *.csv
|   |   |   |   |   |   |   |   |   |   |- Relay
|   |   |   |   |   |   |   |   |   |   |   |- Relay_*.txt
|   |   |   |   |   |   |   |   |   |   |   |- Analyse.evt
|   |   |   |   |   |   |   |   |   |   |   |- *.mqi
|   |   |   |   |   |   |   |   |   |   |   |- *.mqo
|   |   |   |   |   |   |   |   |   |   |   |- *.mrr
|   |   |   |   |   |   |   |   |   |   |   |- Results
|   |   |   |   |   |   |   |   |- prevision
|   |   |   |   |   |   |   |   |   |- {NOM_BASSIN}.mwc
|   |   |   |   |   |   |   |   |   |- {NOM_BASSIN}.net
|   |   |   |   |   |   |   |   |   |- parameters
|   |   |   |   |   |   |   |   |   |- ...
|   |   |   |   |   |   |   |   |   |- prevision.pid
|   |   |   |   |   |   |   |   |   |- prevision.sim
|   |   |   |   |   |   |   |   |   |- ev_Prevision
|   |   |   |   |   |   |   |   |   |   |- data
|   |   |   |   |   |   |   |   |   |   |   |- *.csv

```


Tous les mots clefs sont facultatifs. S'ils ne sont pas renseignés, ils sont pris égaux à leur valeur par défaut.

La liste des mots clefs est la suivante :

MOT CLEF	VALEUR PAR DÉFAUT (VALEURS POSSIBLE)	DESCRIPTION
num_ensemble	Un entier compris en 2 et 999	Nombre d'ensemble à calculer dans le cas d'une prévision d'ensemble
mode	DET (valeur par défaut)	Type de résultat attendu le calcul est lancé en mode « déterministe » sans assimilation
	ENS	le calcul est lancé en mode « ensemble » sans assimilation
assimilation	OUI (valeur par défaut) NON	Résultats avec ou sans assimilation
analyse	Un code de scénario POM	Code du scénario POM d'analyse à utiliser pour initialiser ce calcul. Uniquement valable pour les scénarios POM dédiés à un run de prévision sans analyse.

Tableau 7: mode de calcul du PIPT

5.3.5 Fichier « .ini »

Le fichier « .ini » contient les clefs du PI (cf. 3.2.1) et les clefs présentées ci-après.

Le fichier « ini » peut faire l'objet d'une surcharge comme indiqué dans le PI (cf. 3.2.1).

SECTION	CLEF	OBLIGATION	FORMAT	DESCRIPTION
pipt	versionplathynes	Oui	Chaîne	Version de PLATHYNES à piloter.
	workdir	Oui	Chaîne	Répertoire de travail du PIPT, pour entreposer les fichiers temporaires.
	eventduration	Oui	Entier	Nombre de minutes intervenant dans le calcul du cumul de pluie pour détecter la survenue d'un événement à continuer.
	eventmax	Oui	Réel	Cumul minimum permettant de détecter un événement en cours (sur la durée eventduration).
	grdformat_filename	Non	Chaîne	Masque du nom des fichiers GRD à chercher dans les métadonnées de type fichier ZIP envoyées par la POM.
	grdformat_date	Non	Chaîne	Format de la date contenue dans les noms des fichiers GRP contenus dans

				les fichiers ZIP envoyées par la POM.
	incertitudes	Non	Chaîne	Permet d'indiquer si les résultats seront des membres ou des quantiles : valeurs possibles : « membres », « quantiles » Valeur par défaut « membres »
	quantiles	Non	Chaîne	Une liste de quantiles : pourcentages (entiers entre 0 et 100 suivi de %) ou « MIN » ou « MOY » ou « MAX » séparés par des « », Valeur par défaut « 10 % 50 % 90 % »

Tableau 8: fichier « .ini » du PIPt

Le paramètre « grdformat_filename » est une expression régulière au format Python, utilisable dans la fonction « re.match » (cf. :<https://docs.python.org/2/library/re.html#regular-expression-objects>).

Le paramètre « grdformat_date » est un format de date utilisable dans la fonction « datetime.strptime » (<https://docs.python.org/2/library/datetime.html#strptime-behavior>).

5.3.6 Calculs concurrents

Le PIPt peut s'exécuter plusieurs fois simultanément et est paramétré en ce sens, « lancementsimultanes = true ».

Les fichiers de verrou « analyse.pid » et « prevision.pid » sont créés dans le répertoire courant de travail du PIPt afin qu'aucun autre lancement du PIPt ne puisse venir le concurrencer.

Vue l'arborescence de travail du PIPt (cf §5.3.3), ces fichiers verrous permettent d'interdire des lancements simultanés dans un contexte identique très précis, même date pivot, même scénario, même mode de calcul, ...

5.3.7 Ménage automatique

Le PIPt est paramétré pour faire le ménage après avoir lancé ses calculs. Cela permet d'éviter que des fichiers d'un calcul concurrent soient supprimés par inadvertance.

5.4 FAQ (Foire Aux Questions)

Le PIPt utilise-t-il le PI ?

Non pour le PIPt v1.0, oui à partir du PIPt v2.0.

Le PIPt peut-il être lancé plusieurs fois en parallèle ?

Oui dans la plupart des cas. Il existe cependant des cas très spécifiques où le PIPt interdira un calcul car les fichiers lus ou produits par le PIPt seraient en conflit ou écrasés.

6. PIT

6.1 Installation

Ce chapitre traite de l'installation du PIT dans ses versions postérieures ou égales à 4.0 et de Telemac V8P4 sur la VM Debian 11.

6.1.1 Pré-requis

Le PIT 4 nécessite :

- la création d'un environnement virtuel (cf 2.3 - Installation des utilitaires Python).

Ces pré-requis sont à réaliser avec le compte **schapi**.

Nous supposons pour la suite du document que les dossiers suivants existent dans la VM :

