

Interface POM - PLATHYNES

Spécifications Fonctionnelles Détaillées

Réf. POM3-CS-2021-DS-PIPt-008

	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	23/08/17	Création du document
01	01	09/10/17	Prise en compte des remarques
01	02	11/10/17	Mise à jour suite aux réunions de travail
01	03	24/10/17	Gestion des débits
01	04	22/12/17	Gestion des calculs d'ensemble
21	00	20/06/2018	Nouveaux formats BDImage
22	00	11/04/2019	Gestion débits de base/d'apport
22	01	07/05/2019	Prise en compte des remarques suite relecture
22	05	11/10/2019	[#192265] spécificités modèles GSD
22	06	17/10/2019	[#191823] lancement du run d'analyse
22	07	06/11/2019	[#192265] spécificités modèles GSD
22	08	10/06/2020	Répertoire des données dans fichier mrr
03	00	18/08/2020	Développements du PIPT3 sur le PI3
03	01	26/08/2020	Développements du PIPT3 sur le PI3 : compléments pour assimilation / incertitude
03	02	11/09/2020	Prise en compte de remarques
3.1	00	20/10/2020	Développements du PIPT3 sur le PI3 : mise à jour de points en attente
3.1	01	04/12/2020	[#197955] Corrections génération paramètres événementiels ; filtre données pour analyse
3.1	02	18/11/2021	[#202110] Modification de la fonction coût pour l'assimilation
3.1	03	03/12/2021	[#202750] gestion des cas ENS et/ou ASSIMILATION §3.6.3.2 Génération des fichiers « .sim » [#202965] Gestion des Qbvi §3.6.3.1 Génération des fichiers « .evt » [#139] Gestion des fonctions Multi-linéaires pour le temps réel [#143] Gestion du pas de temps de calcul
3.1	04	07/12/2022	[FT #209521] Modification écriture fichier .sim §3.6.3.2 Génération des fichiers « .sim »
3.2	00	02/02/2023	Initialisation par débits observés (§3.6.3.1.4) Propagation des séries Mise à jour des options des dossiers modèles (§1.6.2)
3.2	01	14/02/2023	Prise en compte des retours SCHAPI
4.0	00	11/10/2023	§1.3.3, §1.5.1, §1.7, §1.8, §2.7 : Maj en PIPT 4.0 : migration

			<p>Python3, compatibilité PI 4.0</p> <p>§3.6.3.2.2 : Ajout de ligne de coût dans fichiers « .sim »</p> <p>§3.6.3.1.4 : Concaténation des données débit pour une même station</p> <p>§2.8.1 : Compléments sur Init par débits observés (paramètres événementiels HU ou Q issus de Plathynes Etudes)</p>
4.0	01	05/01/2024	<p>Gestion des nouveaux ouvrages</p> <p>§3.7.2 : Compléments sur les noms des fichiers résultats lus par le PIPT</p> <p>§1.6.3.13, §3.7.2 : Compléments sur les fichiers résultats de barrage lus par le PIPT</p> <p>§3.6.3.17 : Ajout du tri des fichiers mqi et mqo</p>
4.1	00	28/10/2024	<p>[issue #39] Sandre V2</p> <p>§1.5.2 Version POM</p> <p>§1.6.2 Paramétrage du PIPT</p> <p>§3.7.3.2 Construction des données XML Sandre de sortie</p>

Sommaire

Table des matières

1. Généralités.....	13
1.1 Objet du document.....	13
1.2 Glossaire.....	13
1.3 Présentation globale du système.....	13
1.3.1 But.....	13
1.3.2 Contexte du système.....	14
1.3.3 Historique des versions du PIPT.....	14
1.4 Architecture générale.....	15
1.4.1 Architecture matérielle.....	15
1.4.2 Architecture logicielle.....	16
1.5 Version des composants.....	16
1.5.1 Version PI.....	16
1.5.2 Version POM.....	16
1.5.3 Version PLATHYNES.....	16
1.5.4 Version Python.....	17
1.6 Interfaces.....	17
1.6.1 Généralités.....	17
1.6.2 Paramétrage du PIPT.....	17
1.6.3 Formats de fichiers.....	19
1.6.3.1 Fichier Marine watershed configuration « .mwc ».....	19
1.6.3.2 Fichiers événement « .evt ».....	19
1.6.3.3 Fichiers simulation « .sim ».....	20
1.6.3.4 Fichier network « .net ».....	21
1.6.3.4.1 Identification des modèles GSD.....	22
1.6.3.5 Fichier session.xml.....	22
1.6.3.6 Fichier « mqi », « mqo ».....	24
1.6.3.7 Fichier « mhi ».....	25
1.6.3.8 Fichier « mrr ».....	25
1.6.3.8.1 Fichier « mrr » de données non pixelisés.....	25
1.6.3.8.2 Fichier « mrr » de données de pixels.....	26
1.6.3.9 Fichier « grd ».....	26
1.6.3.10 Fichier « asc ».....	27
1.6.3.11 Fichier « relay ».....	27
1.6.3.12 Fichier XXXX_ResultsRaw.txt.....	28
1.6.3.13 Fichier XXX_YYYY_structureResults.txt.....	29
1.7 Arborescence générale.....	29
1.7.1 Arborescence de travail du PIPT.....	30
1.7.2 Export d'un modèle PLATHYNES.....	34
1.7.2.1 Export.....	34
1.7.2.2 Fichier session.xml.....	35
1.7.2.3 Répertoire opérationnel.....	35
1.7.3 Nomenclatures.....	35
1.8 Installation.....	36
1.9 Lancement du solver.....	36

2. Adaptation du PI v4.....	38
2.1 Mode de calcul POM.....	38
2.2 Nettoyage des fichiers.....	38
2.3 Calculs en parallèle.....	39
2.4 Un « run » par scénario.....	40
2.5 Gestion de l'avancement.....	40
2.6 Gestion des unités.....	40
2.7 Dates des runs analyse/prévision et assimilation.....	41
2.8 Méthodes de démarrage de l'analyse.....	41
2.8.1 Préparation du run d'analyse.....	42
2.8.1.1 Étape de détection d'un événement de crue.....	44
2.8.2 Débit de base.....	46
2.9 Méthodes de démarrage de la prévision.....	46
2.9.1 Préparation du run de prévision.....	47
2.9.2 Gestion des modèles GSD.....	48
2.9.3 Gestion des modèles distribués.....	48
2.10 Archive de reprise – fichier « relay ».....	49
3. Périmètre fonctionnel du PIpt.....	50
3.1 Lancement.....	50
3.2 Ordonnanceur de tâches.....	50
3.3 Initialisation.....	50
3.4 Traitement des scénarios (ScenarioPIptProcessor).....	51
3.4.1 Vérifications.....	51
3.4.2 Gestion multi-modèles.....	51
3.5 Traitement des entrées.....	52
3.5.1 Métadonnées Image (MDBDIMAGE).....	52
3.5.1.1 Grandeur HU (MDBDIMAGE).....	52
3.5.1.2 Grandeur RR et série « pixels » (MDBDIMAGE).....	52
3.5.1.3 Grandeur RR et série sans « pixels » (MDBDIMAGE).....	52
3.5.1.4 Autres possibilités.....	52
3.5.2 Métadonnées d'observation PHyC (MDOBSBDH).....	53
3.5.2.1 Grandeur Q (MDOBSBDH).....	53
3.5.2.2 Grandeur RR (MDOBSBDH).....	53
3.5.2.3 Grandeur H (MDOBSBDH).....	53
3.5.2.4 Autres grandeurs.....	53
3.5.3 Métadonnées BP (MDBDLAMEDOBP).....	53
3.5.4 Métadonnées fichier (MDFICHER).....	53
3.5.5 Métadonnées de prévision interne (MDPREVINT) et externe (MDPREVEXT).....	53
3.5.6 Gestion des codes d'entités.....	53
3.5.7 Pluviométrie ponctuelle.....	54
3.6 Run.....	54
3.6.1 Types de run.....	54
3.6.2 Copie des fichiers du modèle.....	55
3.6.3 Paramétrage de la plateforme.....	55
3.6.3.1 Génération des fichiers « .evt ».....	55
3.6.3.1.1 Grandeur HU.....	61
3.6.3.1.2 Grandeur RR et série « pixels ».....	61
3.6.3.1.3 Grandeur RR et série sans « pixels ».....	62
3.6.3.1.4 Grandeur Q.....	63
3.6.3.1.4.1 Si l'entité est de type « Q ».....	63

3.6.3.1.4.2 Si l'entité est de type « Qin » ou « Qbvi ».....	64
3.6.3.1.4.3 Si l'entité est liée à un paramètre événementiel « Q ».....	65
3.6.3.1.5 Grandeur H.....	65
3.6.3.1.6 Métadonnées fichier (MDFICHER).....	66
3.6.3.1.7 Finalisation pour les fichiers « mqi » et « mqo ».....	67
3.6.3.2 Contrôle des structures.....	67
3.6.3.3 Génération des fichiers « .sim ».....	67
3.6.3.3.1 Définition de la section « Ensemble ».....	68
3.6.3.3.2 Définition de la section Assimilation.....	69
3.6.3.4 Modification des fichiers « .mwc ».....	69
3.6.4 Lancement d'un (des) run(s).....	71
3.7 Traitement des sorties.....	72
3.7.1 Sorties de type « fichier ».....	72
3.7.2 Sorties XML Sandre.....	72
3.7.2.1 Lecture des fichiers résultats.....	72
3.7.2.2 Lecture des fichiers résultats des barrages.....	73
3.7.2.3 Construction des données XML Sandre de sorties.....	74
3.8 Finalisation.....	74

Liste des tableaux

Tableau 1 : Glossaire.....	13
Tableau 2 : Définition de la section « Parameters » du fichier .evt.....	60

Liste des figures

Figure 1 : architecture globale de Plathynes.....	15
Figure 2 : architecture matérielle.....	16
Figure 3 : dates des runs T0, T1 et T2.....	41
Figure 4 : méthode d'initialisation du PIPT : INIT_PIPT.....	42
Figure 5 : détection d'un fichier d'évènement de crue.....	45
Figure 6 : enchaînement run d'analyse/prévision du PI3 pour le PIPT3.....	47

1. Généralités

1.1 Objet du document

Ce document présente la spécification logicielle du programme d'interface entre la plateforme opérationnelle pour la modélisation (POM) et la plateforme de modélisation PLATHYNES. Ce programme est nommé PIpt (Programme d'Interface avec PlaThynes).

La mise au point du PIpt s'appuie sur la librairie générique PI (Programme d'Interface) associée.

1.2 Glossaire

Acronyme	Signification
BDH	Base de Données Hydro.
BDTR	Également appelée BDTR (Base de Données Temps Réel).
PHYC/PHYL	Également appelée PHYC/PHYL (Plateforme Hydro Centrale/Locale) pour inclure ses services web.
CS	Communication System
IHM	Interface Homme Machine
PI	Librairie python PomInterface
pip	package installer for Python
PIpt	Programme d'Interface avec PlaThynes
POM	Plateforme Opérationnelle pour la Modélisation
SCHAPI	Service Central d'Hydrométéorologie et d'Appui à la Prévision des Inondations
SPC	Service de Prévision des Crues

Tableau 1 : Glossaire

1.3 Présentation globale du système

1.3.1 But

La POM est un outil du SCHAPI permettant de lancer des calculs de prévision des crues sur différentes plate-formes de modélisation. Parmi ces plates-formes, certaines sont recommandées nationalement par le SCHAPI : Mascaret, Telemac, GRP, Plathynes.

La POM entre en interaction avec ces plateformes selon un protocole standardisé. Elles doivent donc s'interfacer avec ce protocole pour interagir avec la POM.

Le présent document doit donc définir les méthodes et procédures à implémenter pour piloter PLATHYNES à partir des informations fournies par la POM, d'une manière semblable aux autres plateformes citées ci-dessus.

1.3.2 Contexte du système

Conceptuellement, le serveur POM et le serveur « PLATHYNES » (machine virtuelle le plus souvent) sont des serveurs distants. Le présent logiciel est lancé à distance par la POM à l'aide du protocole SSH.

L'exécutable doit donc être lancé en ligne de commande avec pour paramètre

- ✓ le nom du fichier de paramétrage que la POM fournit (« parameters.xml »)
- ✓ le nom du fichier de paramétrage du PIPt (cf. 1.6.2)

Le chapitre 1.4 détaille la chaîne d'exécutables nécessaires au lancement de PLATHYNES par la POM, via le PIPt (le présent exécutable). Du point de vue de la POM, cette distinction doit être transparente.

1.3.3 Historique des versions du PIPt

A l'origine, la première version du PIPt a été réalisée par le SCHAPI, sans utiliser la librairie générique PI, avec des premières fonctionnalités de gestion des assimilations et incertitudes.

La seconde version consiste en :

- ✓ un portage de ce code sur la librairie PI
- ✓ une extension des fonctionnalités de pilotage de PLATHYNES (modes de calcul, jeu, ...)
- ✓ une extension de compatibilité du PIPt sur la nouvelle libhydro
- ✓ abandon de la gestion des assimilations et incertitudes

La précédente version (2.1) permet de :

- ✓ gérer les nouvelles données et formats de la BDImage 2016

La précédente version (2.2) permet de :

- ✓ gérer les débits de base/d'apport
- ✓ prendre en compte certaines spécificités liées aux modèles GSD, et en particulier aux modèles RL/RLa

La précédente version (3.0) consiste en :

- ✓ un portage sur la librairie PI3
- ✓ l'amélioration des phases d'initialisation afin de gérer en particulier les modèles GSD non RL
- ✓ réintégration de la gestion des assimilations et incertitudes

La présente version (4.0) consiste en :

- ✓ un portage sur la librairie PI4
- ✓ initialisation par débits observés
- ✓ propagation des séries statistiques et des quantiles d'un modèle amont

La présente version (4.1) consiste en :

- ✓ la prise en compte du format XML Sandre V2

1.4 Architecture générale

PLATHYNES est un logiciel structuré en modules comme suit :

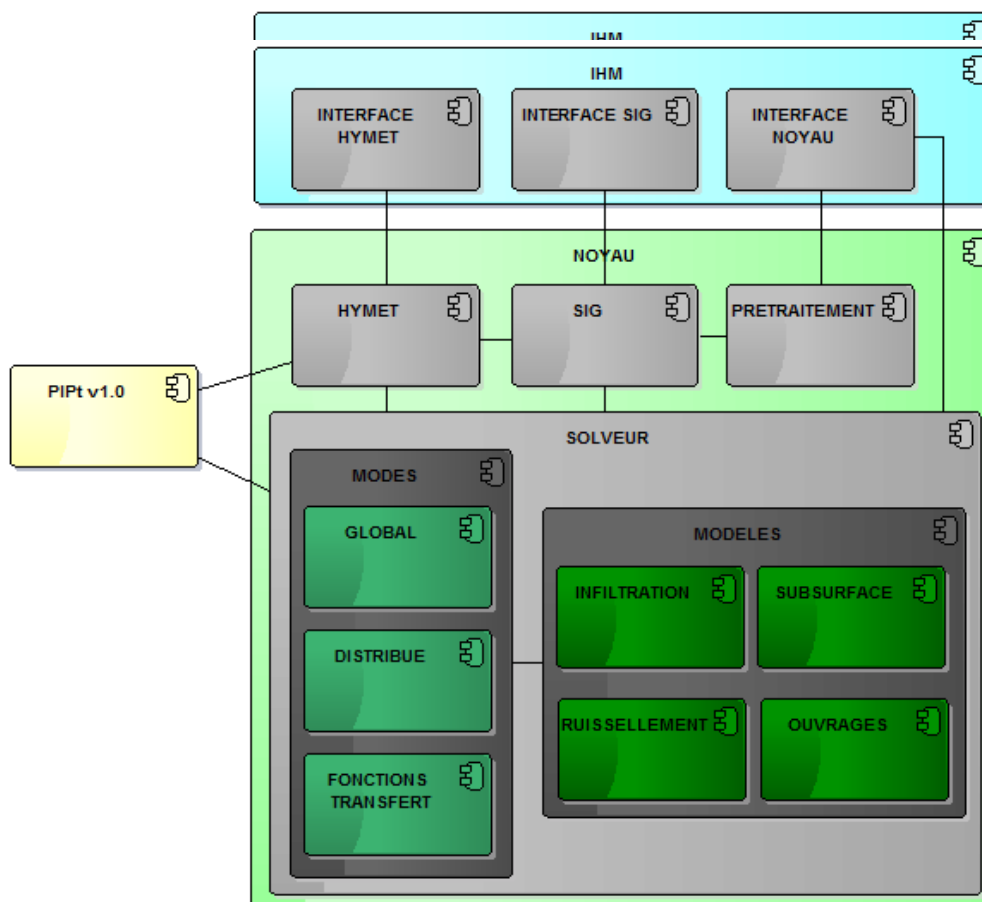


Figure 1 : architecture globale de Plathynes

PLATHYNES fonctionne sous Windows et sous Linux. Il est utilisé en étude, au travers de l'interface graphique, pour mettre au point les modèles de prévisions. L'utilisateur peut ensuite les exporter pour constituer un outil « temps réel » destiné à réaliser uniquement les calculs de prévision.

1.4.1 Architecture matérielle

Le programme d'interface doit être déployé sur le serveur PLATHYNES (qui est une machine virtuelle le plus souvent), accessible en SSH depuis le serveur POM. Il y lance alors le solver PLATHYNES.

