# Plan de transition : Migration des scripts Control-M vers Zabbix et standardisation Python

## Objectif

Rationaliser la surveillance des systèmes informatiques par la réécriture des scripts shell/Python en Python standardisé et la migration de leur exécution de Control-M vers Zabbix.

## Phase 1 : Préparation et Analyse (2 à 4 semaines)

### 1.1 Inventaire des scripts existants

* Collecter tous les scripts shell et Python utilisés dans Control-M.
* Identifier leur fréquence, dépendances, criticité, et propriétaires.
* Catégoriser par priorité métier (critique, important, secondaire).

### 1.2 Analyse technique

* Évaluer qualité du code, gestion d’erreurs, sécurité.
* Identifier patterns communs et dépendances externes.

### 1.3 Normes de codage Python

* Définir un framework standard : structure de script, logging, erreurs.
* Gérer configurations via fichiers standardisés (YAML, JSON).

## Phase 2 : Conception du Plan de Migration (2 semaines)

### 2.1 Mapping Control-M → Zabbix

* Cartographier les jobs vers équivalents Zabbix (items, triggers, actions).
* Définir templates Zabbix adaptés aux besoins.

### 2.2 Planification par lots

* Lot 1 : scripts critiques simples.
* Lot 2 : scripts complexes.
* Lot 3 : scripts secondaires.

## Phase 3 : Révision fonctionnelle (1 à 2 semaines)

* Valider la pertinence de chaque script avec les équipes concernées.
* Supprimer ou fusionner les scripts obsolètes ou redondants.
* Documenter entrées/sorties et logique attendue.

## Phase 4 : Réécriture des Scripts en Python (4 à 12 semaines)

### 4.1 Développement

* Réécriture conforme au framework Python.
* Intégration de journaux adaptés à Zabbix.

### 4.2 Revue de code

* Contrôle pair à pair ou via outils (GitLab CI, SonarQube).
* Tests unitaires et validation des normes.

## Phase 5 : Intégration dans Zabbix (3 à 6 semaines)

### 5.1 Déploiement des scripts

* Uniformisation des emplacements et droits d’exécution.

### 5.2 Configuration Zabbix

* Création d’items personnalisés.
* Définition de triggers et actions.

### 5.3 Automatisation

* Templates et déploiement via API ou Ansible.

## Phase 6 : Testing et Recette (2 à 4 semaines)

### 6.1 Tests techniques

* Tests de non-régression, performance, et résilience.

### 6.2 Tests Zabbix

* Validation des alertes, escalades, et notifications.

### 6.3 Recette fonctionnelle

* Validation par les équipes métier, ajustements.

## Phase 7 : Déploiement en Production et Suivi (2 semaines + monitoring)

### 7.1 Déploiement progressif

* Déploiement par lots avec rollback prévu.
* Documentation détaillée.

### 7.2 Suivi

* Dashboard spécifique de suivi Zabbix.
* Corrections, réunion post-mortem.

## Livrables

* Inventaire complet et priorisé.
* Normes Python et framework.
* Scripts réécrits documentés.
* Templates Zabbix.
* Plans de tests et recettes.
* Guide d’exploitation post-migration.