```
/home/schapi/
|- pom
  |- echanges
  |- pit
    |- install
    |- workspace (répertoire de travail PIT)
  |- uploads
```

6.1.2 Procédure d'installation

✓ Télécharger l'archive d'installation

- depuis le site collaboratif du projet POM (<https://travail-collaboratif.din.developpement-durable.gouv.fr/share/page/site/dgprsrnhshapipom/dashboard>), dans l'espace documentaire ==> Interfaçage avec les modèles ==> Plateformes nationales
- déposer les fichiers **pominterface-4.0.xx.whl** et **pit-4.0.xx.whl** dans le répertoire :

```
/home/schapi/uploads
```

✓ Se connecter sur la VM avec le compte **schapi** et activer l'environnement virtuel

Nous supposons dans la suite que le nom de l'environnement virtuel est « venv-pit-py3 »

```
$ ssh schapi@xx.xx.xx.xx
$ workon venv-pit-py3
(venv-pit-py3) $
```

✓ Installer le PIT

```
(venv-pit-py3) $ cd ~/uploads
```

```
(venv-pit-py3) $ pip install pominterface-4.0.XX.whl [proxy]
(venv-pit-py3) $ pip install pit-4.0.XX.whl [proxy]
```

Le paramètre [proxy] est à définir en fonction de la configuration du réseau local. Ce paramètre peut être vide ou de la forme :

```
(venv-pit-py3) $ pip install pit-4.0.XX.whl
-proxy=http://monproxy:ProxyPort
```

Exemple avec le proxy du ministère :

```
(venv-pit-py3) $ pip install pit-4.0.XX.whl -proxy=http://pfrie-
std.proxy.e2.rie.gouv.fr:8080
```

Ne pas tenir compte des messages d'avertissement suivants :

WARNING: You are using pip version 20.1.1; however, version 23.1.2 is available.

Remarque : Par nécessité de compatibilité avec le script « run_telfile.py » de Telemac, le PIT doit utiliser une version de numpy < 1.24.

✓ Contrôler les packages installés dans l'environnement virtuel

```
(venv-pit-py3) $ pip list
Package           Version
-----
lxml               4.9.2
numpy              1.21.6
pandas             1.3.5
pit               4.0.0rc0
pip                20.1.1
pominterface     4.0.0rc1
python-dateutil    2.8.2
pytz               2023.3
setuptools         57.0.0
six                1.16.0
tomli              2.0.1
wheel              0.36.2
```

✓ Vérifier l'installation du PIT et des outils du PI

```
(venv-pit-py3) $ pit --version
PIT 4.0.0RC0, PI 4.0.0RC1

(venv-pit-py3) $ checkini --help
INFO:pominterface.tools.checkini:Outil de vérification des .ini
(version PI : 4.0.0RC1)
usage: checkini [-h] ini_file [ini_file ...]

checkini executable
```

```
positional arguments:
  ini_file      Provide the ini file path to be checked

optional arguments:
  -h, --help  show this help message and exit
```

6.1.3 Configuration à faire sur la POM

La **plateforme** côté POM doit être configurée ainsi :

- ✓ Exécutable à lancer :

```
. /home/schapi/.bash_profile && /home/mascaret/virtualenvs/venv-pit-  
py3/bin/pit
```

- ✓ Chemin du répertoire des fichiers d'échange :

```
/home/schapi/pom/echanges
```

- ✓ Paramètres de commande :

```
$f /path/to/the/modelpit.ini $e
```

- ✓ Supporte les lancements groupés de modèles :

```
non
```

Le **serveur de calcul** côté POM doit être configuré ainsi :

- ✓ Identifiant : **schapi**
- ✓ Mot de passe : **password**
- ✓ Type de serveurs : **Serveur de calcul**
- ✓ Système d'exploitation : **Linux**
- ✓ Port : **22**

6.1.4 Mise à jour des variables d'environnement

Le lancement de Telemac V8P4 nécessite la présence de 2 variables d'environnement avec des valeurs spécifiques :

```
SYSTELCFG = /home/softs/V8P4R0/configs/systel.schapi.cfg  
LD_LIBRARY_PATH =  
/home/softs/V8P4R0/builds/gfortran/lib:/home/softs/libs/med-4.1.1/  
lib:/home/softs/libs/aed2_1.2/lib:/home/softs/libs/metis-5.1.0/lib/
```

Il est possible de définir ces variables de 2 façons :

- ✓ soit dans un fichier « .bashrc », exemple :

```
export SYSTELCFG=/home/softs/V8P4R0/configs/systel.schapi.cfg  
  
LD_LIBRARY_PATH=/home/softs/V8P4R0/builds/gfortran/lib:$LD_LIBRARY_PATH  
LD_LIBRARY_PATH=/home/softs/libs/med-4.1.1/lib:$LD_LIBRARY_PATH  
LD_LIBRARY_PATH=/home/softs/libs/aed2_1.2/lib:$LD_LIBRARY_PATH  
export LD_LIBRARY_PATH=/home/softs/libs/metis-5.1.0/lib:$LD_LIBRARY_PATH
```

- ✓ soit dans un fichier « .toml », exemple :

```
[setenv]
SYSTELCFG = "/home/softs/V8P4R0/configs/systel.schapi.cfg"