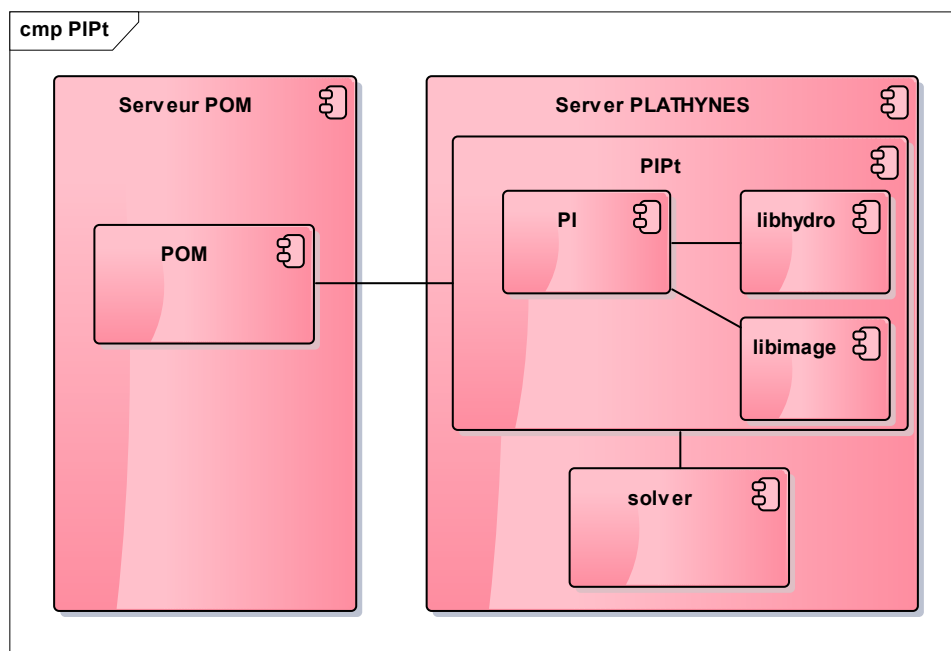


Figure 2 : architecture matérielle

1.4.2 Architecture logicielle

L'architecture logicielle retenue est celle du Programme d'Interface générique PI également nommé librairie PomInterface. Le PIPt consiste en une surcharge de certaines fonctions des processeurs.

1.5 Version des composants

1.5.1 Version PI

Le PIPt utilise la librairie PI dans sa version 4.0 au moins.

la transformation en ascii grid n'est pas intégrée à la libimage. Il est possible de récupérer les lignes de code correspondantes (a priori 8).

1.5.2 Version POM

La présente spécification logicielle est prévue pour s'interfacer avec les versions POM qui implémentent les versions 2.3 et 3.0 du protocole d'échange avec les modèles.

1.5.3 Version PLATHYNES

La présente spécification logicielle est prévue pour s'interfacer avec PLATHYNES dans une version minimum de 1.9.4.

Le PIPt doit vérifier la version de PLATHYNES à piloter. Elle doit correspondre à la version renseignée dans le fichier de paramétrage (cf. 1.6.2).

1.5.4 Version Python

Les développements sont réalisés en Python 3.7.

1.6 Interfaces

1.6.1 Généralités

L'exécutable doit s'interfacer avec certains fichiers ou composants comme indiqué ci-dessous.

Le PIPt est développé sur la base du Programme d'Interface générique (bibliothèque PomInterface) développé pour réaliser les interfaces entre la POM et les modèles distants.

Le PI gère nativement l'interfaçage avec la POM :

- ✓ Arborescence de fichiers
- ✓ parameters.xml
- ✓ progression.xml
- ✓ ...

Le PI utilise la bibliothèque Libhydro pour la lecture / écriture des fichiers au format XML Sandre.

Le PIPt exploite des fichiers aux formats :

- ✓ **XML Sandre (en entrée et sortie)** pour les débits aux sites hydro, pluies aux sites météo (mais pas les hauteurs aux stations hydro),
- ✓ **XML Image (en entrée)** pour les lames d'eau spatialisée ou globale, HU spatialisé ou moyen,
- ✓ **JSON Lamedo (en entrée)** pour les lames d'eau globales rattachées à des sites météo,
- ✓ **ZIP (en entrée)** pour les pluies prévues du modèle Arome.

1.6.2 Paramétrage du PIPt

Le fichier de paramétrage du **PIPt** « .ini » reprend les paramètres du PI qu'il complète des paramètres ci-dessous, spécifiques à Plathynes, dans la section [pipt] :

- ✓ **timeout** : le temps d'exécution maximal de PLATHYNES, en secondes (ex : 600)
- ✓ **lancementsimultanes** : « true » (vrai) pour les plateformes autorisant un lancement en parallèle, « false » (faux) sinon (ici : true)
- ✓ **versionprotocolepom** : numéro de version du protocole POM attendu
- ✓ **rootdirectory** : le chemin complet du répertoire racine PLATHYNES temps réel (ex : /home/plathynes_pom/modeles/plathynes)
- ✓ **commandlineworkdirectory** : répertoire de travail où se placer pour lancer la ligne de commande ci-dessous (ex : /home/plathynes_pom/modeles/plathynes/CODE/)

- ✓ **commandline** : ligne de commande à exécuter pour lancer PLATHYNES, relativement au répertoire « racine » (ex : CODE/bin/solver).

La section « [relay] » doit contenir les valeurs suivantes :

- ✓ **inits** : uniquement la valeur « INIT_PIPT »

Cette section regroupe les paramètres du PIPT (section [pipt]).

- ✓ **versionplathynes** (obligatoire) : version minimum de PLATHYNES pilotable par le PIPT

- ✓ **eventduration** (obligatoire) : durée par défaut en minutes de la plage de calcul de la variation du volume de forçage permettant de détecter un événement et donc nécessitant une reprise sur archive. Elle n'est utilisée que si cette durée n'est pas renseignée dans le fichier session.xml du modèle PLATHYNES.

- ✓ **eventmax** (obligatoire) : valeur maximale par défaut, en mm, de la variation de la lame d'eau de forçage, permettant de détecter un événement et donc nécessitant une reprise sur archive. Elle n'est utilisée que si cette valeur n'est pas renseignée dans le fichier session.xml du modèle PLATHYNES.

- ✓ **grdformat_filename** (facultatif) : masque de vérification des noms de fichiers contenus dans les archives ZIP des métadonnées d'entrée de type fichier. Cette clé est nécessaire s'il y a des données MDFICHER en entrée.

↳ Exemple : `grdformat_filename = ^Pluie_Prevue_(\[d_]*)_RR.asc$`

- ✓ **grdformat_date** (facultatif) : masque de parsing de date des noms de fichiers contenus dans les archives ZIP des métadonnées d'entrée de type fichier. Cette clé est nécessaire s'il y a des données MDFICHER en entrée.

↳ Exemple : `grdformat_date = %Y%m%d_%H%M%S`

- ✓ **incertitudes** (facultatif) : permet de préciser si le PIPT génère des prévisions pour des membres ou des quantiles

↳ valeurs possibles : « membres », « quantiles »

↳ par défaut « membres »

- ✓ **quantiles** (facultatif) : une liste de quantiles qui permet de filtrer les séries de prévisions en sortie du PIPT.

↳ Les valeurs possibles : pourcentages (entiers entre 0 et 100 suivi de %) ou « MIN » ou « MOY » ou « MAX » séparés par des « | »

- par exemple exemple : « 10 %|30 %|50 %|70 %|90 %|MIN|MOY|MAX ».

↳ Une erreur bloquante est générée si une valeur hors bornes (<0 ou >100) est dans la liste

↳ Une erreur bloquante est générée si une valeur non entière (hors « MIN », MAX », « MOY ») est dans la liste

↳ Si la liste est vide, la valeur par défaut est utilisée « 10 %|50 %|90 % ».

1.6.3 Formats de fichiers

1.6.3.1 Fichier Marine watershed configuration « .mwc »

Ce type de fichier est généré par Plathynes Etudes lors de l'action d'export POM.

Exemple de fichier donné à titre informatif. Les clés (« Network file », « Number of stations »...) peuvent être en français ou anglais.

```
#=====
# Network file
#=====
Network file: Anduze500.net

#=====
# Stations
#=====
Number of stations: 4
Station: Anduze 731750.00 1896750.00 569000000.0 Q
Station: Mialet 726250.00 1903750.00 220000000.0 Q
Station: Saumane 718250.00 1901750.00 150000000.0 Q
Station: StJean 723750.00 1901750.00 150000000.0 Q
Baseflow: FIRST_OBS Anduze Anduze 1.0

#=====
# Parameter grids
#=====
Number of parameter grids: 8
Parameter grid: parameters/Anduze500_eta.asc
Parameter grid: parameters/Anduze500_theta.asc
Parameter grid: parameters/Anduze500_kga.asc
Parameter grid: parameters/Anduze500_sf.asc
Parameter grid: parameters/Anduze500_n.asc
Parameter grid: parameters/Anduze500_w1.asc
Parameter grid: parameters/Anduze500_d1.asc
Parameter grid: parameters/Anduze500_i2.asc

#=====
# Model parameters
#=====
Runoff function: 1 KINEMATIC_WAVE_STD G 1.0 5
Runoff function: 2 KINEMATIC_WAVE_TRI2 U 11.68 0 U 9.68 0 G 1.0 6 G 1.0 7 G 1.0 8
Infiltration function: 1 GREEN_AND_AMPT G 10.25 3 G 1.0 4
Subsurface function: 1 DARCY_MOD1 G 4540.0 3 U 0.08 0 G 1.0 8
Soil reservoir function: 1 SIMPLE G 4.62 1 G 1.0 2 E 1.0 1
```

1.6.3.2 Fichiers événement « .evt »

Ce type de fichier est généré par le PIPT et Plathynes Etudes.

Exemple de fichier donné à titre informatif. Les clés (« Nom de l'événement », « Date de debut »...) peuvent être en français ou anglais.

```
#=====
# event settings
#=====
Nom de l'événement: 20020909_MAN
Description:
```

```
#=====
# Temporal settings
#=====
Date de debut: 2002-09-08 06:55:00
Date de fin: 2002-09-09 17:00:00
Pas de temps de forçage: 00:05
Pas de temps de calcul: 00:05
Pas de temps des sorties: 00:05
Pas de temps des bilans: 01:00

#=====
# Parameters
#=====
Nombre de parametres evenementiels: 3
Parametre evenementiel: U 52.8 2 HU2_MOY
Parametre evenementiel: U 52.8 1 HU2_MAX
Parametre evenementiel: U 52.8 3 HU2_MIN

#=====
# Forcing settings
#=====
Nombre de sources de pluies: 1
Source de pluie: Ev_20020909_MAN/20020909_MAN_RRobs.mrr

#=====
# Observations
#=====
Nombre de fichiers d'observations: 1
Fichier d'observation: Ev_20020909_MAN/20020909_MAN_1.mqo
```

1.6.3.3 Fichiers simulation « .sim »

Ce type de fichier est généré par le PIPT et Plathynes Etudes.

Exemple de fichier donné à titre informatif. Les clés (« MWC file », « Number of events »...) peuvent être en français ou anglais.

✓

```
#=====
# MWC file
#=====
MWC file: GoloMarine_Golo_Barchetta_Vcal_tm1.mwc

#=====
# Events
#=====
Number of events: 1
Event file: Ev_20140404t/20140404t.evt

#=====
# Optimisation settings
#=====
Number of attempts: 10
Maximum number of iterations: 30
```

```
Maximum number of simulations: 500
Convergence threshold: 1e-06
#=====
# Variable parameters
#=====
Number of variable items: 5
Variable item: RTF 2 1 5.0 40.0 0.05
Variable item: RTF 2 2 1.0 20.0 0.01
Variable item: SRF 1 1 0.1 2.0 0.002
Variable item: ITF 1 1 0.1 20.0 0.02
Variable item: SSTF 1 1 500.0 5000.0 5.0

#=====
# Assimilation
#=====
Number of cost functions: 1
Cost function: NASH

#=====
# Other criterias (UI specific)
#=====
Number of post-statistics: 5
Post-statistics: CRsim
Post-statistics: CRobs
Post-statistics: volQsim
Post-statistics: QsimMax
Post-statistics: QobsMax
```

1.6.3.4 Fichier network « .net »

Ce type de fichier est généré par Plathynes Etudes.

Exemple de fichier donné à titre informatif.

```
#=====
# MARINE v7.x network
#=====
Type : LP
Grid dimensions : 125 0
Grid extent : 0.00 0.00 0.00 0.00
Grid spacing : 0.00 0.00
Number of cells : 4
Cell type : 1 Bassin_1
Surface unit : 1
Soil unit : 1
Gravity center : -111.00 117.00
Elevation: 0.790230E+02
Area: 0.150000E+09
Mean slope : 0.100000E-03
Mean upstream area : 0.390000E-02
Routing distance : 0.100000E+01
Routing outlet : 4
Number of gauges : 3
1.0000 Saumane
0.0000 Mialet
0.0000 Aval
Number of connections : 1
4 0.100000000000000000000000E+01 0.1000000000000000000000E+02 0.000100
Cell type : 1 Bassin 2
```

```

Surface unit : 2
Soil unit : 2
Gravity center : 97.00 128.00
Elevation: 0.914708E+03
Area: 0.220000E+09
Mean slope : 0.100000E-03
Mean upstream area : 0.217843E+02
Routing distance : 0.100000E+01
Routing outlet : 4
Number of gauges : 3
0.0000 Saumane
1.0000 Mialet
0.0000 Aval
Number of connections : 1
4 0.1000000000000000000000000000000000E+01 0.1000000000000000000000000000000000E+02 0.000100
Cell type : 1 Bassin_3
Surface unit : 3
Soil unit : 3
Gravity center : -40.00 -93.00
Elevation: 0.577457E+04
Area: 0.175000E+09
Mean slope : 0.100000E-03
Mean upstream area : 0.138317E+02
Routing distance : 0.100000E+01
Routing outlet : 4
Number of gauges : 3
0.0000 Saumane
0.0000 Mialet
1.0000 Aval
Number of connections : 1
4 0.1000000000000000000000000000000000E+01 0.1000000000000000000000000000000000E+02 0.000100
Cell type : 3 Station_1
Surface unit : 1
Soil unit : 0
Gravity center : 123.00 -109.00
Elevation: 0.195150E+02
Area: 0.100000E+01
Mean slope : 0.190120E+02
Mean upstream area : 0.000000E+00
Routing distance : 0.100000E+01
Routing outlet : 4
Number of gauges : 0
Number of connections : 1
0 0.1000000000000000000000000000000000E+01 0.1000000000000000000000000000000000E+01 0.000100
Subnetwork : 0

```

1.6.3.4.1 Identification des modèles GSD

Pour déterminer le type du modèle (distribué ou GSD), le PIPT va lire dans l'entête du fichier '.net', la ligne commençant par « Type : » et récupérer la valeur qui suit :

- « GR » : modèle distribué,
- « LP » : modèle GSD.

1.6.3.5 Fichier session.xml

Lors de l'action d'export POM, Plathynes Etudes crée un fichier « session.xml » qui contient des données relatives à l'initialisation des calculs lancés par le PIPT.

La partie « initialisation » du fichier « session.xml » est de la forme

```

<initialisation>
  <wc0 reg="index_reg">index_param_evenementiel</wc0>
  <window>valeur</window>
  <threshold>valeur</threshold>
</initialisation>

```

✓ **wc0 reg**

- ↳ signification : dans le cas où le paramètre choisi est de type « fonction de régression », il s'agit de l'indice de la fonction de régression du fichier « config.mwc »
- ↳ origine : la fonction de régression choisie lors de la création du paramètre événementiel
- ↳ unité : sans, c'est le numéro d'ordre de la fonction de régression dans le fichier mwc

✓ **window**

- ↳ signification : fenêtre de temps (en arrière) dans laquelle on va chercher un run précédent.
- ↳ origine : saisi dans la fenêtre d'export
- ↳ unité : minute

Si cette balise n'est pas présente, le PIPT recherche la valeur dans le pipt.ini.

✓ **threshold**

- ↳ signification : le cumul de pluie à considérer depuis le dernier run pour déterminer si un événement est en cours
- ↳ origine : saisie dans la fenêtre d'export
- ↳ unité : mm

Si cette balise n'est pas présente, le PIPT recherche la valeur dans le pipt.ini.

Le fichier « session.xml » est organisé comme suit :

```

<?xml version="1.0" encoding="UTF-8"?>
<plathynesSession>
  <settings>
    <model>Anduze500_4sta.mwc</model>
    <analyserun>true</analyserun>
    <complet>false</complet>
    <initialisation>
      <wc0 reg="0">1</wc0>
      <window>720</window>
      <threshold>10.000000</threshold>
    </initialisation>
    <parameters>
      <parameter min="0.8" max="1.2">INI,1,2</parameter>
    </parameters>
    <assimilation>
      <window>1440</window>
      <station>
        <code>V7144010</code>
        <threshold>250.0</threshold>
        <weight>1.0</weight>
      </station>
    </assimilation>
    <outputs>

```

```

        <dt>300</dt>
    </outputs>
</settings>
<databases>
    <databaseMeteo>
        <siteMeteo>
            <code>30010003</code>
            <x>0.0</x>
            <y>0.0</y>
            <rscs/>
        </siteMeteo>
    ...
    </databaseMeteo>
    <databaseHydro>
        <siteHydro>
            <name>Anduze</name>
            <code>V7144010</code>
            <x>731750.00</x>
            <y>1896750.00</y>
        </siteHydro>
    ...
    </databaseHydro>
</databases>
<structures>
    <structure name="dam4" station-code="Station_3"
               station-name="Sta_barrage">
        <params>
            <param name="Qdeverse" station-code="Station_2"
                   station-name="Sta_deverse"/>
        </params>
    </structure>
</structures>
</plathynesSession>

```

Les balises « **analyserun** » et « **complet** » sont utilisées par la méthode d'initialisation « INIT_PIPT » (cf 2.8.1 - Préparation du run d'analyse).

La balise « structures » est optionnelle, elle n'est présente que si le modèle a un ou plusieurs ouvrages. Le PIPT va utiliser le contenu de cette balise pour générer des fichiers « .mhi » (cf 1.6.3.7) et lire les fichiers résultats des barrages (cf 3.7.2.2).

1.6.3.6 Fichier « mqi », « mqo »

Ce type de fichier est généré par Plathynes Etudes.

Exemple de fichier donné à titre informatif. Les clés (« MWC file », « Number of events »...) peuvent être en français ou anglais.

```

resourcecode metadatatype grd
entity_0 entity_1 entity_2
3
StationID X Y Type
entity_0 0 0 Qobs
entity_1 1 10 Qobs
entity_2 2 20 Qobs
Date Time Qobs [m3/s]

```

```

2017-10-20 17:23:28 0 1 2
2017-10-20 17:28:28 3 4 5
2017-10-20 17:33:28 6 7 8
2017-10-20 17:38:28 9 10 11
2017-10-20 17:43:28 12 13 14
2017-10-20 17:48:28 15 16 17

```

1.6.3.7 Fichier « mhi »

Ce type de fichier est généré par Plathynes Etudes.