[appendenv]
LD_LIBRARY_PATH = "":/home/softs/V8P4R0/builds/gfortran/lib/:\
/home/softs/libs/med-4.1.1/lib/:\
/home/softs/libs/aed2_1.2/lib/:\
/home/softs/libs/metis-5.1.0/lib/""
```

Nous préconisons de définir ces variables dans un fichier « .bashrc » afin de l'appliquer à tous les modèles de calcul.

6.1.5 Mise à jour des modèles pour Telemac V8P4

Le passage à Telemac V8P4 impose de nettoyer les fichiers « .cas » des modèles :

- ✓ enlever les caractères accentués
- ✓ déplacer les commentaires de fin de ligne à la ligne suivante
- ✓ Si Telemac renvoie encore une erreur alors que le commentaire est à la ligne suivante, il faut supprimer le commentaire

6.2 Exploitation

6.2.1 Arborescence des fichiers

Le PIT gère une arborescence de fichiers de travail qui permet d'éviter la concurrence entre deux calculs.

Cette arborescence est organisée comme suit :

```
/home/schapi/pom/pit
|- workspace (répertoire de travail PIT)
  |- archives
    |- {CODE_MODEL_TELEMAC}
      |-
        {DT_PIVOT}_{DT_ARCHIVE}_{MODE_CALC_POM}_{SCENAR_POM}.pit.slf
      |- {CODE_SESSION_POM}
        |- {DATE_PIVOT}
          |- {MODE_CALCUL_POM}
            |- {CODE_SCENARIO_POM}
              |- {CODE_MODELE_TELEMAC}
                |- analyse
                  |- {NOM_SERIE_1}
                    |- {NOM_SERIE_2}
                      ...
```

Les conventions suivantes sont prises :

- ✓ Les dates sont au format « AAAAMMDD_HHMM » pour effectuer un tri alphabétique chronologique
- ✓ Le mode de calcul POM est lu dans la balise « session → mode » du fichier « parameters.xml ». Il vaut « REAL_TIME » s'il n'est pas renseigné.
- ✓ Le répertoire {CODE_MODELE_TELEMAC} est le nom du répertoire du modèle Télémac à lancer (clef [models]names du fichier « .ini »).
- ✓ Les répertoires « analyse » et « {NOM_SERIE_i} » correspondent aux différents runs à lancer dans Télémac. Ils contiennent les fichiers nécessaires au calcul.

{DT_ARCHIVE} est la date de référence de l'archivage. Cette date correspond au nom du fichier généré par 'export_date_chop'.

6.2.2 Mode de calcul

Le mode de calcul du PIM peut être renseigné dans le champ « mode de calcul » de chaque scénario (le principal et les éventuels complémentaires) de la POM.

La syntaxe est la suivante :