Exemple de fichier donné à titre informatif.

```

MHI file for station entity, series 1

1
StationID X Y Type
entity 0 0 Hobs
Date Time Hobs [m]
2017-10-20 17:23:28 0.0
2017-10-20 17:28:28 0.1
2017-10-20 17:33:28 0.1
2017-10-20 17:38:28 0.15
2017-10-20 17:43:28 0.16
2017-10-20 17:48:28 0.20

```

1.6.3.8 Fichier « mrr »

Ce type de fichier est généré par Plathynes Etudes.

Les exemples de fichiers suivants sont donnés à titre informatif.

1.6.3.8.1 Fichier « mrr » de données non pixelisés

```

#=====
# resourcecode metadatatype grd
#=====
Type de donnees : PLUVIO
Station : metadatacode
Pas de temps : 0 00:10:00
Facteur multiplicatif : 1
3
726250.00 1903750.00
723750.00 1901750.00
718250.00 1901750.00
2017-10-20 17:23:28 0.0 1.0 2.0
2017-10-20 17:28:28 3.0 4.0 5.0
2017-10-20 17:33:28 6.0 7.0 8.0
2017-10-20 17:38:28 9.0 10.0 11.0
2017-10-20 17:43:28 12.0 13.0 14.0
2017-10-20 17:48:28 15.0 16.0 17.0

```

Note : ces fichiers « mrr » sont équivalents aux fichiers « mgr » gérés par Plathynes Etudes.

1.6.3.8.2 Fichier « mrr » de données de pixels

```
#=====
# resourcecode metadatatype grd
#=====
Type de donnees : CSV_RR3
Station : entity_0
Pas de temps : 0 00:10:00
Facteur multiplicatif : 1
Repertoire des donnees : data
201710201723.csv
201710201728.csv
201710201733.csv
201710201738.csv
201710201743.csv
201710201748.csv
```

Fichier ./data/201710201723.csv :

```
X Y VAL
10 1 10.0
10 2 10.0
10 3 10.0
10 4 10.0
11 1 10.0
11 2 10.0
11 3 10.0
11 4 10.0
12 1 10.0
12 2 10.0
12 3 10.0
12 4 10.0
...
```

1.6.3.9 Fichier « grd »

Ce type de fichier est généré par Plathynes Etudes.

Exemple de fichier donné à titre informatif.

```
NCOLS 22
NROWS 20
XLLCORNER 917000
YLLCORNER 1815000
CELLSIZE 1000
NODATA_VALUE -1
```

```

26 24 26 26 27 26 26 25 25 27 27 28 29 29 30 32 32 33 34 34 34 32
27 26 27 26 26 24 24 24 26 26 28 28 28 30 30 30 33 33 34 34 33 32
25 26 25 25 23 23 23 24 25 26 26 28 29 29 30 31 31 34 34 33 33 31
22 25 24 23 21 21 23 24 26 25 26 26 27 28 29 28 30 31 32 32 30 29
21 22 22 21 21 22 23 24 25 25 24 25 25 26 26 27 27 27 29 28 26 25
20 20 20 20 22 22 22 23 23 24 24 23 23 24 24 24 24 23 25 25 23 23
20 20 20 20 22 21 21 21 22 22 22 23 21 22 23 23 21 21 23 23 22 22
...
```

Ce format permet de stocker des données spatialisées : https://en.wikipedia.org/wiki/Esri_grid.

Le PIPt existant dispose d'une fonction pour le faire (convertPointsHu2SIM). Cette fonction est codée en C++ (CODE\src\hymet\moisture) puis wrappée en Python sous forme de librairie dynamique (DLL windows, .so linux). Le nouveau PIPt doit conserver ce fonctionnement.

1.6.3.10 Fichier « asc »

Ce type de fichier est généré par Plathynes Etudes.

Exemple de fichier donné à titre informatif.

```

ncols      107
nrows      74
xllcorner  1138906.000000
yllcorner  1717195.380000
cellsize   500.000000
NODATA_value -9.00
-9.0 -9.0 -9.0 -9.0 -9.0 -9.0 ...
...
```

1.6.3.11 Fichier « relay »

Un fichier « relay » est un fichier texte qui est généré par le solveur Plathynes lors d'un run d'analyse.

Un fichier « relay » est directement réutilisable par le solveur en cas de reprise (cf. 2.8.1 - Préparation du run d'analyse, cf 2.9.1 - Préparation du run de prévision).

Le PIPt n'écrit pas ce type de fichier, il l'archive pour servir de démarrage à un nouveau calcul Plathynes et le lit seulement pour récupérer le volume d'eau. Pour cela, il va lire la première ligne qui suit la ligne « # ! VOLUMES ».

Exemple de fichier donné à titre informatif. :

```

#! RELAY FILE, VERSION: 7.6
#! VOLUMES
480559.79166666599
70409.501807345761
1860639.7329060843
```

```

1718.0202796523706
0.000000000000000000
#! STATE
    2278
    0
    0.000000000000000000 0.000000000000000000
1.9997643873931555E-003 0.000000000000000000
0.000000000000000000
    0.000000000000000000 0.000000000000000000
1.9997348746039882E-003 0.000000000000000000
0.000000000000000000
    0.000000000000000000 0.000000000000000000
1.9997852772817215E-003 0.000000000000000000
0.000000000000000000
    0.000000000000000000 0.000000000000000000
1.9982805995313324E-003 0.000000000000000000
0.000000000000000000
#! UH BASEFLOWS
    0.000000000000000000

```

Autre exemple d'une version plus récente :

```

#! RELAY FILE, VERSION: 7.7
#! EVENT PARAMETERS
    1
    U XX
    70.0814000000000002
#! VOLUMES
    11937.5000000003081
    1397.6400389341634
    10539.859961068918
    0.000000000000000000
    0.000000000000000000
#! STATE
    3779
    1
    0.000000000000000000 0.000000000000000000 -
3.5900872290257370E-004 0.000000000000000000
0.000000000000000000
    9.9606390971791320E-006 0.000000000000000000 -
    0.000000000000000000 0.000000000000000000
3.5900872290257370E-004 0.000000000000000000
0.000000000000000000
    9.9033655964574842E-006

```

1.6.3.12 Fichier XXXX_ResultsRaw.txt

Ce type de fichier est généré par le solveur : il contient les résultats du calcul.

Exemple de fichier donné à titre informatif.

```
Raw results for event 20191222t
```

2								
StationID X Y Type								
Valpajola 1179156.00 1748445.38 Qsim								
EXUTOIRE 1192156.00 1751445.38 Qsim								
Date	Time	Rainfall [mm/h]	QSim [m3/s]	CSim [-]	Rainfall [mm/h]	QSim [m3/s]	CSim [-]	
2019-12-21	19:00:00	0.00000	176.436	0.000000	0.00000	0.000	0.000000	
2019-12-21	20:00:00	0.01174	177.059	0.000000	0.01120	0.000	0.000000	
2019-12-21	21:00:00	0.16131	177.250	0.000000	0.15383	0.000	0.000000	
2019-12-21	22:00:00	1.48889	177.686	0.000000	1.43061	0.000	0.000000	
2019-12-21	23:00:00	4.51182	178.481	0.000000	4.33399	0.000	0.000000	
2019-12-22	00:00:00	6.07681	179.134	0.000000	5.82663	0.000	0.000000	
2019-12-22	01:00:00	11.54867	180.261	0.000000	11.11097	0.000	0.000000	
2019-12-22	02:00:00	3.98700	201.017	0.000000	3.92538	0.040	0.000000	
2019-12-22	03:00:00	0.62020	444.080	0.000000	0.60105	0.401	0.000000	
2019-12-22	04:00:00	0.03735	597.954	0.000000	0.03562	1.315	0.000000	
2019-12-22	05:00:00	0.05390	540.749	0.000000	0.05140	5.272	0.000000	
2019-12-22	06:00:00	0.14917	419.046	0.000000	0.14226	8.440	0.000000	
...								

1.6.3.13 Fichier XXX_YYYY_structureResults.txt

Ce type de fichier est généré par le solveur : il contient des résultats du calcul pour un barrage.

Exemple de fichier donné à titre informatif.

Structure results for event Analyse								
StructureID X Y								
dam5 -83.00 -83.00								
Date	Time	Height [m]	Volume [m3]	VolumeIn [m3]	Derived [m3/s]	Deversed [m3/s]	Turbined	
2019-12-20	00:30:00	245.332	2818061.090	42975.000	0.000	18.980	0.000	
0.000	0.000							
2019-12-20	01:00:00	245.430	2844051.324	43072.200	0.000	20.787	0.000	
0.000	0.000							
2019-12-20	01:30:00	245.470	2854609.403	46348.200	0.000	21.586	0.000	
0.000	0.000							
2019-12-20	02:00:00	245.498	2862076.676	45603.000	0.000	22.152	0.000	
0.000	0.000							
2019-12-20	02:30:00	245.541	2873461.915	50749.200	0.000	23.014	0.000	
0.000	0.000							
2019-12-20	03:00:00	245.569	2880756.360	47943.000	0.000	23.566	0.000	
0.000	0.000							
2019-12-20	03:30:00	245.589	2886032.786	47197.800	0.000	23.965	0.000	
0.000	0.000							
2019-12-20	04:00:00	245.606	2890584.043	47329.200	0.000	24.310	0.000	
0.000	0.000							
2019-12-20	04:30:00	245.639	2899262.650	52126.200	0.000	24.967	0.000	
0.000	0.000							
2019-12-20	05:00:00	245.681	2910324.785	55411.200	0.000	25.804	0.000	
0.000	0.000							
2019-12-20	05:30:00	245.757	2930618.602	65988.000	0.000	27.341	0.000	
0.000	0.000							

1.7 Arborescence générale

L'arborescence de calcul utilisée par le PIPT et le solveur Plathynes est :

```

/home/plathynes_pom/
|- uploads
  |- pominterface-4.0.0.whl
  |- pipt-4.0.0.whl
|- modeles (répertoire destiné aux calculs lancés par la POM)
  |- plathynes
    |- CODE
      |- bin
      |- solver

```

```

|- prepro
|- lib
|- liggis.so
|- liggis_sip.so
|- lighymet.so
|- lighymet_sip.so
|- ligprepro.so
|- ligprepro_sip.so
|- ligsolver.so
|- ligsolver_sip.so
|- src      (les sources de Plathynes Etudes...)
...
|- PIPt
|- pipt_{NOM_MODELE}.ini
|- echanges (répertoire d'échange avec la POM)
|- MODELES
|- {NOM_MODELE}
|- SESSION
|- parameters
|- {NOM_BASSIN}_{NOM_PARAMETRE}*.asc
|- ...
|- {NOM_BASSIN}*.mwc
|- {NOM_BASSIN}*.net
|- ...
|- session.xml
|- WORKDIR (répertoire de travail du PIPt, cf. ci-dessous)

```

Le fonctionnement du PIPt nécessite une organisation des fichiers comprenant :

- ✓ Un répertoire d'échange, alimenté par la POM, indiqué dans le fichier « parameters.xml » fourni par la POM
- ✓ Le fichier du solver PLATHYNES pour exécuter les calculs (fichier indépendant du PIPt) paramétré dans le fichier « ini » du PIPt (cf §1.6.2 - Paramétrage du PIPt), sous la clef « commandline ».
- ✓ Un répertoire contenant l'installation du PIPt (sources)
- ✓ Un répertoire contenant les exports de modèles PLATHYNES
- ✓ Un répertoire de travail du PIPt, paramétré dans le fichier « ini » du PIPt (cf §1.6.2 - Paramétrage du PIPt), sous la clef « [general]workspacedirectory »

1.7.1 Arborescence de travail du PIPt

L'arborescence présentée ci-dessous constitue l'organisation des fichiers manipulés par le PIPt, dénommée ci-après « répertoire racine de travail ». Elle est paramétrée dans le fichier « ini » du PIPt sous la clef « [general]workspacedirectory » (cf. 1.6.2)

L'arborescence des fichiers permet d'éviter un écrasement des jeux de données de chaque calcul dans un souci d'archivage et parallélisation des calculs.

A partir du PIPt 3.2, il y a propagation des séries, comme défini par le PI 3.

Les calculs prévision sont dans une arborescence de travail contenant le nom de la série courante.

Notes :

- ✓ les termes entre accolades sont à remplacer par leur valeur effective au moment du calcul ,
- ✓ les termes entre crochets sont optionnels.

```

/home/plathynes_pom/PIPt
|- WORKDIR (répertoire de travail PIPt)
|   |- Archives
|       |- {CODE_MODEL_PLATHYNES}
|           |-
|               {DATE_PIVOT}__{MODE_CALCUL_POM}__{SCENARIO_POM}__{CODE_SESSION_POM}__{DET|
|               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
|                                               |- *.mhi
|                                               |- *.mrr
|                                               |- Results
|                                           |- {NOM_SERIE_1}
|                                               |- {NOM_BASSIN}.mwc
|                                               |- {NOM_BASSIN}.net
|                                               |- parameters
|                                               |- ...
|                                           |- {NOM_SERIE_1}.pid
|                                           |- {NOM_SERIE_1}.sim
|                                           |- ev_{NOM_SERIE_1}
|                                               |- data
|                                                   |- *.csv
|                                               |- Relay
|                                                   |- Relay_*.txt
|                                               |- Prevision.evt
|                                               |- *.mqi
|                                               |- *.mqo
|                                               |- *.mrr
|                                               |- Results
|                                           |- {NOM_SERIE_2}
|                                               ...
|                   ...
|               ...
|           ...
|       ...
|   ...

```

Les conventions suivantes sont prises :

- ✓ Le répertoire {CODE_MODELE_PLATHYNES} est le nom du répertoire du modèle Plathynes lancer (clef « [models]names » du fichier « .ini »).
- ✓ Les dates sont au format « AAAAMMDD_HHMM » pour effectuer un tri alphabétique chronologique
- ✓ {MODE_CALCUL_POM} : le mode de calcul POM est lu dans le fichier « parameters.xml », la balise « session → mode ». Si cette balise n'est pas présente, la valeur par défaut est « REAL_TIME ».

- ✓ Les répertoires « analyse » et « {NOM_SERIE} » correspondent aux différents runs à lancer dans Plathynes. Ils contiennent les fichiers nécessaires au calcul (cf. 1.9)
- ✓ Les fichiers « analyse.pid » et « prevision.pid » sont des fichiers de lock du PIPT. Ils sont créés en début de calcul et supprimés à la fin. S'il existe déjà un fichier « analyse.pid » ou « prevision.pid » en début de calcul une erreur bloquante est levée. cf. 2.3 Calculs en parallèle
- ✓ Les fichiers « .sim », « .evt », « .mqi », « .mqo », « .mhi », « .mrr », « .csv » sont générés par le PIPT
- ✓ Le fichier « .mwc » est une copie du fichier correspondant pour le modèle dans le répertoire des exports PLATHYNES.
- ✓ Le fichier « .net » est une copie du fichier correspondant pour le modèle dans le répertoire des exports PLATHYNES.
- ✓ Le répertoire « parameters » est une copie du répertoire correspondant pour le modèle dans le répertoire des exports PLATHYNES.
- ✓ Pour un calcul d'analyse, le répertoire « Relay » est généré par le solver et contient les fichiers de reprise.
- ✓ Pour un calcul de prévision, le répertoire « Relay » est créé par le PIPT et contient les fichiers de reprise choisis/copiés par le PIPT.
- ✓ Pour les fichiers relay :
 - ↳ « _DET » : signifie que le fichier relay a été généré par un run déterministe,
 - ↳ « _ENS_nnn » : signifie que le fichier relay a été généré par un run ensembliste, « nnn » est le numéro d'ensemble, affiché sur 3 caractères,
 - ↳ « _ASSIM » : signifie que le fichier relay a été généré par un run avec assimilation.

Exemple sur la VM PLATHYNES temps réel :

```
/home/plathynes_pom/PIPT/WORKDIR
|- Archives
|
| ...
|- SESSION_TR
|   |- 20171110_150000
|   |   |- REAL_TIME
|   |   |   |- 00sPLATEST_S0
|   |   |   |   |- ...
|   |   |   |- 00sPLATEST_S1
|   |   |   |   |- ...
|   |   |- SIMULATEUR
|   |   |   |- 00sPLATEST_S0
|   |   |   |   |- ...
|   |   |   |- 00sPLATEST_S1
|   |   |   |   |- ...
|- 201710111257_REJEU
|   |- 20171110_150000
|   |   |- RECONSTITUTION
|   |   |   |- 00sPLATEST_S0
|   |   |   |   |- ...
|   |   |   |- 00sPLATEST_S1
|   |   |   |   |- ...
|   |   |- SIMULATEUR
|   |   |   |- 00sPLATEST_S0
|   |   |   |   |- ...
```

```
| | | |- 00sPLATEST_S1
| | | |- ...
```

1.7.2 Export d'un modèle PLATHYNES

L'export d'un modèle temps réel PLATHYNES est un répertoire contenant les informations de définition d'un modèle. Il est constitué :

- d'un fichier « session.xml » qui regroupe le paramétrage général
- d'un fichier « *.mwc » (MARINE watershed configuration) qui définit la structure physique du modèle
- d'un fichier « *.net » (network) qui décrit le maillage de calcul
- d'un répertoire optionnel « Doc » contenant un fichier « stations.csv » descriptif de la liste des stations de mesure utilisées dans la modélisation (nom, code, coordonnées géographiques)
- d'un répertoire « parameters » contenant un ensemble de fichiers des valeurs des différentes variables statiques utilisées dans la modélisation.
- d'un répertoire « barrages » contenant l'ensemble des fichiers de définitions des ouvrages utilisés par le modèle

Note : certains de ces fichiers sont facultatifs selon la nature du modèle exporté.

1.7.2.1 Export

Chaque calcul du solveur exploite des informations de configuration des calculs. Parmi ces paramètres figurent ceux du modèle PLATHYNES à lancer. Ces informations sont générées par l'interface PLATHYNES sous forme d'un jeu de fichier comme suit :

```
SESSION
|- {NOM_BASSIN}*.mwc
|- {NOM_BASSIN}*.net
|- session.xml
|- Doc (répertoire)
|   |- stations.csv
|- parameters (répertoire)
|   |- {NOM_BASSIN}*.asc
|   |- ...
|- barrages (répertoire)
|   |- barragel.txt
|   |- barragel_XXXX (répertoire)
|       |- loi_HV.txt
|       |- ...
```

Parmi ces fichiers, seuls les fichiers « session.xml » et « .mwc » font l'objet d'une lecture et d'analyse de leur contenu. Les autres sont uniquement exploités par copie de fichiers.