```
[series={SERIES}] [inits={INITS}] [init_files={INIT_FILES}]
init_cotes={INIT_COTES} [analyse={ANALYSE}]
```

Tous les mots clefs sont facultatifs. S'ils ne sont pas renseignés, ils sont pris égaux à leur valeur par défaut.

La liste des mots clefs est la suivante :

MOT CLEF	VALEUR PAR DÉFAUT (VALEURS POSSIBLE)	DESCRIPTION
series	MOY (MIN, MAX, MOY, x%, xx %, 100%)	Séries à propager. Sa valeur est une liste de noms séparés par des « ». Les noms désignent des séries de données qui sont renseignées sur les entrées du fichier « parameters.xml ». Le nom d'une série peut être un pourcentage (un entier suivi de %).
inits	REPRISE (REPRISE, DEFAULT, CONSTANTE)	Méthodes d'initialisation à lancer consécutivement et dans l'ordre indiqué. Sa valeur est une liste des codes de méthodes séparés par des « ». Chaque code ne peut apparaître qu'une fois
init_files	<vide>	Noms des fichiers du répertoire « backupdirectory » pour le mode d'initialisation « DEFAULT », séparés par des « ». Il n'est utilisé que si le mot clef « inits » contient « DEFAULT ».

init_cotes	<vide>	Les valeurs cotes constantes. Il s'agit de nombres réels séparés par des « ». Il n'est utilisé que si le mot clef « inits » contient « CONSTANTE »
analyse	Un code de scénario POM	Code du scénario POM d'analyse à utiliser pour initialiser ce calcul. Il doit s'agir d'un code d'un des scénarios POM présent dans le fichier « parameters.xml » et qui donne lieu à un run d'analyse. Uniquement valable pour les scénarios POM dédiés à un run de prévision sans analyse.

Tableau 9: mode de calcul du PIM

6.2.3 Fichier « .ini »

Le fichier « .ini » contient les clefs du PI (cf. 3.2.1) et les clefs présentées ci-après.

Le fichier « ini » peut faire l'objet d'une surcharge comme indiqué dans le PI (cf. 3.2.1).

SECTION	CLEF	OBLIG ATOIRE	FORMAT	DESCRIPTION
general	timeout	Oui		le timeout d'exécution de Telemac en secondes
	rootdirectory	Oui		INUTILISE par le PIT, mais il faut définir une valeur : un chemin absolu existant
	commandline	Oui		La commande de lancement de Telemac. Exemple : /home/mascaret/telemac-mascaret/v8p3r1/scripts/python3/telemac2d.py Le PIT va y ajouter automatiquement le nom du fichier cas.
	commandlineworkdirectory	Oui		INUTILISE par le PIT mais il faut définir une valeur
pit	telemacnodes	Oui	Chaîne	La liste des correspondances entre les codes station et les numéros de nœuds Télémac. Il s'agit d'une liste de couples « code nœud;code site hydro » séparés par des « »
	qcomputenodes	Non	Chaîne	Les codes des nœuds de calcul Télémac constituant les sections de contrôles pour lesquelles il faut

				calculer un débit, avec le code site hydro POM attendu. Il s'agit d'une liste de couples « Code nœud debut-Code nœud fin ;code site hydro » séparés par des « »
	conditionslimit	Non	Chaîne	Le nom du fichier des conditions limites de Télémac, sans le chemin associé car il est défini dans le répertoire du modèle (facultatif, vaut par défaut : 'conditions_limites.liq')
	typeconditionslimit	Non	Chaîne	Le nom du fichier du type des conditions limites, sans le chemin associé car il est défini dans le répertoire du modèle (facultatif, vaut par défaut : 'type_conditions_limites.cli')
	backuptimesteps	Non	Chaîne	Le moment de génération des fichiers de reprise (obligatoire, « 0 » par défaut). Il s'agit d'entiers (positifs ou négatifs) séparés par des « ». Le moment de génération des fichiers de reprise est {TEMPS_BASE} + {PAS_DE_TEMPS}. Cela signifie que les pas de temps positifs se situent après le temps de base (ce qui est autorisé mais à utiliser avec précautions). La valeur « 0 » est obligatoire, si elle n'existe pas, elle est ajoutée.
	liqfilecolumns	Oui	chaîne	Les colonnes du fichier « .liq ». Il s'agit d'une série de couples « code entité ; code grandeur » séparés par des « », où le code grandeur vaut H (ou SL pour surface libre) ou Q
	supplmodelfiles	Non	chaîne	Les fichiers supplémentaires du modèle. Il s'agit d'une série de noms de fichiers séparés par des « » (facultatif, vaut par défaut : ""). Ces noms de fichiers sont définis par rapport au dossier du modèle choisi
	commandline_selafin	oui	chaîne	La commande complète 'selafin'

Tableau 10: fichier « .ini » du PIT

Attention de bien formater ces .ini avec un encodage UTF-8 (sans BOM).

6.2.4 Calculs concurrents

Le PIT peut s'exécuter plusieurs fois simultanément et est paramétré en ce sens :

lancementssimultanes = true

Les fichiers de verrou « analyse.pid » et « prevision.pid » sont créés dans le répertoire courant de travail du PIT afin qu'aucun autre lancement du PIT ne puisse venir le concurrencer.

Vue l'arborescence de travail du PIT (cf §6.2.1), ces fichiers verrous permettent d'interdire des lancements simultanés dans un contexte identique très précis, même date pivot, même scénario, même mode de calcul, ...

6.2.5 Ménage automatique

Le PIT peut être configuré pour faire un ménage automatique des fichiers avant et/ou après chaque exécution.

Pour cela, il faut utiliser les réglages 'cleaners' du PI (cf §3.2.1)

7. PIM

7.1 Installation

Ce chapitre traite de l'installation du PIM dans ses versions postérieures ou égales à 4.0 et de Mascaret version v8p2r0 sur la VM Debian 10.

7.1.1 Système d'exploitation

Le fonctionnement du PIM / Mascaret est testé et validé avec la distribution Linux Debian 10. Une VM dédiée est mise à disposition, elle contient tous les pré-requis nécessaires au fonctionnement de PIM3 / Mascaret, ainsi que Mascaret dans sa version v8p2r0.

7.1.1.1 Caractéristiques de la VM

- 4 VCPU
- 3 Go de mémoire
- 16 Go de disque dur

Sur la VM Debian 10 livrée, l'espace disque disponible sur / est de 4.6Go.

Il est très fortement conseillé :

- ➔ soit de configurer les espaces de travail «/home/mascaret/pom/echanges» et «/home/mascaret/pom/pim/workspace» sur un disque externe partagé,
- ➔ soit de configurer des « cleaners » (via les fichiers « .ini ») en cohérence avec la fréquence et le nombre de calculs réalisés et côté POM, de configurer la plateforme pour ne pas conserver les fichiers d'échange après le calcul.

La première option est celle que nous préconisons.

7.1.1.2 Utilisateurs et groupes

Un groupe et utilisateur sont disponibles :

- le groupe « mascaret » contient l'utilisateur « mascaret » pour l'exécution du PIM et de Mascaret

7.1.1.3 Contenu de la VM

Les dossiers utilisés par le PIM3 sont :

- /home/mascaret/pom/
 - /home/mascaret/pom/modèles : contient les modèles installés
 - /home/mascaret/pom/echanges : répertoire utilisé par la POM pour fournir les données nécessaires à chaque calcul à effectuer par le PIM
 - /home/mascaret/pom/pim/workspace : le répertoire de travail du PIM
 - /home/mascaret/pom/pim/workspace/archives : le répertoire de fichiers relais utilisés par le PIM
 - /home/mascaret/uploads : le répertoire de dépôt des archives à utiliser pour installer le PI, PIM

- /home/mascaret/pom/pim/install/virtualenvs : le répertoire des environnements virtuels Python, où sera réellement installé le PIM
- /home/mascaret/pom/pim/install/devel : un répertoire spécial pour les environnements virtuels Python

7.1.2 Pré-requis

Le PIM 4 nécessite :

- la création d'un environnement virtuel (cf 2.3 - Installation des utilitaires Python).

Ces pré-requis sont à réaliser avec le compte **mascaret**.

Les autres pré-requis sont déjà effectués sur la VM livrée. C'est pourquoi nous conseillons fortement d'utiliser cette VM.

7.1.3 Procédure d'installation

- ✓ Télécharger l'archive d'installation

- depuis le site collaboratif du projet POM (<https://travail-collaboratif.din.developpement-durable.gouv.fr/share/page/site/dgprsrnhshapipom/dashboard>), dans l'espace documentaire ==> Interfaçage avec les modèles ==> Plateformes nationales
- déposer les fichiers **pominterface-4.0.XX.whl** et **pim-4.0.XX.whl** dans le répertoire :

```
/home/mascaret/uploads
```

- ✓ Se connecter sur la VM avec le compte **mascaret** et activer l'environnement virtuel

Nous supposons dans la suite que le nom de l'environnement virtuel est « venv-pim-py3 »

```
$ ssh mascaret@xx.xx.xx.xx
$ workon venv-pim-py3
(venv-pim-py3) $
```

- ✓ Installer le PIM

```
(venv-pim-py3) $ cd ~/uploads
(venv-pim-py3) $ pip install pominterface-4.0.XX.whl [proxy]
(venv-pim-py3) $ pip install pim-4.0.XX.whl [proxy]
```

Le paramètre [proxy] est à définir en fonction de la configuration du réseau local. Ce paramètre peut être vide ou de la forme :

```
(venv-pim-py3) $ pip install pim-4.0.XX.whl
-proxy=http://monproxy:ProxyPort
```

Exemple avec le proxy du ministère :

```
(venv-pim-py3) $ pip install pim-4.0.XX.whl -proxy=http://pfrie-std.proxy.e2.rie.gouv.fr:8080
```

Ne pas tenir compte des messages d'avertissement suivants :

WARNING: You are using pip version 20.1.1; however, version 23.1.2 is available.

- ✓ Contrôler les packages installés dans l'environnement virtuel

```
(venv-pim-py3) $ pip list
Package          Version
-----
lxml              4.9.2
numpy            1.21.6
pandas           1.3.5
pim             4.0.0rc0
pip              20.1.1
pominterface    4.0.0rc0
python-dateutil  2.8.2
pytz             2023.3
setuptools       57.0.0
shapely          2.0.1
six              1.16.0
tomli            2.0.1
wheel            0.36.2
```

- ✓ Vérifier l'installation du PIM et des outils du PI

```
(venv-pim-py3) $ pim --version
PIM 4.0.0RC0, PI 4.0.0RC0

(venv-pim-py3) $ checkini --help
INFO:pominterface.tools.checkini:Outil de vérification des .ini
(version PI : 4.0.0RC0)
usage: checkini [-h] ini_file [ini_file ...]

checkini executable

positional arguments:
  ini_file      Provide the ini file path to be checked

optional arguments:
  -h, --help    show this help message and exit
```

7.1.4 Configuration à faire sur la POM

La **plateforme** côté POM doit être configurée ainsi :

- ✓ Exécutable à lancer :

```
. /home/mascaret/.bash_profile &&
/home/mascaret/pom/pim/install/virtualenvs/venv-pim-py3/bin/pim
```

- ✓ Chemin du répertoire des fichiers d'échange :

- | | |
|--|-----------------------------|
| | /home/mascaret/pom/echanges |
|--|-----------------------------|
- ✓ Paramètres de commande :
- | | | |
|-----|---------------------------|-----|
| \$f | /path/to/the/modelpim.ini | \$e |
|-----|---------------------------|-----|
- ✓ Supporte les lancements groupés de modèles :
- | |
|-----|
| non |
|-----|

Le **serveur de calcul** côté POM doit être configuré ainsi :

- ✓ Identifiant : **mascaret**
- ✓ Mot de passe : **password**
- ✓ Type de serveurs : **Serveur de calcul**
- ✓ Système d'exploitation : **Linux**
- ✓ Port : **22**

7.2 Exploitation

7.2.1 Arborescence des fichiers

Le PIM gère une arborescence de fichiers de travail qui permet d'éviter la concurrence entre deux calculs.

Cette arborescence est organisée comme suit :

```

/home/mascaret/pom/pim
|- workspace (répertoire de travail PIM)
|  |- archives
|     |- {CODE_MODEL_MASCARET}
|        |- {TPS_BASE}__{TYPE_SESSION}__{SCENAR_POM}__{MODE}.pim
|  |- {CODE_SESSION_POM}
|     |- {DATE_PIVOT}
|        |- {MODE_CALCUL_POM}
|           |- {CODE_SCENARIO_POM}
|              |- {CODE_MODELE_MASCARET}
|                 |- {MODE_CALCUL_MASCARET}
|                    |- analyse
|                       |- FichierCas.txt
|                       |- ...
|                    |- {NOM_SERIE_1}
|                       |- FichierCas.txt
|                       |- ...
|                    |- {NOM_SERIE_2}
|                       |- FichierCas.txt
|                       |- ...
|                    ...
|  ...

```

Les conventions suivantes sont prises :

- ✓ Le répertoire {CODE_MODELE_MASCARET} est le nom du répertoire du modèle Mascaret lancé (clef « modelsubdirectory » du fichier « .ini »).
- ✓ Les dates sont au format « AAAAMMDD_HHMM » pour effectuer un tri alphabétique chronologique
- ✓ Le mode de calcul POM est lu dans la balise « session → mode » du fichier « parameters.xml ». Il vaut « REAL_TIME » s'il n'est pas renseigné.
- ✓ Le répertoire {MODE_CALCUL_MASCARET} correspond au mode de calcul Mascaret lu dans le mode de calcul du scénario POM.
- ✓ Les répertoires « analyse » et « {NOM_SERIE_i} » (pour les runs de prévision) correspondent aux différents runs à lancer dans Mascaret. Ils contiennent les fichiers nécessaires au calcul

7.2.2 Mode de calcul

Le mode de calcul du PIM peut être renseigné dans le champ « mode de calcul » de chaque scénario (le principal et les éventuels complémentaires) de la POM.

La syntaxe est la suivante :