1.7.2.2 Fichier session.xml

Le fichier « session.xml » est organisé comme suit :

cf 1.6.3.5 Fichier session.xml

1.7.2.3 Répertoire opérationnel

Cet ensemble de fichier doit être copié manuellement dans un répertoire idoine du serveur PLATHYNES, pour constituer le jeu de fichiers de référence des différents modèles à faire tourner en temps réel :

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

Chaque répertoire « SESSION » peut contenir d'autres fichiers non exploités par le PIPT, ce qui n'est pas gênant.

Le nom de ce répertoire {RACINE_EXPORT} est indiqué dans le fichier « ini » du PIPT (cf. 1.6.2) sous la clef « [models]directory ».

Exemple sur la VM PLATHYNES temps réel :

```
/home/plathynes_pom/modeles/plathynes/WORKDIR
|- 00sPLATEST
|  |- SESSION
|  |  |- Anduze500_4sta.mwc
|  |  |- parameters (répertoire)
|  |  |- session.xml
```

1.7.3 Nomenclatures

Par la suite, les dénominations suivantes sont prises :

- ✓ le répertoire ci-dessous est appelé « répertoire du modèle Plathynes »

```
{RACINE_EXPORT}/{NOM_MODELE}
```

- ✓ Le répertoire ci-dessous est appelé « répertoire de travail PIPT » :

```
PIPT_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}
```

- ✓ le répertoire ci-dessous est appelé « répertoire de travail Plathynes »

```
{répertoire de travail PIPT}/analyse/
```

ou

```
{répertoire de travail PIPT}/prevision/
```

1.8 Installation

A partir de la version 3.2, l'OS d'installation est Linux Debian 10.

Pour l'installation, le répertoire suivant permet de déposer les fichiers nécessaires au PIPT :

```
/home/plathynes_pom/PIPt/install
```

Il s'agit des fichiers :

- ✓ d'installation du PI : **pominterface-4.X.X.whl**
- ✓ d'installation du PIM : **pipt-4.0.x.whl**

Le PIPT est prévu pour être installé de manière standard avec l'outil « pip ».

Exemple de commande d'installation :

```
$ pip install pipt-4.0.0.whl
```

1.9 Lancement du solver

Le solver PLATHYNES se lance en ligne de commande.

Pour un run d'analyse la commande est :

```
-root [RACINE_DIR] analyse analyse -mode analysis
```

Pour un run de prévision la commande est :

```
-root [RACINE_DIR] {NOM_SERIE} {NOM_SERIE} -mode forecast
```

Note : les paramètres d'appels ci-dessus sont nécessaires au fonctionnement du solver :

- La première valeur « analyse » ou « {NOM_SERIE} » correspond pour le solver au répertoire de travail pour le run à effectuer
- La seconde valeur « analyse » ou « {NOM_SERIE} » correspond au nom de la simulation pour le run à effectuer, fichier « analyse.sim » ou « {NOM_SERIE}.sim »
- cf 1.7.1 - Arborescence de travail du PIPT

Le solver lit les fichiers suivants :

- ✓ **Le fichier de simulation**
 ↳ [RACINE_DIR]/{analyse | {NOM_SERIE}}/{analyse | {NOM_SERIE}}.sim
- ✓ **Le fichier de configuration**
 ↳ [RACINE_DIR]/{analyse | {NOM_SERIE}}/[BASSIN].mwc
 (sauf si le fichier de configuration est défini dans le fichier de simulation)



Avec L'arborescence de travail du PIPT, [RACINE_DIR] correspond à :

- ✓ PIPT_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}

2. Adaptation du PI v4

L'architecture logicielle est basée sur le PI avec surcharge de processeurs ou fonctions du PI.

Les points ci-dessous précisent les fonctionnalités du PI à réutiliser.

2.1 Mode de calcul POM

L'utilisateur doit renseigner un mode de calcul sur chaque scénario POM. Celui-ci est interprété par le PIpt pour réaliser les calculs.

Le mode de calcul comporte les mots clefs supplémentaires à ceux gérés dans le PI suivants :

```
[mode={MODE}] [assimilation={ASSIMILATION}] [num_ensemble={N}]
```

Note : les crochets indiquent les paramètres facultatifs, les accolades indiquent les variables modifiables.

Le fait de nommer les paramètres permet d'éviter d'avoir à les ordonner. L'utilisateur peut les saisir dans un ordre quelconque.

Les variables avec :

- ✓ {MODE} (facultatif) : s'il n'est pas renseigné, ce paramètre est pris égal à « DET », sinon il doit prendre une valeur parmi
 - ↳ DET : le calcul est lancé en mode « déterministe » sans assimilation
 - ↳ ENS : le calcul est lancé en mode « ensemble » sans assimilation
- ✓ {ASSIMILATION} (facultatif) : s'il n'est pas renseigné, il est pris égal à « NON ». S'il est renseigné il doit prendre une valeur parmi :
 - ↳ OUI : le calcul est lancé avec assimilation
 - ↳ NON : le calcul est lancé sans assimilation
- ✓ {N} (facultatif) : entier strictement positif, compris entre 2 et 999. Ce paramètre est obligatoire si le mode vaut « ENS ».

2.2 Nettoyage des fichiers

L'objectif du nettoyage est :

- ✓ éviter d'écraser des fichiers utilisés dans des runs en cours,
- ✓ éviter de réutiliser par mégarde des fichiers issus d'un run précédent,
- ✓ éviter de saturer l'espace disque de la VM.

Pour un scénario POM, dans l'arborescence prévue (cf. 1.7.1 Arborescence de travail du PIpt), cela correspond aux répertoires suivants :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}
|- analyse
```

```
|- prevision
...
```

Voici une configuration du processeur de nettoyage par défaut.

```
[cleaning]
cleaners=Relay,Relay_TR,Analyse,Analyse_TR,Prevision,Prevision_TR

[Relay]
directory=/home/plathynes_pom/PIPt/WORKDIR/*/*/Relay

filter=*
age=90
mode=postrun

[Relay_TR]
directory=/home/plathynes_pom/PIPt/WORKDIR/SESSION_TR/*/Relay
filter=*
age=7
mode=postrun

[Analyse]
directory=/home/plathynes_pom/PIPt/WORKDIR/*/ANALYSE_*/
filter=*
age=7
mode=postrun

[Analyse_TR]
directory=/home/plathynes_pom/PIPt/WORKDIR/*/ANALYSE_*/
filter=*
age=0
mode=postrun

[Prevision]
directory=/home/plathynes_pom/PIPt/WORKDIR/*/PREVISION_*/
filter=*
age=7
mode=postrun

[Prevision_TR]
Directory=/home/plathynes_pom/PIPt/WORKDIR/SESSION_TR/*/
PREVISION_*/
filter=*
age=0
mode=postrun
```

2.3 Calculs en parallèle

Le PIPt peut s'exécuter plusieurs fois simultanément et est paramétré en ce sens.

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, ces fichiers verrous permettent d'interdire des lancements simultanés dans un contexte identique très précis, cf §1.7.1 - Arborescence de travail du PIPT, même date pivot, même scénario, même mode de calcul, ...

2.4 Un « run » par scénario

Le PI prévoit de fonctionner différemment selon qu'il doit lancer (au moins) un « run » par scénario POM ou un « run » pour tous les scénarios.

Dans le cas présent, chaque scénario POM donne lieu à au moins un « run ».

2.5 Gestion de l'avancement

L'avancement est géré de manière générique comme suit :

- ✓ 5% : lancement de PIPT
- ✓ X% : run
- ✓ 95% : finalisation
- ✓ 100% : fin de PIPT

La phase « run » représente donc 90 % : 45 % pour l'analyse, 45 % pour la prévision.

Chaque scénario est représenté à part égale dans ces 45 %. Chaque scénario occupe donc 45/N % (N = nombre de scénarios) et son avancement se décompose comme suit :

- ✓ 20 % : lecture des entrées
- ✓ 60 % : lancements de l'exécutable Plathynes
- ✓ 20 % : traitement des sorties

2.6 Gestion des unités

Les valeurs des données dans PLATHYNES sont dans les unités suivantes :

- ✓ Pour les débits
 - ↳ m³/s pour les débits
- ✓ Pour les pluies
 - ↳ 1/10ème de mm cumulé sur le pas de temps pour les fichiers ascii grid (asc ou grd)
 - ↳ 1/10ème de mm cumulé sur le pas de temps pour les fichiers CSV
 - ↳ 1/10ème de mm cumulé sur le pas de temps CSV XML Image (CSV RR3 dans PLATHYNES)

Les hauteurs ne sont pas acceptées en entrée, une erreur est levée en ce sens en cas de besoin.

La gestion des conversions d'unités sont spécifiées pour chaque type d'entrée (cf. 3.5).

2.7 Dates des runs analyse/prévision et assimilation

Le PIpt fonctionne en ANALYSE / PREVISION sur l'architecture du PI3.

Les dates T0, T1 et T2 ont calculées par le PI3 (cf §2.4.1, §2.4.2, §2.4.3).

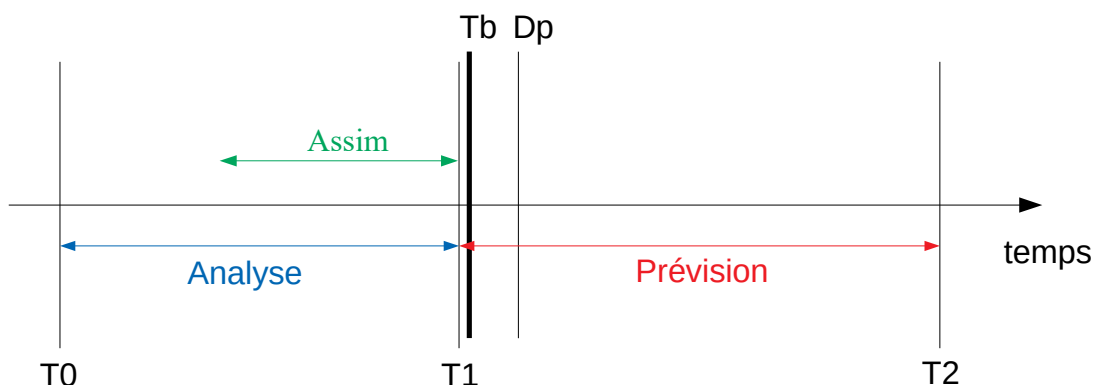


Figure 3 : dates des runs T0, T1 et T2

Le PI calcule T1 avec un arrondi sur le plus petit pas de temps des données observées du scénario courant.

Si l'assimilation de données est activée (cf §2.1 - Mode de calcul POM), la plage de temps pour l'assimilation est fonction de la valeur définie pour la balise « assimilation → window » définie dans le fichier « session.xml » (cette valeur est en minute) (cf §1.6.3.5 - Fichier session.xml) :

- ✓ si la valeur de « window » est $> T1 - T0$, l'assimilation se fera sur $[T0 - T1]$
- ✓ si la valeur de « window » est $< T1 - T0$, l'assimilation se fera sur $[T1 - \text{window}, T1]$

Pour le calcul d'analyse, contrairement au PI, le PIpt va utiliser toutes les données $< T1$.

2.8 Méthodes de démarrage de l'analyse

Pour les runs d'analyse, le PIpt implémente une méthode d'initialisation dédiée INIT_PIPT :

- ✓ cette méthode combine plusieurs tests (sur le contenu du fichier « session.xml », sur la détection d'évènement de crue, sur la présence de données HU pour l'initialisation du modèle)
- ✓ cette méthode permet d'effectuer des calculs avec les modes standards suivants :
 - ↳ REPRISE
 - ↳ INIT_OBS
 - ↳ SANS_ANALYSE.

Les autres méthodes directes (REPRISE, DEFAULT, CONSTANTE, PERMANENT, AUCUNE...) ne sont pas gérées et produisent une erreur bloquante.

2.8.1 Préparation du run d'analyse

Le schéma suivant illustre le fonctionnement de la méthode d'initialisation INIT_PIPT.

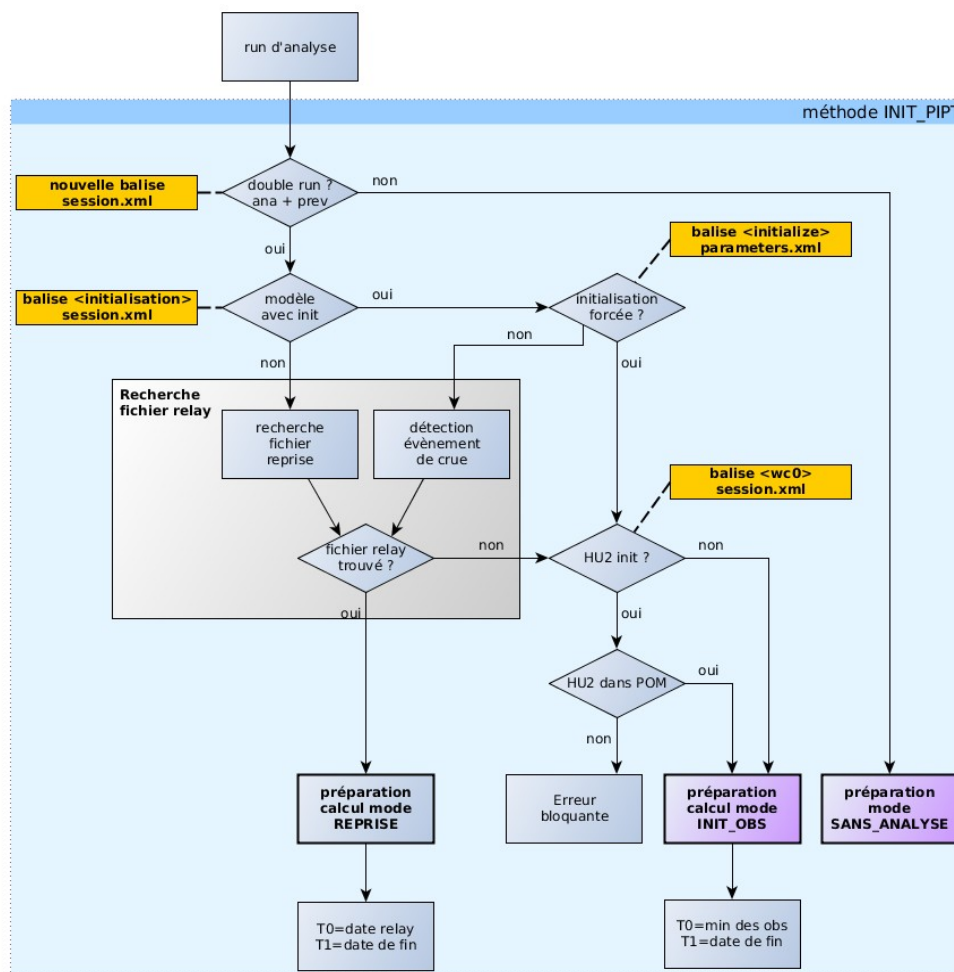


Figure 4 : méthode d'initialisation du PIPT : INIT_PIPT

- ✓ La possibilité d'effectuer un run d'analyse sera défini dans Plathynes Etudes et sera exporté dans le fichier « session.xml » pour être utilisé par le PIPT. La balise « **analyserun** » est définie pour cela. La valeur de cette balise sera appliquée avec la même valeur pour tous les scénarios du modèle. Les valeurs possibles seront « true » et « false » :
 - ✎ **analyserun=true** : le modèle lance un run d'analyse et un run de prévision :
 - ✎ **analyserun=false** : le modèle ne lance pas de run d'analyse, mais seulement un run de prévision

Les conditions d'activation des runs d'analyse et prévision sont définies au 3.6.1 - Types de run

- ✓ En l'absence de données observées, il n'y aura pas de run d'analyse lancé. Cela est géré en standard par le PI.
- ✓ La méthode d'initialisation « **SANS_ANALYSE** » sera celle utilisée par le PIPIt, si pas de run d'analyse.
- ✓ La balise « **initialize** » du fichier « parameters.xml » fourni par la POM permet de gérer les 2 cas suivants :
 - ↳ **initialize=true** : initialisation du modèle est forcée
 - ↳ **initialize=false** : détection d'un évènement de crue
- ✓ La prise en compte de la phase d'initialisation du modèle Plathynes, à savoir la présence et l'utilisation de données HU utilisera la balise « **wc0** » comme c'est le cas actuellement dans le PIPIt2. Si des données HU sont obligatoires et non présentes une erreur bloquante sera levée et les traitements seront interrompus avec un statut d'erreur.

Note : jusqu'à la version v1.9.22, Plathynes Etudes n'exporte que les paramètres événementiels de type HU. A partir de la version v1.9.23RC0, Plathynes Etudes peut exporter les paramètres événementiels de type HU ou Q. Pour cela, il ajoute 2 attributs à la balise <wc0> : 'par_type' et 'par_station' :

- 'par_type' : la valeur est "HU" ou "Q"
- 'par_station' : la valeur est définie si 'par_type'="Q" ; la valeur est le nom d'une station.

↳ **wc0 existe** :

- recherche du type de données événementielle :
 - Le PIPIt lit l'attribut 'par_type' de la balise 'wc0' pour déterminer le type du paramètre événementiel : HU ou Q.
 - Le PIPIt gère la compatibilité ascendante : si l'attribut 'par_type' n'existe pas, le PIPIt force le type à 'HU'.
- recherche de la station associée :
 - Le PIPIt lit l'attribut 'par_station' de la balise 'wc0' pour déterminer le nom de la station associée au paramètre événementiel Q.
- Si le type est 'HU' : recherche de la présence de données HU fournis par la POM.
 - Si pas de données HU fournies par la POM, une erreur bloquante est générée.
 - Si des données HU sont fournies pas la POM, initialisation du modèle avec les données HU, utilisation de la méthode **INIT_OBS**
- Si le type est 'Q' et une station associée est définie : recherche de la présence de données Q pour cette station fournis par la POM.
 - Si pas de données Q pour cette station, une erreur bloquante est levée
 - Si des données Q sont fournies pas la POM, initialisation du modèle avec les données Q, utilisation de la méthode **INIT_OBS**

↳ **wc0 absente** : pas d'initialisation du modèle, utilisation de la méthode **INIT_OBS**

- ✓ Dans le cas de l'utilisation d'un fichier relay, c'est la méthode d'initialisation existante **REPRISE** qui sera utilisée, cf specs PI3, §2.1.2. Il s'agira de l'adapter pour permettre de lui fournir un fichier relay détecté par le PIPT.

Les dates T0, T1 et T2 ont été calculées par le PI3 (cf §2.4.1, §2.4.2, §2.4.3).

2.8.1.1 Étape de détection d'un événement de crue

Ce chapitre décrit l'algorithme permettant de déterminer si l'on est (ou pas) « en événement de crue ». Cette méthode renvoie soit un couple (fichier relay, date) soit VIDE.

Comme indiqué au chapitre §2.10 - Archive de reprise – fichier « relay », la détermination repose sur l'évolution des volumes de forçage au cours des X dernières minutes, X étant un paramètre en minutes (issu du fichier « session.xml » ou à défaut du fichier « ini », clef « eventduration », cf. §1.6.2 - Paramétrage du PIPT).

Il suffit de s'appuyer sur les archives de reprise (et non sur les observations fournies par la POM) car quel que soit le cumul des données observées, l'initialisation se fait toujours avec les données HU s'il n'existe pas d'archive de reprise.

Le seuil de détection d'un événement est également un paramètre du fichier « session.xml » ou à défaut du fichier « ini » (clef « eventmax »). Notons le C. Il est exprimé en mm.

Un avertissement est tracé lorsque la valeur utilisée est celle du fichier « .ini ».

Pour commencer, il faut rechercher le fichier de reprise le plus récent entre D1 et D2 :

- ↳ D1 : Temps de Base POM – min(profondeur des entrées observées)
- ↳ D2 : Temps de Base POM – max(profondeur des sorties)

Le répertoire de recherche est le répertoire des archives du modèle :

PIPT_WORKDIR/Archives/{CODE_MODEL_PLATHYNES}

S'il n'existe pas un tel fichier, on ne peut pas initialiser le solveur à partir d'une reprise, la méthode renvoie VIDE.

Si un tel fichier existe, il est associé à une date D. Il faut le comparer à un fichier de reprise situé vers D-X, c'est-à-dire le plus récent entre :

- ↳ D'1 : D-X-(D2-D1)
- ↳ D'2 : D-X

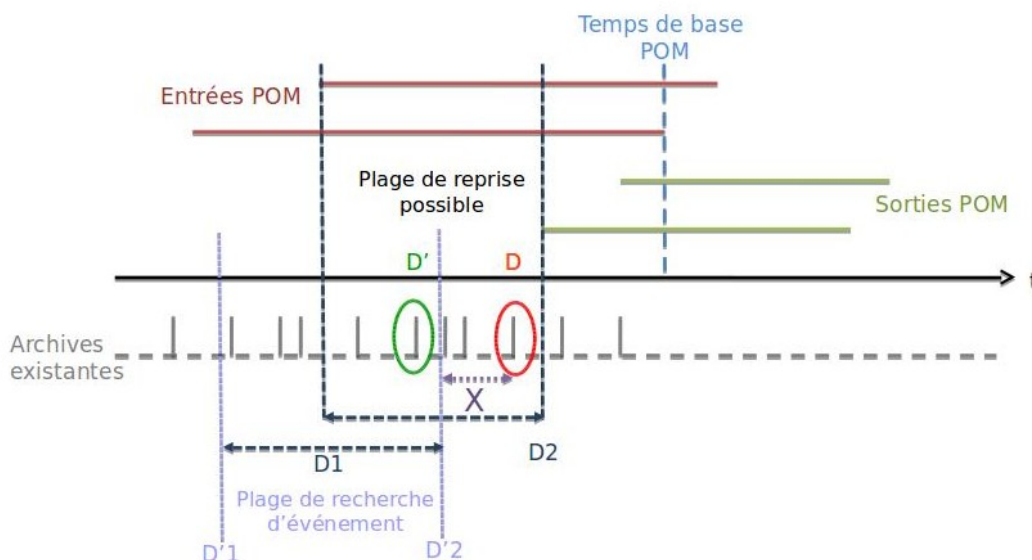


Figure 5 : détection d'un fichier d'évènement de crue

Différents cas se présentent :

- ✓ Aucun fichier n'a été trouvé entre D'1 et D'2 : on considère que le volume de forçage X minutes avant est nul. Le volume de forçage de D représente donc l'évolution du volume de forçage.
- ✓ Si un fichier a été trouvé (disons D'), on calcule l'évolution du volume de forçage entre D' et D, qui est en réalité la lame d'eau écoulée :

$$(\text{VolumeD} - \text{VolumeD}') / \max(\text{Surf des BV})$$

Attention : D'-D peut être supérieur à X mais ça ne pose pas de problème car on considère la situation la plus critique à savoir que le cumul est tombé sur la période la plus récente.

Note : les surfaces des BV sont renseignées dans le fichier « .mwc » pour chaque site hydro.

Note 2 : les volumes sont en m³ et les surfaces en m², puis la lame d'eau est convertie en mm.

En fonction de la valeur de l'évolution du volume de forçage (lame d'eau écoulée), deux cas se produisent :

- ✓ L'évolution est supérieure ou égale à C, on considère qu'un événement débute ou continue et la méthode renvoie le fichier et la date D.
- ✓ Sinon (elle est strictement inférieure à C), il n'y a pas d'événement et la méthode renvoie VIDE.

Note : pour trouver un fichier de reprise entre D1 et D2 :

- ✓ On initialise une date de début {D} à « vide ».
- ✓ On boucle sur les fichiers du répertoire de reprise du modèle :