```
[series={SERIES}] [inits={INITS}] [init_files={INIT_FILES}]
[analyse={ANALYSE}]
```

Tous les mots clefs sont facultatifs. S'ils ne sont pas renseignés, ils sont pris égaux à leur valeur par défaut.

La liste des mots clefs est la suivante :

MOT CLEF	VALEUR PAR DÉFAUT (VALEURS POSSIBLE)	DESCRIPTION
series	MOY (MIN, MAX, MOY, x%, xx %, 100%)	Séries à propager. Sa valeur est une liste de noms séparés par des « ». Les noms désignent des séries de données qui sont renseignées sur les entrées du fichier « parameters.xml ». Le nom d'une série peut être un pourcentage (un entier suivi de %).
inits	REPRISE (REPRISE, DEFAULT)	Méthodes d'initialisation à lancer consécutivement et dans l'ordre indiqué. Sa valeur est une liste des codes de méthodes séparés par des « ». Chaque code ne peut apparaître qu'une fois
init_files	<vide>	Noms des fichiers du répertoire « backupdirectory » pour le mode d'initialisation « DEFAULT », séparés par des « ».

		Il n'est utilisé que si le mot clef « inits » contient « DEFAULT ».
analyse	Un code de scénario POM	Code du scénario POM d'analyse à utiliser pour initialiser ce calcul. Il doit s'agir d'un code d'un des scénarios POM présent dans le fichier « parameters.xml » et qui donne lieu à un run d'analyse. Uniquement valable pour les scénarios POM dédiés à un run de prévision sans analyse.

Tableau 11: mode de calcul du PIM

7.2.3 Fichier « .ini »

Le fichier « .ini » contient les clefs du PI (cf. 3.2.1) et les clefs présentées ci-après.

Le fichier « ini » peut faire l'objet d'une surcharge comme indiqué dans le PI (cf. 3.2.1).

SECTION	CLEF	OBLIG ATOIRE	FORMAT	DESCRIPTION
general	timeout	Oui		le timeout d'exécution de Mascaret en secondes
	rootdirectory	Oui		INUTILISE par le PIM, mais il faut définir une valeur : un chemin absolu existant
	commandline	Oui		INUTILISE par le PIM, mais il faut définir une valeur
	commandlineworkdirectory	Oui		INUTILISE par le PIM mais il faut définir une valeur
	stationsloi	Oui	Chaîne	liste des codes sites (8 caractères) et/ou stations (10 caractères) hydro pour lesquelles le PIM doit fournir un résultat, séparés par des « ». Chaque élément de la liste est un n-uplet dont les champs sont séparés par des « ; » : code site ou station ; code section ; décalage vertical en m. Le numéro de section est utilisé pour lire les fichiers « opt ».
	backuptimesteps	Non	Chaîne	une liste de valeurs entières négatives, pour la génération de fichiers de reprise. Les valeurs sont séparées par des « ». L'unité est la minute. Les valeurs doivent être triées de manière ascendante et sans doublons. Si au moins une valeur n'est pas triée,

				<p>une erreur est générée.</p> <p>Si au moins une valeur est un doublon, une erreur est générée.</p> <p>« 0 » est la valeur par défaut. Si « 0 » n'est pas dans la liste, la valeur est automatiquement ajoutée.</p> <p>Le moment de génération des fichiers de reprise est {TEMPS_BASE} + {valeur}.</p> <p>Exemple : backuptimesteps=-600 -120 0</p>
	stationsresultat	Oui	Chaîne	<p>liste des codes sites (8 caractères) et/ou stations (10 caractères) hydro pour lesquelles le PIM doit fournir un résultat, séparés par des « ». Chaque élément de la liste est un n-uplet dont les champs sont séparés par des « ; » : code site ou station ; code section ; décalage vertical en m. Le numéro de section est utilisé pour lire les fichiers « opt ».</p>
	stationsresultatcasiers	Non	Chaîne	<p>liste des codes sites (8 caractères) et/ou stations (10 caractères) hydro pour lesquelles le PIM doit fournir un résultat, séparés par des « ». Chaque élément de la liste est un n-uplet dont les champs sont séparés par des « ; » : code site ou station ; code section. Le numéro de section est utilisé pour lire les fichiers « cas_opt ».</p>
	stationsresultatliaisons	Non	Chaîne	<p>liste des codes sites (8 caractères) et/ou stations (10 caractères) hydro pour lesquelles le PIM doit fournir un résultat, séparés par des « ».</p> <p>Chaque élément de la liste est un n-uplet dont les champs sont séparés par des « ; » : code station ; code section. Le numéro de section est utilisé pour lire les fichiers « liai_opt ».</p>
	cheminmascaretbarrage	Non	chaîne	<p>chemin complet de l'exécutable Mascaret, spécial pour les calculs de barrages</p>
[init-rampe]	rampe-sections	Non	chaîne	<p>Liste des noms des clefs définissant les 'rampe-section', séparés par des « , ».</p> <p>Chaque nom doit correspondre à une section définie dans le fichier « .ini ».</p>

Tableau 12: fichier « .ini » du PIM

Chaque répertoire à nettoyer fait l'objet d'une section du fichier « .ini », dont le nom doit être indiqué dans la clef « cleaning → cleaners ». Chaque section de « nettoyage » peut prendre les clefs suivantes :

SECTION	CLEF	OBLIGATOIRE	FORMAT	DESCRIPTION
[XXXX] (où XXXX est le nom de la section renseignée dans la clef rampe-sections)	file	Oui	Chaîne	le nom du fichier ligne d'eau à utiliser qui doit être dans le répertoire « backupdirectory » pour le mode « DEFAULT »
	time	Oui	Entier	le temps d'initialisation avant T0 de début de calcul, l'unité est la seconde. La valeur est toujours ≥ 0 . Une valeur positive indique un temps orienté vers le passé à partir de T0. La valeur « 0 » indique que cette « rampe-sections » est inactive. Une valeur négative génère une erreur de lecture.
	stations	Oui	Chaînes	une liste de tuples séparés par des « », chaque tuple est un code de site ou station et une valeur, séparés par « ; », la valeur est en m ³ /s pour des débits ou en m pour des hauteurs

Tableau 13: fichier « .ini » du PIM - « Rampe-sections »

Attention de bien formater ces .ini avec un encodage UTF-8 (sans BOM).

7.2.4 Calculs concurrents

Le PIM peut s'exécuter plusieurs fois simultanément et est paramétré en ce sens, « lancementsimultanes = true ».

Les fichiers de verrou « analyse.pid » et « prevision.pid » sont créés dans le répertoire courant de travail du PIM afin qu'aucun autre lancement du PIM ne puisse venir le concurrencer.

Vue l'arborescence de travail du PIM (cf §7.2.1), ces fichiers verrous permettent d'interdire des lancements simultanés dans un contexte identique très précis, même date pivot, même scénario, même mode de calcul, ...

7.2.5 Ménage automatique

Le PIM peut être configuré pour faire un ménage automatique des fichiers avant et/ou après chaque exécution.

Pour cela, il faut utiliser les réglages 'cleaners' du PI (cf §3.2.1)

7.2.6 Conseils pour le portage des modèles de la VM DAMP

Récupération des données de l'ancienne VM :

→ Les fichiers .ini

- ↳ il faut les copier dans la nouvelle vm
- ↳ puis les mettre à jour
 - vérifier/modifier les chemins
 - Mettre à jour des clés suivantes :
 - 'stationsloi' : changement de contenu
 - 'stationsresultats' : changement de contenu ; cela revient à reprendre la liste des balises <stationRes> du fichier « parametresAssim.xcas » : garder les valeurs de l'attribut 'cdStationHydro' et de la balise <sectionStRes>
 - Gérer les nouvelles clés :
 - 'backuptimesteps' (optionnelle)
 - supprimer les clés obsolètes :
 - 'crue', 'commandlinepalm', 'commandlinepalmworkdirectory', 'resultdirectory', 'observationdirectory', 'loidirectory', 'waterlinedirectory', 'assimvariablesfilepath', 'resultfilepalm'

→ Les modèles mascaret :

- ↳ il faut les copier du dossier /home/admin/HYDRO/MAD_KsorQ_V8/MascaretDA/ vers le dossier /home/mascaret/pom/modeles,
- ↳ puis les mettre à jour :
 - supprimer le dossier intermédiaire de crue :

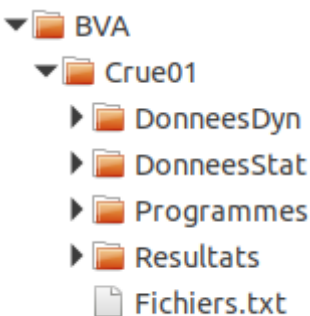
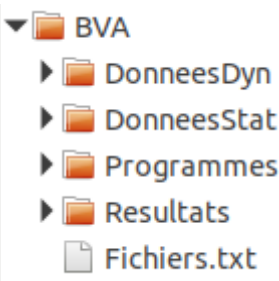
AVANT	APRES
	

Tableau 14 : Exemple de suppression de dossier de crue

- supprimer le dossier « Programmes »
- supprimer le fichier « Fichiers.txt »

- créer un fichier « FichierCas.txt » qui contient le chemin relatif du fichier xcas
- vérifier/modifier les chemins dans « parametresMascaret.xcas »
- supprimer le fichier « parametresAssim.xcas » (car il est devenu obsolète)
- supprimer les fichiers « .obs » (car ils ne sont plus supportés par Mascaret)
- vérifier que chaque fichier « .loi » est bien nécessaire dans le modèle ; par exemple, les fichiers « .loi » dans le dossier « DonneesDyn » sont en général générés par le PIM et ne sont pas nécessaires dans le modèle.

Exemple :

AVANT	APRES
	<p>Contenu du « FichierCas.txt » :</p> <p>DonneesStat/ParametresMascaret.xcas</p>

Tableau 15 : Exemple de mise à jour complète d'un modèle Mascaret

8. PIFG

8.1 Installation

Ce chapitre traite de l'installation du PIFG dans ses versions postérieures ou égales à 4.0.

8.1.1 Système d'exploitation

Ce chapitre suppose que l'installation est faite dans une VM Modèle SCHAPI Debian 11.

8.1.2 Pré-requis

Le PIFG 4 nécessite :

- la création d'un environnement virtuel (cf 2.3 - Installation des utilitaires Python).

8.1.3 Procédure d'installation

- ✓ Télécharger l'archive d'installation
 - depuis le site collaboratif du projet POM (<https://travail-collaboratif.din.developpement-durable.gouv.fr/share/page/site/dgprsrnhshapipom/dashboard>), dans l'espace documentaire ==> Interfaçage avec les modèles ==> Plateformes nationales
 - déposer les fichiers `pominterface-4.0.XX.whl` et `pifg-4.0.XX.whl` dans le répertoire :

```
/home/<user>/uploads
```

- ✓ Se connecter sur la VM avec le compte **mascaret** et activer l'environnement virtuel

Nous supposons dans la suite que le nom de l'environnement virtuel est « venv-pifg-py3 »

```
$ ssh <user>@xx.xx.xx.xx
$ workon venv-pifg-py3
(venv-pifg-py3) $
```

- ✓ Installer le PIFG

```
(venv-pifg-py3) $ cd ~/uploads
(venv-pifg-py3) $ pip install pominterface-4.0.XX.whl [proxy]
(venv-pifg-py3) $ pip install pifg-4.0.XX.whl [proxy]
```

Le paramètre `[proxy]` est à définir en fonction de la configuration du réseau local. Ce paramètre peut être vide ou de la forme :

```
(venv-pifg-py3) $ pip install pifg-4.0.XX.whl
--proxy=http://monproxy:ProxyPort
```

Exemple avec le proxy du ministère :

```
(venv-pifg-py3) $ pip install pifg-4.0.XX.whl
--proxy=http://pfrie-std.proxy.e2.rie.gouv.fr:8080
```

Ne pas tenir compte des messages d'avertissement suivants :

WARNING: You are using pip version 20.1.1; however, version 23.1.2 is available.

- ✓ Contrôler les packages installés dans l'environnement virtuel