```
PIPt_WORKDIR/Archives/{CODE_MODEL_PLATHYNES}/
{DATE_PIVOT}_{MODE_CALCUL_POM}_{SCENARIO_POM}_{CODE_SESSION_POM}_
_{DET|ENS_XXX}[__ASSIM].zip
```

✎ Si {DATE_PIVOT} est entre D1 et D2 et si {DATE_PIVOT} > {D} alors on prend cette {DATE_PIVOT} pour nouvelle {D}.

La recherche s'effectue uniquement sur les fichiers qui ont été générés avec

- ✓ le même mode de calcul DET ou ENS,
 - ✓ le même nombre d'ensemble si le mode est ENS,
 - ✓ l'assimilation si elle est demandée pour le run en cours,
 - ✓ le même code de session POM {CODE_SESSION_POM}.
- ✓ A la fin de la boucle sur les fichiers, si {D} est « vide » c'est qu'aucun fichier de reprise n'est utilisable. Sinon, il faut utiliser le fichier correspondant à {D} :

```
PIPt_WORKDIR/Archives/{CODE_MODEL_PLATHYNES}/
{D}_{MODE_CALCUL_POM}_{SCENARIO_POM}_{CODE_SESSION_POM}_{DET|
ENS_XXX}[__ASSIM].zip
```

2.8.2 Débit de base

Pour information, le débit de base est géré uniquement par le solveur comme suit :

- ✓ Si l'on repart d'une archive de reprise (run de prévision ou run d'analyse avec fichier relay), le débit de base est issu de l'archive,
- ✓ Si l'on ne repart pas d'une archive de reprise, le débit de base est calculé en fonction de la méthode indiquée dans le fichier « mwc ».

Le PIPt vérifie alors que les stations listées comme 'baseflow' dans le fichier « .mwc » ont des données de débits observés. Dès qu'une de ces stations n'a pas de données de débit observé, une erreur bloquante est levée.

Note : en temps réel, le débit de base de la dernière initialisation HU est propagé dans les fichiers de reprise. Il suffit donc que le PIPt génère des fichiers à partir des données POM.

2.9 Méthodes de démarrage de la prévision

Pour les runs de prévision, le PIPt implémente une méthode d'initialisation dédiée :

- ✓ cette méthode combine plusieurs tests (sur le contenu du fichier session.xml)
- ✓ cette méthode permet d'effectuer des runs de prévision.

A partir de PIPT3.2, cette méthode est améliorée pour prendre en compte la propagation des séries prévue par le PI :

- ✓ cette méthode est exécutée pour chaque série à calculer
- ✓ le PIPT filtre les données d'entrée en fonction de la série à calculer pour le run de prévision courant.

2.9.1 Préparation du run de prévision

Le schéma suivant illustre le fonctionnement global du PIPT, enchaînement run d'analyse / run de prévision :

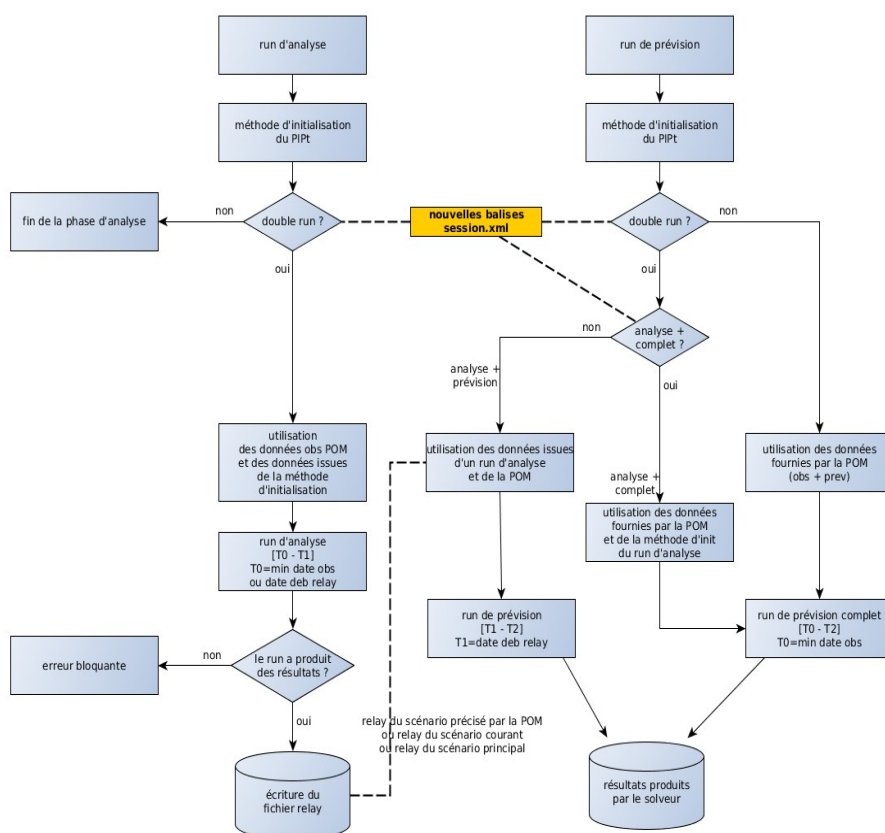


Figure 6 : enchaînement run d'analyse/prévision du PI3 pour le PIPT3

Pour les runs de prévision, deux cas ont implémentés :

- ✓ un run de prévision sur [T1-T2] avec utilisation des données fournies par la POM, les prévisions, et du fichier relay produit à T1 par le run d'analyse

- ✓ un run de prévision sur [T0-T2] :
 - ↳ avec utilisation de toutes les données fournies par la POM, les observations et les prévisions (sans run d'analyse)
 - ↳ avec utilisation de toutes les données fournies par la POM, les observations et les prévisions, ainsi que les paramètres de lancement du calcul d'analyse.

Note : des filtres sont appliqués sur les données d'entrée fournies par la POM, cf 3.5 - Traitement des entrées.

- ✓ La recherche du run d'analyse est effectuée par le PI 3 (cf specs du PI3 : 4.6.3 – Recherche scénario d'analyse). Si aucun run d'analyse ne correspond, une erreur bloquante interrompt le calcul.
- ✓ Les dates T0, T1 et T2 sont calculées par le PI3 (cf §2.4.1, §2.4.2, §2.4.3).
- ✓ La possibilité d'effectuer un run d'analyse sera définie dans Plathynes Etudes et sera exportée dans le fichier « session.xml » pour être utilisé par le PI3. La balise « **analyserun** » est définie pour cela. La valeur de cette balise sera appliquée avec la même valeur pour tous les scénarios du modèle. Les valeurs possibles seront « true » et « false » :
 - ↳ **analyserun=true** : le run de prévision va utiliser des résultats issus du run d'analyse
 - ↳ **analyserun=false** : pas de run d'analyse lancé, le run de prévision utilise uniquement des données fournies par la POM (qui peuvent être des données observées et des données prévues)
- ✓ La balise « **complet** » du fichier « session.xml » permet de préciser si le run de prévision sera « COMPLET ». Cette balise n'est utilisée que dans le cas où « **analyserun=true** »
 - ↳ **complet=false** : la plage de date pour le run de prévision est [T1 – T2]
 - ↳ **complet=true** : la plage de date pour le run de prévision est [T0 – T2] avec T0=date minimum des données observées fournies par la POM.

Note : dans ce cas, le run d'analyse produit un fichier relay à T1, mais ce fichier relay produit à T1 n'est pas utilisé par le run de prévision sur [T0-T2].

2.9.2 Gestion des modèles GSD

Exemples d'entrées possibles pour un calcul de prévision d'un modèle GSD :

- ✓ un débit observé à une station amont,
- ✓ un HU moyen sur un BV,
- ✓ une lame d'eau observée moyenne Antilope sur le BV,
- ✓ une lame d'eau prévue moyenne Sympo sur le BV.

2.9.3 Gestion des modèles distribués

Exemples d'entrées possibles pour un calcul de prévision d'un modèle distribué :

- ✓ un débit observé à un site amont (apport),
- ✓ un débit de base à un autre site hydro,
- ✓ un HU spatialisé sur un BV,
- ✓ une lame d'eau observée spatialisée Antilope sur le BV,
- ✓ une lame d'eau prévue moyenne Sympo sur le BV.

2.10 Archive de reprise – fichier « relay »

A chaque lancement du solver, un (ou des) fichier(s) de reprise (ou « relay ») est (sont) généré(s) dans un répertoire spécifique de résultats :

```
Ev_[EVENEMENT]/Relay.
```

Soit dans le cas présent (uniquement pour les runs d'analyse)

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/  
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/ev_Analyse/Relay
```

Pour un run déterministe, un seul fichier est généré :

```
relay_[DATE_FIN_ANALYSE].txt
```

Pour un run ensembliste, un fichier relay est généré par nombre d'ensemble où [NUMERO_MEMBRE] est codé sur 3 chiffres (001, 002, etc.) :

```
relay_[DATE_FIN_ANALYSE].[NUMERO_MEMBRE].txt
```

En mode déterministe, le fichier de reprise est à référencer dans le fichier « .evt ».

En mode ensembliste, il faut référencer tous les fichiers de reprise.

cf §1.6.3.2 - Fichiers événement « .evt ».

3. Périmètre fonctionnel du PIpt

Le présent chapitre décrit les étapes à mener pour le lancement des calculs PLATHYNES. Il détaille les étapes à modifier par rapport au PI.

3.1 Lancement

Le lancement du PIpt est identique au lancement du PI (générique) : mêmes arguments de ligne de commande, avec la possibilité facultative de renseigner un fichier « .ini ».

Note : le nom du fichier « .ini » utilisé par défaut est « pipt.ini ». (cf 1.6.2)

3.2 Ordonnanceur de tâches

L'ordonnanceur hérite de la classe « ModelBase » et s'appelle « ModelPipt ». Il est créé lors du démarrage par le « StartPiptProcessor ».

La méthode principale d'ordonnancement n'est pas modifiée, elle reprend donc les étapes du PI :

- ✓ Initialisation
- ✓ Nettoyage pré run
- ✓ Traitement des scénarios
- ✓ Nettoyage post run

Comme indiqué ci-avant, le PIpt est paramétré pour lancer au moins un run par scénario : le lancement éventuel de run après le traitement des scénarios (fonctionnalité utilisée dans le PIG) n'est donc pas effectué.

Pour la prise en compte du multi modèles, le PIpt utilise le fonctionnement du PI3 qui intègre dans ses fonctions de base la gestion multi-modèles :

- ✓ vérification de cohérence des scénarios et des modèles (cf specs PI 3 §4.3.2)

3.3 Initialisation

L'initialisation du PIpt est identique à celle du PI aux points ci-dessous près :

- ✓ La lecture du fichier « .ini » est adaptée pour lire les paramètres spécifiques au PIpt (cf. 1.6.2).
- ✓ La phase d'initialisation vérifie également la version de PLATHYNES qui doit être égale à celle paramétrée dans le fichier « .ini » du PIPT (cf. 1.6.2). La version PLATHYNES est stockée dans la variable « l_versionPlathynes » du fichier :

```
CODE\src\gui\common\globalvars.py
```

3.4 Traitement des scénarios (ScenarioPIPtProcessor)

3.4.1 Vérifications

Chaque scénario doit faire l'objet de vérifications de cohérence :

- ✓ La syntaxe du mode de calcul est vérifiée (cf. 2.1 - Mode de calcul POM)
- ✓ Chaque code modèle à lancer (renseigné dans le mode de calcul, cf. 2.1 - Mode de calcul POM) doit correspondre à un répertoire d'export PLATHYNES (cf. 1.7.2.3 - Répertoire opérationnel).
- ✓ Pour chacun de ces modèles, il faut lire le fichier « session.xml » :
 - charger la liste des sites hydro. Si un site hydro de sortie apparaît dans deux fichiers session.xml, une erreur bloquante est levée (un même site hydro ne doit pas faire l'objet de deux prévisions différentes dans un même scénario POM)
 - charger la liste des structures. Le PIPt vérifie que toutes les stations associées (à chaque structure, à chaque paramètre de structure) existent dans les entités du scénario POM. Si des stations n'existent pas, le PIPt génère une erreur bloquante avec la liste des stations inexistantes et leur association aux structures.