```
(venv-pim-py3) $ pip list
Package          Version
-----
lxml              4.9.2
numpy             1.21.6
pandas            1.3.5
pifg            4.0.0rc0
pip              20.1.1
pominterface    4.0.0rc0
python-dateutil  2.8.2
pytz              2023.3
setuptools        57.0.0
shapely           2.0.1
six               1.16.0
tomli             2.0.1
wheel             0.36.2
```

- ✓ Vérifier l'installation du PIFG et des outils du PI

```
(venv-pifg-py3) $ pifg --version
PIFG 4.0.0RC0, PI 4.0.0RC0

(venv-pifg-py3) $ checkini --help
INFO:pominterface.tools.checkini:Outil de vérification des .ini
(version PI : 4.0.0RC0)
usage: checkini [-h] ini_file [ini_file ...]

checkini executable

positional arguments:
  ini_file      Provide the ini file path to be checked

optional arguments:
  -h, --help    show this help message and exit
```

8.1.4 Procédure de recompilation de l'outil de conversion

Le PIFG contient un exécutable **res2bdmp**, ainsi que les sources correspondants.

Si le PIFG renvoie des erreurs lors de l'exécution de **res2bdmp**, il est nécessaire de le recompiler.

Voici les étapes :

- ✓ Installer les pré-requis avec la commande suivante :

- ✓

```
sudo apt install build-essential
```

- ✓ Pour générer l'exécutable, se placer dans le dossier où est installé le PIFG, et exécuter les commandes suivantes :

```
cd <chemin/vers>/pifg/engine
make clean
make
```

Le PIFG est installé dans un sous-dossier de **libs** de l'environnement virtuel. Par exemple :

```
cd /home/schapi/virtualenvs/venv-pifg-py3/lib/python3.9/site-packages/pifg/engine
make clean
make
```

8.1.5 Configuration à faire sur la POM

La **plateforme** côté POM doit être configurée ainsi :

- ✓ Chemin complet de l'exécutable à lancer : **/home/<user>/virtualenvs/XXX/bin/pifg**
- ✓ Chemin du répertoire des fichiers d'échange : **/path/to/the/echanges**
- ✓ Paramètres de commande : **\$f /path/to/the/modelpifg.ini \$e**
- ✓ Supporte les lancements groupés de modèles : **non**

Les paramètres de la commande sont décrit en détail en 4.2.3 - Ligne de commande.

Le **serveur de calcul** côté POM doit être configuré ainsi :

- ✓ Identifiant : **<user>**
- ✓ Mot de passe : **<password>**
- ✓ Type de serveurs : **Serveur de calcul**
- ✓ Système d'exploitation : **Linux**
- ✓ Port : **22**

8.1.6 Ligne de commande

Le lancement en ligne de commande s'effectue comme suit :

```
/home/<user>/.virtualenvs/XXX/bin/pifg {FICHIER_PARAMETERS}
{FICHIER_INI} [--force] [--logs]
```

où :

- ✓ XXX est le nom de l'environnement virtuel du PIFG
- ✓ {FICHIER_PARAMETERS} est le chemin complet du fichier 'parameters.xml' fourni par la POM
- ✓ {FICHIER_INI} est le nom du fichier « .ini » à utiliser pour ce calcul.
- ✓ --force (facultatif) permet de forcer un lancement sans tenir compte des tests de calculs concurrents.
- ✓ --logs (facultatif) permet de forcer la génération des fichiers de log pour debug.

8.2 Exploitation

8.2.1 Arborescence des fichiers

Le PIFG ne crée pas d'arborescence de fichiers de travail. Il effectue les traitements directement dans le dossier créé par la POM et qui contient le **parameters.xml**.

8.2.2 Mode de calcul

Le mode de calcul du PIFG peut être renseigné dans le champ « mode de calcul » de chaque scénario (le principal et les éventuels complémentaires) de la POM.

La syntaxe est la suivante :

```
mode={FLOW|LIMNI}
```

La liste des mots clefs est la suivante :

MOT CLEF	VALEUR PAR DÉFAUT (VALEURS POSSIBLE)	DESCRIPTION
mode	<aucune> (FLOW, LIMNI)	Séries à propager. Sa valeur est une liste de noms séparés par des « ». Les noms désignent des séries de données qui sont renseignées sur les entrées du fichier « parameters.xml ». Le nom d'une série peut être un pourcentage (un entier suivi de %).

Tableau 16: mode de calcul du PIFG

8.2.3 Fichier « .ini »

Le fichier « .ini » contient les clefs du PI (cf. 3.2.1) et les clefs présentées ci-après.

Le fichier « ini » peut faire l'objet d'une surcharge comme indiqué dans le PI (cf. 3.2.1).

SECTION	CLEF	OBLIG ATOIRE	FORMAT	DESCRIPTION
general	timeout	Oui		le timeout d'exécution de Mascaret en secondes
	rootdirectory	Oui		INUTILISE par le PIFG, mais il faut définir une valeur : un chemin absolu existant
	commandline	Oui		INUTILISE par le PIFG, mais il faut définir une valeur
	commandlineworkdirectory	Oui		INUTILISE par le PIFG mais il faut définir une valeur
pifg	enginedirectory	Oui	Chaîne	Chemin absolu du dossier 'engine' ²
	enginecommand	Non	Chaîne	Nom du binaire à exécuter. Habituellement ce nom est « ./res2bdmp »

Tableau 17: fichier « .ini » du PIFG

8.2.4 Calculs concurrents

Le PIFG crée un fichier « .pid » dans le dossier défini par la clé « rootdirectory » du fichier « .ini » ou « .toml ».

Donc 2 PIFG ne peuvent pas être lancés en même temps avec le même « .ini ».

² Il est situé dans le dossier du pifg, situé dans le virtualenv. Exemple :
/home/schapi/virtualenvs/venv-pifg-py3/lib/python3.9/site-packages/pifg/engine