Exemple : « la station 'A123456789' associée au paramètre 'Qrestitue' de la structure 'Barrage1' n'existe pas dans le référentiel du scénario 'Sp001'.
- ✓ Pour chacun de ces modèles, il faut lire le fichier « .mwc » :
 - charger la liste des stations. L'identifiant des stations peut être un nom (« Millau », « Agen », « marmande »...) ou un code POM (« O6400010 », « O8481530 »...)
 - charger la liste des noms des stations 'baseflow' : le nom est défini dans la 4ème colonne de la ligne 'Baseflow' :

```
Baseflow: FIRST_OBS Millau Millau 1.0
              ^^^^^^
              4ème colonne lue
```

La correspondance entre le nom et le code POM des stations sera faite au moment de l'écriture des fichiers « mrr », « mqi », « mqo », « mhi », cf 3.5.6 - Gestion des codes d'entités.

La liste des noms des stations 'Baseflow' sera utilisée lors de la vérification des débits observés quand il n'y a pas de fichier de reprise (cf 2.8.2 - Débit de base).

3.4.2 Gestion multi-modèles

Pour la prise en compte du multi modèles, le PIPt utilise le fonctionnement du PI3 qui intègre dans ses fonctions de base la gestion multi-modèles :

- ✓ **pour chaque modèle**, il y a création d'un LaunchProcessor (cf specs PI 3 §4.6.4.1) qui effectue la préparation et le lancement d'un run

La condition suivante est spécifique au PIPt3 :

- ✓ pour un même scénario, tous les modèles doivent être du même type (soit GSD, soit distribué).

3.5 Traitement des entrées

La lecture des entrées est réalisée par le PI.

Pour chaque scénario, seules les données du scénario courant sont lues.

Les données sont filtrées pour ne conserver que les données des entités qui sont configurées dans le modèle.

La gestion des chevauchements de données est faite dans le PI3 (cf spécifications PI, § 4.7.5.2).

Le PIPIt ajoute les vérifications suivantes en fonction du type de métadonnées.

3.5.1 Métadonnées Image (MDBDIMAGE)

3.5.1.1 Grandeur HU (MDBDIMAGE)

Plusieurs ressources HU peuvent être fournies en entrée, le code du site hydro est utilisé pour filtrer la bonne ressource au modèle du run en cours. A noter que ce filtrage est effectué pour toutes les entrées, cf paragraphe précédent.

Chaque ressource HU ne doit être associée qu'à une et une seule entité. Dans le cas contraire une erreur bloquante est levée.

Chaque ressource HU ne doit être associée qu'à une et une seule série, dont la valeur ne peut être que « PIXELS » ou « MOY » . Dans le cas contraire une erreur bloquante est levée.

3.5.1.2 Grandeur RR et série « pixels » (MDBDIMAGE)

La métadonnée peut être associée à différentes zones (différentes « entités »). Il faut ajouter autant de données que d'entités associées à la métadonnée.

3.5.1.3 Grandeur RR et série sans « pixels » (MDBDIMAGE)

Cette entrée est une donnée de pluie observée ponctuelle.

La métadonnée ne doit être associée qu'à une et une seule série. Dans le cas contraire, une erreur bloquante est levée.

Comme pour les pluies spatialisées, la métadonnée peut être associée à plusieurs zones (entités), mais dans ce cas, un seul fichier de paramétrage suffit, dans lequel on peut positionner toutes les entités.

3.5.1.4 Autres possibilités

Les autres possibilités ne sont pas prises en compte et génèrent une erreur bloquante.

3.5.2 Métadonnées d'observation PHyC (MDOBSBDH)

3.5.2.1 Grandeur Q (MDOBSBDH)

La métadonnée peut être associée à plusieurs entités (sites hydro).

Rappel § 3.6.3.1.4.3 : S'il y a un paramètre événementiel 'Q' avec une station associée, le PIPt vérifie la présence de données d'entrée pour cette station.

Si ce n'est pas le cas, une erreur bloquante est générée.

3.5.2.2 Grandeur RR (MDOBSBDH)

Ce cas est traité comme le cas des données XML Image statistiques (cf. 3.6.3.1.3), à savoir des données de pluie ponctuelles/globale observée.

3.5.2.3 Grandeur H (MDOBSBDH)

La métadonnée peut être associée à plusieurs entités (stations hydro).

3.5.2.4 Autres grandeurs

Les autres grandeurs ne sont pas prises en compte et génèrent une erreur bloquante.

3.5.3 Métadonnées BP (MDBDLAMEDOBP)

Ce type de données est exploité de la même manière que les données de pluies ponctuelles (cf. 3.6.3.1.3). Les valeurs sont les valeurs associées à la série « MOY ». Si elle n'est pas renseignée, une erreur bloquante est levée.

3.5.4 Métadonnées fichier (MDFICHIER)

Une seule métadonnée de type « fichier » maximum est attendue par scénario POM. Dans le cas contraire une erreur bloquante est levée.

Cette métadonnée doit être associée à une archive « ZIP » contenant une liste de fichiers dont l'extension est « grd » ou « asc ». Dans le cas contraire une erreur bloquante est levée.

3.5.5 Métadonnées de prévision interne (MDPREVINT) et externe (MDPREVEXT)

Si la grandeur est Q, cette entrée est gérée comme une MDOBSBDH de grandeur Q (cf 3.6.3.1.4). Sinon une erreur bloquante est levée.

3.5.6 Gestion des codes d'entités

Dans les fichiers « .mrr », « .mqi », « .mqo », il y a des références aux stations.

Le PIPT va toujours utiliser les codes POM des sites hydro/sites météo (« O6400010 », « O8481530 »...) pour générer ces fichiers, même si le fichier « .mwc » ne contient que les noms des stations (« Millau »...).

Pour cela, le PIPT utilise les fichiers « session.xml » et « .mwc » pour avoir une correspondance entre les noms et les codes POM des sites hydro/sites météo, cf 3.6.3.4 - Modification des fichiers « .mwc ».

Dans les fichiers « .mhi », il y a des références aux stations.

Le PIPT va toujours utiliser les codes POM des stations (« O640001000 », « O848153000 »...) pour générer ces fichiers. Pour cela, le PIPT utilise le fichier « session.xml » (attributs « station-name » et « station-code ») pour avoir une correspondance entre les noms des stations associées aux structures et les codes POM des stations du scénario.

3.5.7 Pluviométrie ponctuelle

Les données de pluviométrie ponctuelles sont les données non spatialisées. Cela inclut les données moyennes sur une zone (BP, stats image (Antilope, Sympo), site météo...).

Elles doivent être associées à un pluviomètre (côté Plathynes) pour être prises en compte par le solveur qui répartit uniformément la pluie (dans le cas des modèles spatialisés)

- ✓ sur le bassin : chaque maille est alimentée par une valeur commune.
- ✓ dans le temps, sur chaque pas de calcul, pour les données dont le pas de temps est supérieur au pas de calcul. Par exemple, les pluies BP à 24h sont réparties sur chaque pas de calcul (5 minutes).

3.6 Run

Le lancement des runs de Plathynes suit la procédure du PI. Les chapitres ci-dessous précisent les fonctions à surcharger.

Pour rappel, les runs sont liés au scénario POM en cours de traitement.

3.6.1 Types de run

Chaque scénario POM donne lieu à différents types de runs :

- ✓ un run d'analyse si le scénario POM contient des données situées avant le temps de base
- ✓ un ou plusieurs run de prévision si le scénario contient des données après le temps de base :
 - ✎ si le mode de calcul est **ENSEMBLISTE**
 - si la POM a envoyé une seule série, ou si la clé « series » contient une seule valeur, le PIPT lance le calcul ENSEMBLISTE sur la seule série définie
 - si la POM a envoyé plusieurs séries de prévision et si la clé « series » contient plusieurs valeurs, alors le PIPT génère une erreur bloquante avec le message « un calcul ENSEMBLISTE ne doit avoir qu'une seule série de prévision en entrée »
 -

- ↳ si le mode de calcul est **DETERMINISTE**, le PIPT fait la propagation des séries : il boucle sur les séries définies par la clé « series » avec les données d'entrées correspondantes

3.6.2 Copie des fichiers du modèle

Pour chacun des codes modèle à lancer (cf §3.4.2 - Gestion multi-modèles), il faut :

- ✓ Créer le répertoire de travail Plathynes selon l'arborescence prévue (cf §1.7.1 - Arborescence de travail du PIPT).

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/analyse
```

ou

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/{NOM_SERIE}
```

- ✓ Copier le paramétrage du modèle en question depuis le répertoire d'export PLATHYNES :
 - ↳ Fichier « *.mwc ». Le nom du fichier « mwc » de l'export PLATHYNES est renseigné dans la balise « settings → model » du fichier « session.xml »
 - ↳ Fichier « *.net ». le nom du fichier « net » de l'export PLATHYNES est renseigné dans la clé « Network file » du fichier « mwc »
 - ↳ Répertoire « parameters »

3.6.3 Paramétrage de la plateforme

3.6.3.1 Génération des fichiers « .evt »

- ✓ Pour un run d'analyse, il faut générer le répertoire et le fichier correspondant :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/analyse/ev_Analyse/
ev_Analyse.evt
```

- ✓ Pour un run de prévision, il faut générer le répertoire et le fichier correspondant :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/{NOM_SERIE}/
ev_{NOM_SERIE}/ev_{NOM_SERIE}.evt
```

Le fichier « .evt » se constitue comme suit :

```
#=====
# event settings
#=====
```

```

Nom de l'evenement: {CODE_SESSION_POM} {DATE_PIVOT}
{MODE_CALCUL_POM} {CODE_SCENARIO_POM}
Identifiant: ev_Analyse
Description:

#=====
# Temporal settings
#=====
Date de debut: [DATE_DEBUT]
Date de fin: [DATE_FIN]
Pas de temps de forçage: [PDT_FORCAGE]
Pas de temps de calcul: [PDT_CALCUL]
Pas de temps des sorties: [PDT_SORTIES]
Pas de temps des bilans: 01:00

#=====
# Parameters
#=====
[PARAMETERS]

#=====
# Reprise
#=====
Fichier relais: [FICHIER_RELAY]

#=====
# Forcing settings
#=====
Nombre de sources de pluies: [NB_PLUIES]
Source de pluie: ev_Analyse/[FICHIER_MRR1]
Source de pluie: ev_Analyse/[FICHIER_MRR2]
...
Nombre de sources de debits: [NB_DEBITS]
Source de debits: ev_Analyse/[FICHIER_MQI_DEBIT1]
Source de debits: ev_Analyse/[FICHIER_MQI_DEBIT2]
...

#=====
# Observations
#=====
Nombre de fichiers d'observations: [NB_OBS]
Fichier d'observation: ev_Analyse/[FICHIER_MQO_OBS1]
Fichier d'observation: ev_Analyse/[FICHIER_MQO_OBS2]
...

#=====
# Structure height files
#=====
Nombre de fichiers de hauteurs de barrages: [NB_MHI]
Fichier de hauteurs de barrage: ev_Analyse/[FICHIER_MHI_OBS1]

```

- ✓ [IDENTIFIANT] est « analyse » pour un run d'analyse, « {NOM_SERIE} » pour un run de prévision.
- ✓ Les dates sont au format « AAAA-MM-JJ HH:MM:SS »

- ✓ Les pas de temps sont au format « HH:MM »
- ✓ [DATE_DEBUT] :
- ✓ [DATE_FIN] :
 - ↳ cf §2.8.1 - Préparation du run d'analyse et § 2.9.1 - Préparation du run de prévision pour les règles utilisées pour le calcul de ces 2 dates.
- ✓ [PDT_FORCAGE] :
 - ↳ lors d'un run d'analyse, c'est égal au minimum des pas de temps des données d'entrées observées,
 - ↳ lors d'un run de prévision, c'est égal au pas de temps de forçage utilisé lors du run d'analyse.
- ✓ [PDT_CALCUL] : renseigné dans la balise optionnelle « settings → compute → dt » du fichier session.xml. Si la balise n'est pas renseignée, la valeur par défaut « 00:05 » est choisie.
- ✓ [PDT_SORTIES] : renseigné dans la balise « settings → outputs → dt » du fichier session.xml. S'il n'est pas renseigné, une erreur bloquante est levée.
- ✓ [PARAMETERS]
- ✓ Il y a trois états possibles pour cette section :
 - ↳ Etat 1 :

```
Nombre de parametres evenementiels: 1
Parametre evenementiel : R [TYPE_PARAM]
```

[TYPE_PARAM] vaut « HU2 » ou « Q » suivant le type du paramètre événementiel lu dans le fichier « session.xml »

↳ Etat 2 :

```
Nombre de parametres evenementiels: 1
Parametres evenementiel: [PARAM_HU]
```

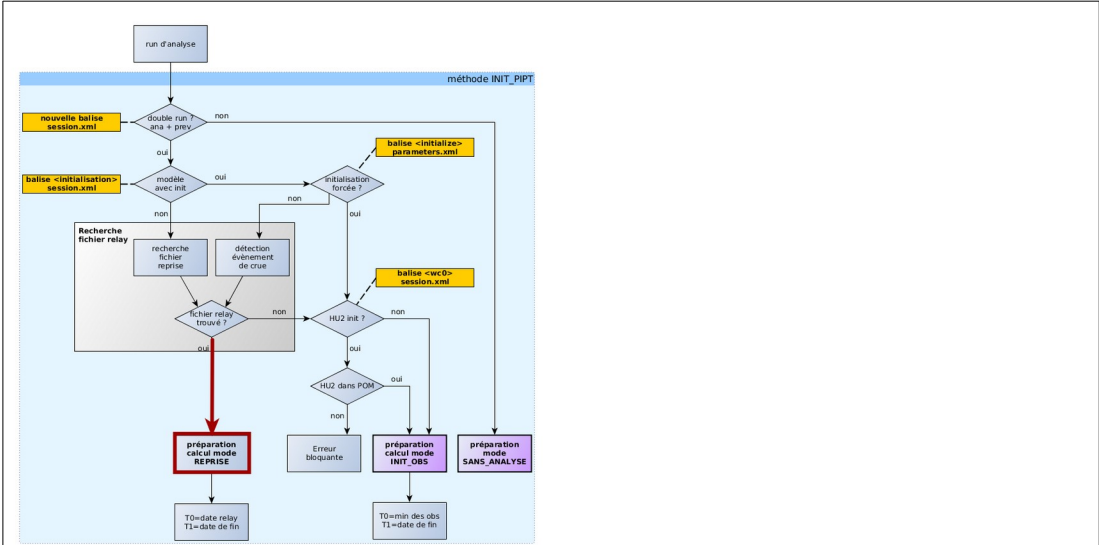
[PARAM_HU] : la génération du paramètre est indiqué au chapitre 3.6.3.1.1 pour « HU » et 3.6.3.1.4.3 pour « Q ».

↳ Etat 3 : section vide

Le tableau suivant décrit pour chaque cas de calcul analyse / prévision, l'état de cette section.

CALCUL ANALYSE

Mode REPRISE, avec fichier relay



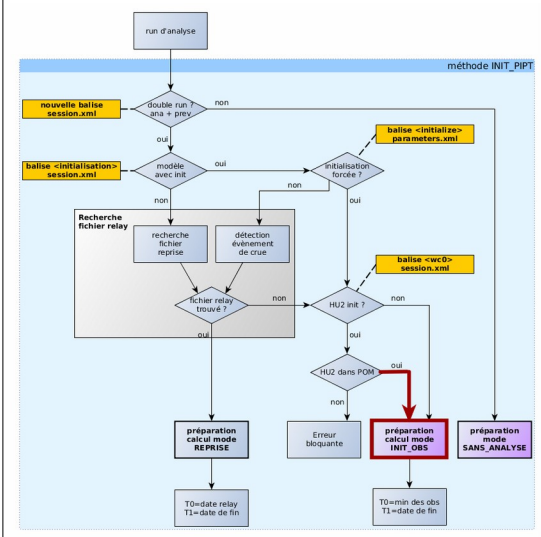
Le contenu de « Parameters » dépend du fichier relay :

- si le fichier relay contient des "EVENT PARAMETERS", la section contient (Etat 1) :

Nombre de parametres evenementiels: 1
Parametre evenementiel : R [TYPE_PARAM]

- si le fichier relay ne contient pas "EVENT PARAMETERS", la section est vide (Etat 3)

Mode INIT_OBS, avec wc0

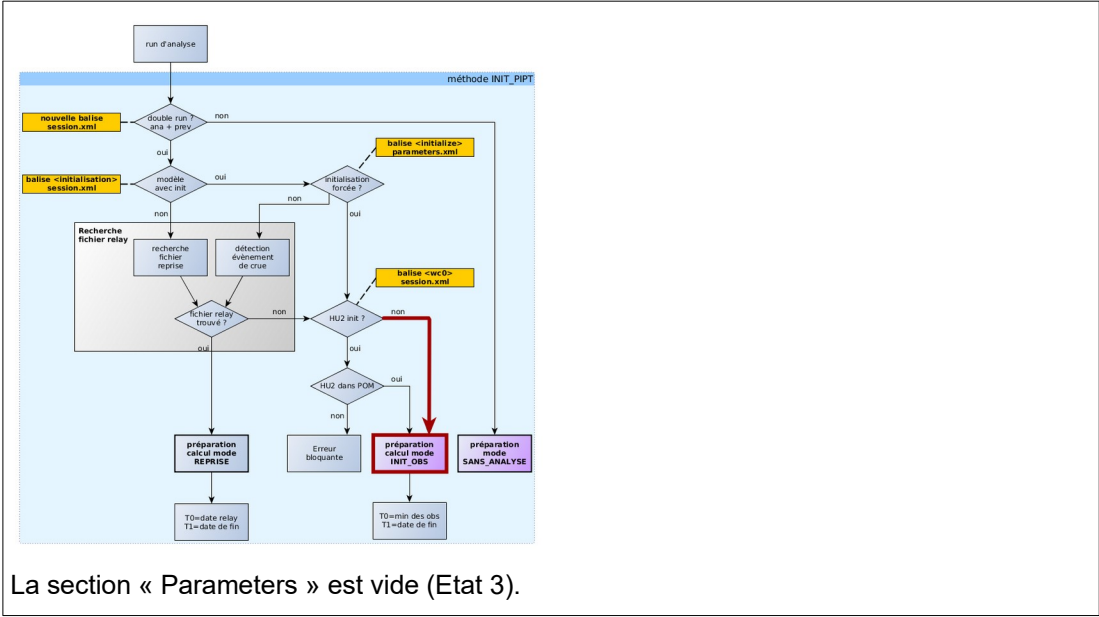


Le contenu de « Parameters » est Etat 2 :

Nombre de parametres evenementiels: 1
Parametres evenementiel: [PARAM_HU]

[PARAM_HU] : la génération du paramètre est indiqué au chapitre 3.6.3.1.1 pour « HU » et 3.6.3.1.4.3 pour « Q ».

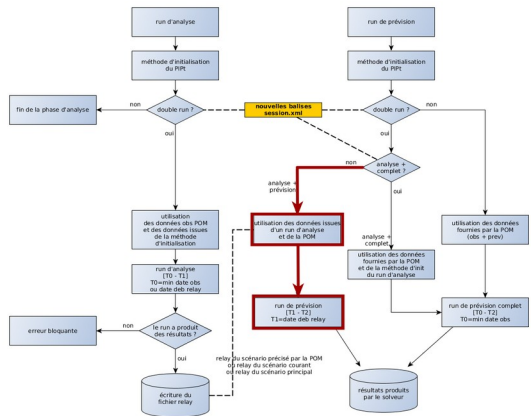
Mode INIT_OBS , sans wc0



La section « Parameters » est vide (Etat 3).

CALCUL PREVISION

Mode analyse+prevision



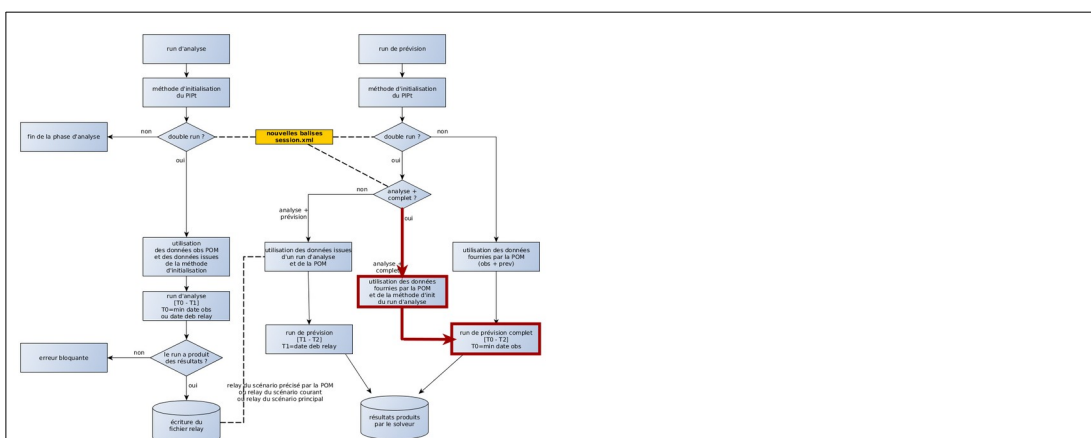
Le contenu de « Parameters » dépend du fichier relay :

- si le fichier relay contient des "EVENT PARAMETERS", la section contient Etat 1 :

Nombre de parametres eventementiels: 1
Parametre eventementiel : R [TYPE_PARAM]

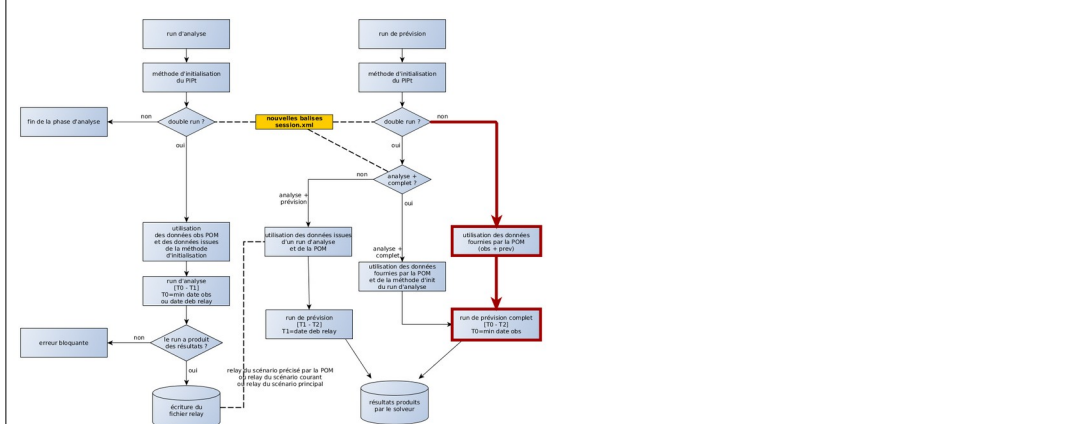
- si le fichier relay ne contient pas "EVENT PARAMETERS", la section est vide (Etat 3)

Mode analyse+complet



Le contenu de « Parameters » est identique à celui du calcul d'analyse (Etat 1 ou Etat 2 ou Etat 3).

Mode complet



La section « Parameters » est vide (Etat 3).

Tableau 2 : Définition de la section « Parameters » du fichier .evt

✓ « Reprise »

La section « Reprise » n'est présente que si un fichier relay est utilisé (run de prévision ou d'analyse avec initialisation sur fichier de reprise). Son contenu comprend la liste du(des) nom(s) de fichier de reprise (il peut y en avoir plusieurs pour les calculs ensemblistes).

- ✓ [NB_PLUIES] : nombre de fichier « .mrr » (paramétrage des entrées de type « pluie ») générés
- ✓ [FICHIER_MRR] : autant de lignes que de fichiers « .mrr » générés. Le nom du répertoire est fonction du type de run (ev_Analyse ou ev_Prevision).
- ✓ [NB_DEBITS] : nombre de fichier « .mqj » sur les stations « Qin » ou « Qbvi »

- ✓ [FICHIER_MQI_DEBIT] : autant de lignes que de fichiers « .mqi » générés pour les stations « Qin » ou « Qbvi » avec le chemin du fichier MQI (cf. 3.6.3.1.4). Le nom du répertoire est fonction du type de run (ev_Analyse ou ev_Prevision).
- ✓ [NB_OBS] : nombre de fichier « .mqo » générés sur les stations « Q »
- ✓ [FICHIER_MQO_OBS] : autant de lignes que de fichiers « .mqo » générés pour les stations « Q », avec le chemin du fichier MQO (cf. 3.6.3.1.4). Le nom du répertoire est fonction du type de run (ev_Analyse ou ev_Prevision).
- ✓ « Structure height files »
La section « Structure height files » n'est présente que si des fichiers « .mhi » sont générés pour des ouvrages.
- ✓ [NB_MHI] : nombre de fichier « .mhi » générés sur les structures
- ✓ [FICHIER_MHI_OBS] : autant de lignes que de fichiers « .mhi » générés pour les structures, avec le chemin du fichier MHI (cf 1.6.3.7). Le nom du répertoire est fonction du type de run (ev_Analyse ou ev_Prevision).

3.6.3.1.1 Grandeur HU

Note : si le fichier « evt » (cf. §1.6.3.2 - Fichiers événement « .evt ») indique un fichier de reprise (relay), l'initialisation HU est ignorée par le solver.

Cette entrée est considérée comme un paramètre événementiel. Il faut donc

- ✓ ajouter 1 au nombre de paramètres événementiels du fichier « evt »
- ✓ ajouter le paramètre événementiel en fonction de la série de la métadonnée POM

↳ Si elle vaut « PIXELS », la paramètre vaut :

```
Parametres evenementiel: G ev_Analyse/data/[CODE_RESSOURCE]_hu2.asc
```

Il faut alors générer le fichier « ev_Analyse/data/[CODE_RESSOURCE]_hu2.asc » au format GRD. Cf 1.6.3.9

↳ Si la série ne vaut pas « PIXELS », on utilise les valeurs de la série « MOYENNE ». Si elle n'est pas dans le fichier, une erreur bloquante est levée. Le paramètre événementiel vaut alors :

```
Parametres evenementiel: U [VALEUR] [NUM_FONCT_REGRES] XX
```

avec :

- [VALEUR] : valeur la plus récente de la série moyenne
- [NUM_FONCT_REGRES] : contenu de l'attribut « reg » de la balise « settings → initialisation → reg » du fichier « session.xml ». Si l'attribut n'existe pas, la valeur par défaut est « 0 ».

3.6.3.1.2 Grandeur RR et série « pixels »

La métadonnée peut être associée à différentes zones (différentes « entités »). Il faut ajouter autant de données que d'entités associées à la métadonnée. Les actions suivantes sont donc à réaliser pour chaque entité.

Mettre à jour le fichier « evt » :

- ✓ Ajouter 1 au nombre de sources de pluies
- ✓ Ajouter une source de pluie
« ev_Analyse/[CODE_RESSOURCE]_[CODE_ENTITE].mrr »

- ✓ Générer le fichier « .mrr » (nom ci-dessus) au format suivant :

```
#=====
# [CODE_RESSOURCE] [TYPE_METADONNEE] [GRANDEUR_METADONNEE]
#=====
Type de donnees : RR3
Station : [CODE_ENTITE]
Pas de temps : [PDT_METADONNEE_POM]
Facteur multiplicatif : 1
Repertoire des donnees : [CODE_MODELE_PLATHYNES]/[type_run]/data
[NOM DU FICHIER CSV 1]
[NOM DU FICHIER CSV 2]
...
```

- ↳ [CODE_ENTITE] est le code POM (site hydro, du site météo ou de la zone BNBV) de la station ou du site météo Plathynes, cf 3.5.6 - Gestion des codes d'entités.
Cette information n'est pas utilisée par le solver.
- ↳ le pas de temps des données est au format « JJ HH:MM:SS ».
- ↳ le pas de temps est la valeur des attributs « duree » des balises « image ». Toutes ces durées sont théoriquement identiques dans un même fichier.
- ↳ le facteur multiplicatif est à 1 car le solver ingère des données en 1/10ème de mm cumulé sur le pas de temps (cf. 2.6), comme le contenu du fichier.
- ↳ le répertoire des données est un chemin relatif au répertoire [RACINE_DIR], avec [type_run] égal à « Ev_Analyse » ou « Ev_Prevision ».
- ↳ un fichier CSV est généré pour chaque balise « Image » de l'entité correspondante du fichier XML fourni par la POM. Il est intitulé : [DATE_IMAGE].csv ([DATE_IMAGE] est au format AAAAMMJJ_HHMMSS), Chaque fichier « .csv » contient uniquement trois colonnes « [X] [Y] [RR] » avec autant de lignes que de pixels, avec une ligne d'en tête générale (non exploitée).

Exemple de fichiers, cf 1.6.3.8.1 - Fichier « mrr » de données non pixelisés.

3.6.3.1.3 Grandeur RR et série sans « pixels »

Comme pour les pluies spatialisées, la métadonnée peut être associée à plusieurs zones (entités), mais dans ce cas, un seul fichier de paramétrage suffit, dans lequel on peut positionner toutes les entités. Les actions suivantes sont donc à réaliser une seule fois par grandeur RR sans pixels.

Mettre à jour le fichier « evt » :

- ✓ Ajouter 1 au nombre de sources de pluies
- ✓ Ajouter une source de pluie « ev_Analyse/[CODE_RESSOURCE].mrr »
- ✓ Générer le fichier « .mrr » ci-dessus au format suivant :

```
#=====
# [CODE_RESSOURCE] [TYPE_METADONNEE] [GRANDEUR_METADONNEE]
#=====
Type de donnees : PLUVIO
Station : [METADATA_CODE]
Pas de temps : [PDT_METADONNEE_POM]
```

```
Facteur multiplicatif : 1
[NB_ENTITES_MD_POM]
[X1] [Y1]
[X2] [Y2]
...
[DATE YYYY-MM-JJ HH:MM] [VALEURS1] [VALEUR2] ...
```

- ↪ le [METADATA_CODE] est le code POM (site hydro, du site météo ou de la zone BNBV) de la station ou du site météo Plathynes, cf 3.5.6 - Gestion des codes d'entités
- ↪ Le pas de temps des données est au format « JJ HH:MM:SS »
- ↪ Le pas de temps est la valeur des attributs « duree » des balises « image ». Toutes ces durées sont théoriquement identiques dans un même fichier.
- ↪
- ↪ le facteur multiplicatif est à 1 car le solver ingère des données en 1/10ème de mm cumulé sur le pas de temps (cf. 2.6), comme le contenu du fichier.
- ↪ [NB_ENTITES_MD_POM] est le nombre d'entités associées à la métadonnée POM
- ↪ [Xi] [Yi] : une ligne pour chaque entité de la métadonnée. Les coordonnées X et Y sont lues dans le fichier « session.xml » (balises « siteMeteo »). Si l'entité n'est pas dans le fichier « session.xml » une erreur bloquante est levée.
- ↪ Les valeurs sont les valeurs associées à la série « MOYENNE ». Si elle n'est pas renseignée, une erreur bloquante est levée. Il y a [NB_ENTITES_MD_POM] valeurs par ligne. Ces valeurs sont séparées par des espaces.

Exemple de fichiers , cf 1.6.3.8.1 - Fichier « mrr » de données non pixelisés.

3.6.3.1.4 Grandeur Q

La métadonnée peut être associée à plusieurs entités (sites hydro). Les actions suivantes sont donc à réaliser pour chaque entité de la métadonnée.

3.6.3.1.4.1 Si l'entité est de type « Q »

Le PIPT vérifie s'il n'a pas déjà généré un fichier « .mqo » pour cette entité.

- ✓ S'il existe déjà un fichier, le PIPT ajoute les données à la fin du fichier, au format :

```
[DATE] [HEURE] [VALEUR]
```

- ✓ S'il n'existe pas encore de fichier, le PIPT effectue les opérations suivantes :

Mettre à jour le fichier « evt » :

- ✓ Vérifier que l'entité a une balise « siteHydro » de même code dans le fichier « session.xml »
- ✓ Ajouter 1 au nombre de fichiers d'observations
- ✓ Ajouter un fichier d'observation
« ev_Analyse/[CODE_RESSOURCE]_[CODE_ENTITE].mqo »
- ✓ Générer le fichier comme suit :

```
[CODE_RESSOURCE] [TYPE_METADONNEE] [GRANDEUR_METADONNEE]
```

```
[CODE_ENTITE]
1
StationID X Y Type
[CODE_ENTITE] [X] [Y] Qobs
Date Time Qobs [m3/s]
[DATE] [HEURE] [VALEUR]
```

- ↳ lignes 1 et 2 : entête quelconque, non exploitée par le solver
- ↳ ligne 2 : le [CODE_ENTITE] est le code POM (Site hydro) de la station Plathynes, cf 3.5.6
- ↳ ligne 3 : nombre d'entités du fichier (ici 1)
- ↳ ligne 4 : en tête du tableau des stations
- ↳ ligne 5 (autant que de stations du fichier, donc ici 1) : Code, coordonnées X et Y (lues dans le fichier « session.xml »), chaîne « Qobs »
- ↳ ligne 5 : le [CODE_ENTITE] est le code POM (Site hydro) de la station Plathynes, cf 3.5.6
- ↳ Ligne 6 : en tête du tableau des valeurs (Date Time Qobs [m³/s])
- ↳ Lignes suivantes : données observées pour chaque date
 - [DATE] : date au format AAAA-MM-JJ
 - [HEURE] : heure au format HH:MM:SS
 - [VALEUR] : valeur
- ↳

3.6.3.1.4.2 Si l'entité est de type « Qin » ou « Qbvi »

Le PIPT vérifie s'il n'a pas déjà généré un fichier « .mqi » pour cette entité.

- ✓ S'il existe déjà un fichier, le PIPT ajoute les données à la fin du fichier, au format :

```
✓ [DATE] [HEURE] [VALEUR]
```

- ✓ S'il n'existe pas encore de fichier, le PIPT effectue les opérations suivantes :

Mettre à jour le fichier « evt » :

- ✓ Vérifier que l'entité a une balise « siteHydro » de même code dans le fichier « session.xml »
- ✓ Ajouter 1 au nombre de fichiers de source de débit
- ✓ Ajouter un fichier d'observation
« ev_Analyse/[CODE_RESSOURCE]_[CODE_ENTITE].mqi »
- ✓ Générer le fichier comme suit :

```
[CODE_RESSOURCE] [TYPE_METADONNEE] [GRANDEUR_METADONNEE]
[CODE_ENTITE]
1
StationID X Y Type
[CODE_ENTITE] [X] [Y] Qobs
Date Time Qobs [m3/s]
[DATE] [HEURE] [VALEUR]
```

- ↳ lignes 1 et 2 : entête quelconque, non exploitée par le solver
- ↳ ligne 2 : le [CODE_ENTITE] est le code POM de la station, cf 3.5.6

- ↪ ligne 3 : nombre d'entités du fichier (ici 1)
- ↪ ligne 4 : entête du tableau des stations
- ↪ ligne 5 (autant que de stations du fichier, donc ici 1) : Code, coordonnées X et Y (lues dans le fichier « session.xml »), chaîne « Qobs »
- ↪ ligne 5 : le [CODE_ENTITE] est le code POM de la station, cf 3.5.6
- ↪ Ligne 6 : entête du tableau des valeurs (Date Time Qobs [m³/s])
- ↪ Lignes suivantes : données observées pour chaque date
 - [DATE] : date au format AAAA-MM-JJ
 - [HEURE] : heure au format HH:MM:SS
 - [VALEUR] : valeur

3.6.3.1.4.3 Si l'entité est liée à un paramètre événementiel « Q »

Note : si le fichier « evt » (cf. §1.6.3.2 - Fichiers événement « .evt ») indique un fichier de reprise (relay), l'initialisation Q est ignorée par le solver.

Cette entrée est considérée comme un paramètre événementiel. Il faut donc

- ✓ incrémenter le nombre de paramètres événementiels du fichier « evt »
- ✓ ajouter le paramètre événementiel
 - ↪ le PIPT utilise les valeurs de la série « MOYENNE ». Le paramètre événementiel vaut alors :

```
Parametres evenementiel: U [VALEUR] [NUM_FONCT_REGRES] XX
```

avec :

- [VALEUR] : valeur la plus récente de la série moyenne
- [NUM_FONCT_REGRES] : contenu de l'attribut « reg » de la balise « settings → initialisation → reg » du fichier « session.xml ». Si l'attribut n'existe pas, la valeur par défaut est « 0 ».

Si il n'y a pas de données d'entrée pour l'entité pour la série « MOYENNE », une erreur bloquante est levée.

3.6.3.1.5 Grandeur H

Le PIPT ne génère des fichiers « mhi » qu'en mode Analyse et quand il n'y a pas d'événement (c'est à dire, pas de fichier relais).

Dans ce cas, le PIPT effectue les opérations suivantes :

Mettre à jour le fichier « evt » :

- ✓ Vérifier que l'entité a une balise « siteHydro » de même code dans le fichier « session.xml »
- ✓ Ajouter 1 au nombre de fichiers de structures
- ✓ Ajouter un fichier de hauteurs d'observation
« ev_Analyse/[CODE_RESSOURCE]_[CODE_ENTITE].mhi »
- ✓ Générer le fichier comme suit :

```
[CODE_RESSOURCE] [TYPE_METADONNEE] [GRANDEUR_METADONNEE]  
[CODE_ENTITE]  
1
```

```

StationID X Y Type
[CODE_ENTITE] 0.0 0.0 Hobs 0.0
Date Time Qobs [m]
[DATE] [HEURE] [VALEUR]

```

- ↳ lignes 1 et 2 : entête quelconque, non exploitée par le solver
- ↳ ligne 2 : le [CODE_ENTITE] est le code POM de la station, cf 3.5.6
- ↳ ligne 3 : nombre d'entités du fichier (ici 1)
- ↳ ligne 4 : entête du tableau des stations
- ↳ ligne 5 (autant que de stations du fichier, donc ici 1) : Code, chaîne « 0.0 0.0 Hobs 0.0 »
- ↳ ligne 5 : le [CODE_ENTITE] est le code POM de la station, cf 3.5.6
- ↳ Ligne 6 : entête du tableau des valeurs (Date Time Hobs [m])
- ↳ Lignes suivantes : données observées pour chaque date
 - [DATE] : date au format AAAA-MM-JJ
 - [HEURE] : heure au format HH:MM:SS
 - [VALEUR] : valeur

3.6.3.1.6 Métadonnées fichier (MDFICHIER)

Cette métadonnée doit être associée à une archive « ZIP » contenant une liste de fichiers dont l'extension est « grd » ou « asc ». Dans le cas contraire une erreur bloquante est levée.

Le nom de ces fichiers contient la date des données du fichier. Un paramètre du fichier « ini » (cf. 1.6.2) permet de spécifier ce nom et le format de date attendu. Il faut ensuite calculer le pas de temps des données (il peut y en avoir plusieurs). Pour chaque fichier :

- ✓ on détermine sa date à l'aide du nom du fichier et du masque indiqué dans le fichier « ini » (cf. 1.6.2)
- ✓ si cette date est inférieure ou égale au temps de base, le fichier n'est pas pris en compte.
- ✓ on compare cette date à la date du précédent fichier :
 - ↳ S'il existe : la différence de dates est le pas de temps des données. On ajoute alors ce fichier à la liste des fichiers « grd » ou « asc » de ce pas de temps
 - ↳ S'il n'existe pas, on stocke ce nom de fichier GRD pour l'ajouter au pas de temps calculé sur le fichier suivant.

Pour chaque pas de temps [PDT] (en minutes) calculé précédemment :

- ✓ Ajouter 1 au nombre de sources de pluies du fichier « evt »
- ✓ Ajouter une source de pluie
« ev_Prevision/[CODE_RESSOURCE]_[CODE_ENTITE]_GRD_[PDT].mrr »
- ✓ Copier les fichiers « grd » de ce pas de temps dans le répertoire « ev_Prevision/data »
- ✓ Générer le fichier « .mrr » (nom ci-dessus) au format suivant :

```

#=====
# [CODE_RESSOURCE] [TYPE_METADONNEE] [GRANDEUR_METADONNEE] [PDT]
#=====
Type de donnees : GRD
Station : [CODE_ENTITE_EXUTOIRE]
Pas de temps : [PDT_METADONNEE_POM]

```

```
Facteur multiplicatif : 1
Repertoire des donnees : [CODE_MODELE_PLATHYNES]/Ev_Prevision/data
[NOM DU FICHIER GRD 1]
[NOM DU FICHIER GRD 2]
...
```

- ↳ La station exutoire est la station dont le bassin versant a la plus grande surface (cf. fichier « .mwc »)
- ↳ le [CODE_ENTITE_EXUTOIRE] est le code POM de la station, cf 3.5.6
Cette information n'est pas utilisée par le solver.
- ↳ le pas de temps des données est au format « JJ HH:MM:SS ». Il est calculé en fonction des dates des fichiers (déduite de leur nom). Le pas de temps est le plus petit écart de dates entre deux fichiers.
- ↳ le pas de temps est la valeur du pas de temps courant sur lequel on est en train de boucler.
- ↳ le facteur multiplicatif est à 1 car le solver ingère des données en 1/10ème de mm cumulé sur le pas de temps (cf. 2.6), comme le contenu du fichier.
- ↳ le répertoire des données est un chemin relatif au répertoire [RACINE_DIR].
- ↳ la fin du fichier comporte une ligne par fichier GRD associé à ce pas de temps.

3.6.3.1.7 Finalisation pour les fichiers « mqi » et « mqo »

Quand toutes les données ont été ajoutées aux fichiers « mqi » et « mqo », le PIPt trie le contenu de chacun de ces fichiers, par date croissante.

3.6.3.2 Contrôle des structures

En mode Analyse et s'il n'y a pas d'événement, le PIPt vérifie le point suivant pour les structures :

- ✓ chaque structure doit avoir des données H associées, un fichier « mhi » doit exister.
Si ce n'est pas le cas, une erreur bloquante est levée.

3.6.3.3 Génération des fichiers « .sim »

- ✓ pour un run d'analyse, il faut générer le fichier correspondant :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/analyse/Analyse.sim
```

- ✓ Pour un run de prévisions, il faut générer le fichier correspondant :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/prevision/Prevision.sim
```

Dans ce qui suit, [ANALYSE_OU_PREVISION] vaut « Analyse » pour le fichier d'analyse, « Prevision » pour le fichier de prévision.

Le fichier de simulation « .sim » contient le nom du fichier du modèle « .mwc » (renseigné dans la balise « settings → model » du fichier « session.xml »), les critères de calcul et les informations d'événements :

- ✓ nombre d'événements
- ✓ fichiers de configuration d'événements « .evt » (dans notre cas un seul)

```
#=====
# MWC file
#=====
MWC file: {NOM_BASSIN}.mwc

#=====
# Events
#=====
Number of events: 1
Event file: ev_[ANALYSE_OU_PREVISION]\ev_[ANALYSE_OU_PREVISION].evt
```

3.6.3.3.1 Définition de la section « Ensemble »

La section « Ensemble » est générée dans le fichier « .sim » si le mode ensembliste est activé **ou** si le mode assimilation est activé.

La ligne « Taille ensemble modele:[NUM_ENSEMBLE] » est générée dans le fichier « .sim » **si et seulement si** le mode ensembliste est activé.

La ligne « Taille ensemble modele:1 » est générée dans le fichier « .sim » **si et seulement si** le mode assimilation est activé **et si** le mode ensembliste n'est pas activé.

```
#=====
# Ensemble
#=====
Taille ensemble modele: [NUM_ENSEMBLE]
Nombre de parametres variables : [NB_PARAM]
Parametre variable : [TYPE_FCT] [UNITE] [IDX_PARAM_FCT] [MIN] [MAX]
[PRECISION]
```

où

- ✓ [NUM_ENSEMBLE] est le nombre d'ensemble paramétré dans le mode de calcul POM.
- ✓ [NB_PARAM] est le nombre de paramètres variables, définis dans les balises « settings → parameters → parameter » du fichier « session.xml »
- ✓ [TYPE_FCT] : 1^{er} élément du triplet (séparé par des « , ») contenu dans la balise « settings → parameters → parameter »
- ✓ [UNITE] : 2^{ème} élément du triplet (séparé par des « , ») contenu dans la balise « settings → parameters → parameter »
- ✓ [IDX_PARAM_FCT] : 3^{ème} élément du triplet (séparé par des « , ») contenu dans la balise « settings → parameters → parameter »
- ✓ [MIN] : contenu de l'attribut « min » de la balise « settings → parameters → parameter »

- ✓ [MAX] : contenu de l'attribut « max » de la balise « settings → parameters → parameter »
- ✓ [PRECISION] : 1 % de [MIN]

3.6.3.3 Définition de la section Assimilation

La section « Assimilation » est générée dans le fichier « .sim » **si et seulement si** le mode assimilation est activé **et si** le run est un run d'analyse.

```
#=====
# Assimilation
#=====

Fonction cout: NASH(Qth=[Q],Wsta=[L])
Fenetre de temps des fonctions cout: [F]
Data assimilation: active
```

où :

- ✓ [Q] vaut le contenu de la balise « settings → assimilation → station → threshold » du fichier « session.xml ». Si cette balise n'existe pas ou est vide, une erreur bloquante est levée.
- ✓ [L] vaut le contenu de la balise « settings → assimilation → station → weight » du fichier « session.xml ». Si cette balise n'existe pas ou est vide, une erreur bloquante est levée.
- ✓ [F] vaut le contenu de la balise « settings → assimilation → window » du fichier « session.xml ». Si cette balise n'existe pas ou est vide, une erreur bloquante est levée.

La balise « settings → assimilation → window » contient la fenêtre d'assimilation. Les données fournies doivent être renseignées sur toute cette plage. Dans le cas contraire, le PIPT génère une erreur.

3.6.3.4 Modification des fichiers « .mwc »

Dans le fichier « .mwc », le PIPT remplace les noms des stations (« Millau »...) par les codes POM des sites hydro/sites météo (« O6400010 », « O8481530 »...), cf 3.5.6 - Gestion des codes d'entités. Le PIPT fait ce remplacement pour les lignes « Station » et « Baseflow ».

Pour cela, il utilise les fichiers « session.xml » et « .mwc » pour avoir une correspondance entre les noms et les codes POM des sites hydro/sites météo.

Exemple :

Voici la partie des stations du fichier « .mwc » du modèle PLATHYNES :

```
#=====
# Stations
#=====

Number of stations: 4
Station: Millau 706974.50 6332974.50 2142750000.00 Q
```

```

Station: Montbrun 739474.50 6359474.50 598250000.00 Q
Station: Meyrueis 734474.50 6342474.50 97750000.00 Q
Station: Nant 724474.50 6324974.50 324500000.00 Q
Baseflow: FIRST_OBS Millau Millau 1.0

```

Voici la correspondance définie dans le fichier « session.xml » du modèle PLATHYNES :

```

<?xml version="1.0" ?>
<plathynesSession>
  <settings>...</settings>
  <databases>
    <databaseMeteo>...</databaseMeteo>
    <databaseHydro>
      <siteHydro>
        <name>Bedoues</name>
        <code>03031010</code>
        <x>749262.000000</x>
        <y>6360862.000000</y>
        <rsc/>
      </siteHydro>
      <siteHydro>
        <name>Florac</name>
        <code>03064030</code>
        <x>747012.000000</x>
        <y>6359362.000000</y>
        <rsc/>
      </siteHydro>
      <siteHydro>
        <name>Vebron</name>
        <code>03064040</code>
        <x>746262.000000</x>
        <y>6348962.000000</y>
        <rsc/>
      </siteHydro>
      <siteHydro>
        <name>Cassagnas</name>
        <code>03084350</code>
        <x>758362.000000</x>
        <y>6352562.000000</y>
        <rsc/>
      </siteHydro>
      <siteHydro>
        <name>Montbrun</name>
        <code>03121020</code>
        <x>739562.000000</x>
        <y>6359462.000000</y>
        <rsc/>
      </siteHydro>
      <siteHydro>
        <name>Meyrueis</name>
        <code>03194010</code>
        <x>734262.000000</x>
        <y>6342462.000000</y>

```

```

    <racs/>
  </siteHydro>
  <siteHydro>
    <name>Dourbies</name>
    <code>03314010</code>
    <x>737362.000000</x>
    <y>6330162.000000</y>
    <racs/>
  </siteHydro>
  <siteHydro>
    <name>Nant</name>
    <code>03334020</code>
    <x>724412.000000</x>
    <y>6324912.000000</y>
    <racs/>
  </siteHydro>
  <siteHydro>
    <name>Millau</name>
    <code>03401010</code>
    <x>705962.000000</x>
    <y>6332612.000000</y>
    <racs/>
  </siteHydro>
</databaseHydro>
</databases>
</plathynesSession>

```

Voici le fichier « .mwc » obtenu après remplacement :

```

#=====
# Stations
#=====
Number of stations: 4
Station: 03401010 706974.50 6332974.50 2142750000.00 Q
Station: 03121020 739474.50 6359474.50 598250000.00 Q
Station: 03194010 734474.50 6342474.50 97750000.00 Q
Station: 03334020 724474.50 6324974.50 324500000.00 Q
Baseflow: FIRST_OBS 03401010 03401010 1.0

```

3.6.4 Lancement d'un (des) run(s)

Seule la commande de lancement du solver est à lancer telle qu'indiquée au chapitre 1.9, dans le dossier « répertoire de travail Plathynes ».

3.7 Traitement des sorties

Cette étape vise à produire les fichiers de sortie attendus par la POM à partir des résultats générés par Plathynes.

- ✓ Résultats numériques au format XML Sandre attendu par la POM
- ✓ Archive ZIP éventuellement

La création des fichiers XML Sandre est effectuée de manière générique par le PI, dans les spécifications du PI, cf §4.9 - Sorties (OutputsBaseProcessor).

3.7.1 Sorties de type « fichier »

Une seule métadonnée de type fichier peut être positionnée en sortie. Si plusieurs métadonnées de type fichier existent, une erreur bloquante est levée.

La métadonnée de sortie est prévue pour contenir une archive ZIP contenant elle-même tous les fichiers de reprise utilisés pour les modèles PLATHYNES lancés dans ce scénario. Cela peut permettre de relancer Plathynes en mode étude (hors POM) à partir de ces fichiers.

Il s'agit des fichiers :

```
PIPt_WORKDIR/Archives/{CODE_MODEL_PLATHYNES}/
{DATE_PIVOT}_{MODE_CALCUL_POM}_{SCENARIO_POM}__{CODE_SESSION_POM}__{D
ET|ENS_XXX}[__ASSIM].zip
```

Si aucun fichier de reprise n'a été utilisé, l'archive est vide.

3.7.2 Sorties XML Sandre

3.7.2.1 Lecture des fichiers résultats

Les fichiers « XXXX_ResultsRaw.YYY » sont lus et chargés en mémoire par les méthodes `pix_get_hydrodata_analyse()` et `pix_get_hydrodata_prevision()`.

Le nombre de fichiers résultats à lire et la valeur de « YYY » est définie par :

- ↳ si le calcul est en mode DET sans assimilation :
 - 1 seul fichier résultat à lire,
 - YYY = « txt »
- ↳ si le calcul est en mode DET avec ASSIMILATION :
 - 1 seul fichier résultat à lire,
 - YYY = « 001 »
- ↳ si le calcul est en mode ENS :
 - autant de fichiers résultats que d'ensembles définis par la clé « num_ensemble »
 - YYY = « 001 » à N ; exemple : « 001 », « 002 », « 003 »... « 010 », « 011 »...

La valeur de XXXX est définie par :

- ↳ Pour un run d'analyse :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/analyse/ev_Analyse/
Results/analyse_Analyse_Analyse_ResultsRaw.YYY
```

↳ Pour un run de prévision :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/{NOM_SERIE}/
ev_{NOM_SERIE}/Results/
{NOM_SERIE}_{NOM_SERIE}_{NOM_SERIE}_ResultsRaw.YYY
```

Les valeurs produites par Plathynes sont en « m3/s » pour les débits.

Une conversion doit être effectuée car le XML Sandre contient des débits en « l/s ».

3.7.2.2 Lecture des fichiers résultats des barrages

Pour chaque barrage de chaque modèle, le PIPt cherche un fichier résultat de barrage nommé « XXXX_YYYY_structureResults.txt » (cf 1.6.3.13), avec :

La valeur de XXXX est définie par :

↳ Pour un run d'analyse :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/analyse/ev_Analyse/
Results/analyse_Analyse_Analyse_YYYY_structureResults.txt
```

↳ Pour un run de prévision :

```
PIPt_WORKDIR/{CODE_SESSION_POM}/{DATE_PIVOT}/{MODE_CALCUL_POM}/
{CODE_SCENARIO_POM}/{CODE_MODELE_PLATHYNES}/{NOM_SERIE}/
ev_{NOM_SERIE}/Results/
{NOM_SERIE}_{NOM_SERIE}_{NOM_SERIE}_YYYY_structureResults.txt
```

La valeur de YYYY est le nom du barrage.

Si le PIPt trouve ce fichier résultat, il le lit : il ne prend en compte que les colonnes associées aux paramètres de sortie définis dans le fichier « session.xml » pour ce barrage (cf 1.6.3.5).

Les valeurs produites par Plathynes sont en « m3/s » pour les débits.

Une conversion doit être effectuée car le XML Sandre contient des débits en « l/s ».

Aucune conversion n'est effectuée pour les autres types de valeurs.

3.7.2.3 Construction des données XML Sandre de sorties

En mode déterministe :

- ✓ si la série « MOY » et pas de série « MIN » ou « MAX », la série « MOY » est utilisée pour générer une prévision déterministe, balise « **PrevsDeterministe** »
- ✓ si une série « MIN » ou « MAX » : les séries « MIN », « MOY », « MAX » sont utilisées pour générer une prévision de tendance, balise « **PrevsTendance** »
- ✓ les séries probabilistes génèrent des prévisions probabilistes, balise « **PrevsProbs** »

En mode ensembliste :

- ✓ si incertitude = **membres**, des prévisions d'ensemble sont générées, balise « **PrevsEnsemble** »
- ✓ si incertitude = **quantiles**, les prévisions sont filtrées pour ne conserver que les prévisions des quantiles précisés, des prévisions probabilistes sont générées, balise « **PrevsProbs** ».

3.8 Finalisation

Cette étape est identique au PI. Il s'agit de la mise à jour du fichier de progression de manière à signifier à la POM la fin de l'exécution du modèle : statut à 0, avancement à 100% (avec éventuellement un message de fin d'exécution).