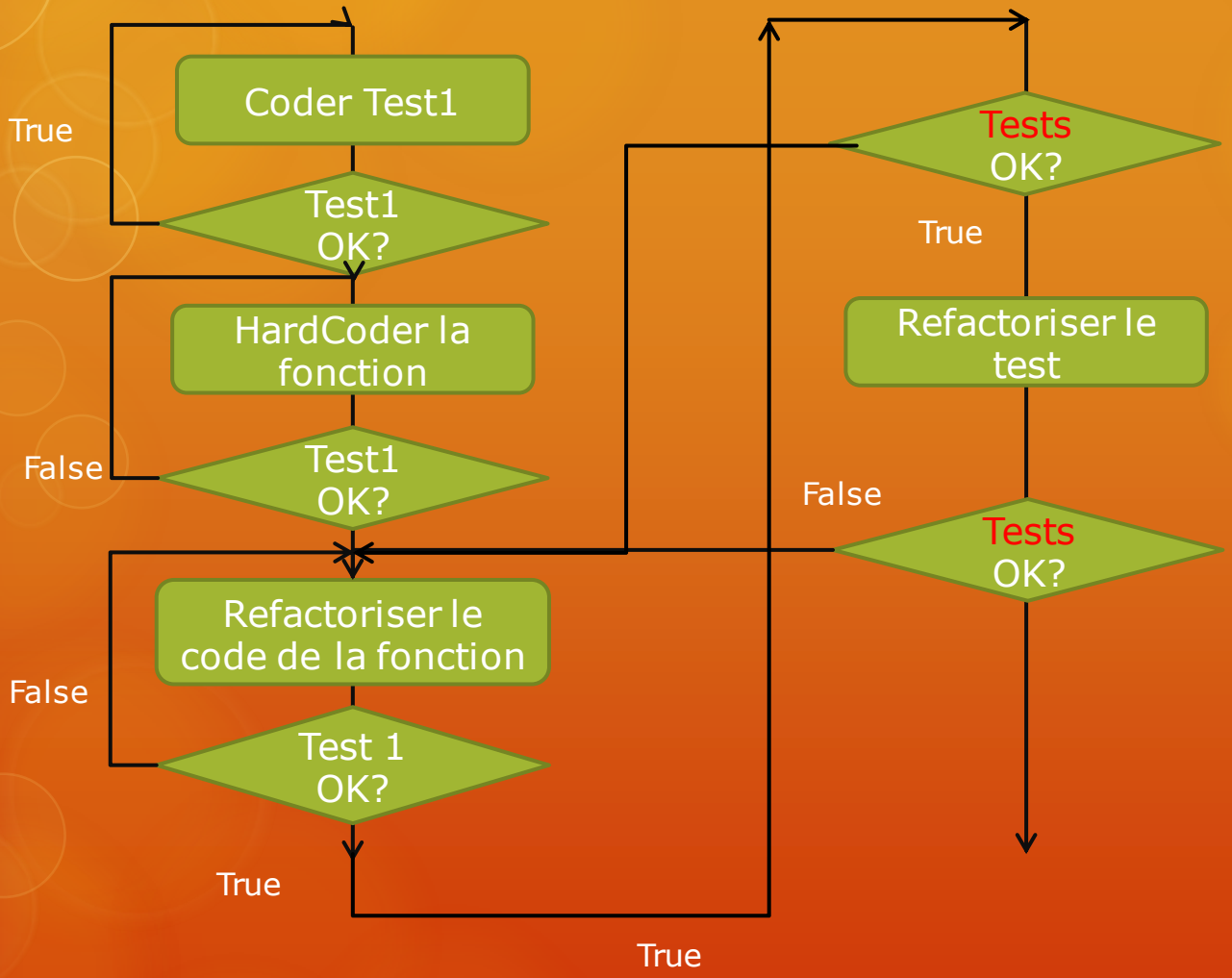
Test Driven Development

- XP a été mise au point à la fin des années 90 par **Kent Beck**, **Ward Cunningham** et **Ron Jeffries** mais elle nait officiellement en 1999 à travers l'ouvrage *Extreme Programming Explained* écrit par Kent Beck. C'est une méthode agile de gestion de projet informatique adaptée aux équipes réduites avec des besoins changeants. Elle doit son nom au fait qu'elle place l'activité de programmation au centre du projet, et qu'elle obtient ses résultats en combinant et en poussant à l'extrême certaines pratiques de développement.

TDD

- Méthode de développement qui place les tests au cœur du processus de développement
- Le code n'est écrit que pour faire passer un test
  - => pas de test =>Pas de code !!!!
- Le test doit être écrit avant le code



TDD

- Ecrire le code du test , vérifier que le test échoue
- Ecrire le code de la fonction, vérifier que le test réussit
- Refactoriser le code de la fonction, vérifier que le test réussit
- Refactoriser le code du test, vérifier que le test réussit

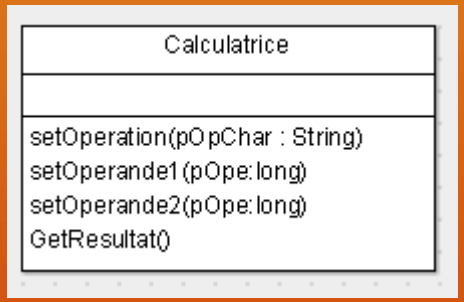
- Utilisation d'un framework de TU
  - Exécuter rapidement les tests unitaires
  - Chaque modif de code => Execution d'un test

TDD : Les 2 approches

- Top –Down
  - Test le besoin fonctionnel d'abord , puis son implémentation
  - => utilisation des bouchons, puis des classes effectives
- Botton-up
  - Tester la fonction de base, puis son utilisation

TDD :exemple

- Classe Calculatrice
- Saisie des opérandes :Setoperande1, SetOperande2
- Affichage du résultat (Par défaut 0)
  - Operande1 + Operande2 (Classe Addition)



TDD : Exemple Approche Top-Down

- Cas de l'addition
- Besoin fonctionnel
  - Saisie des opérandes 1 et 2
  - Affichage du résultat
- Structures des classes Calculatrice , Operation
  - Méthodes par défaut

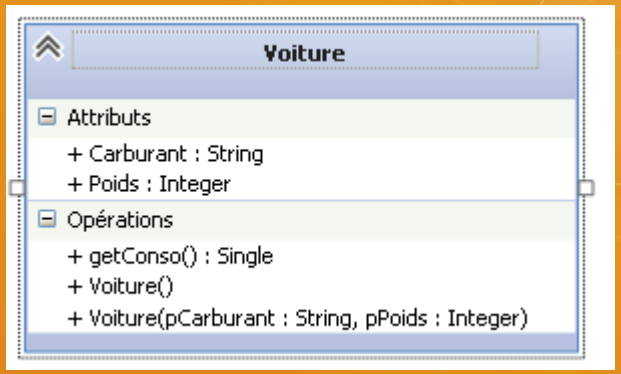
- Test de l'Addition (Pas de niveau operation)
- S0 : implémentation du test fonctionnel
- S3 : GetResultat dans Addition
  - => utilisation d'un Mock
  - => remplacement du mock par l'object réel
  - => Test avec un autre jeu d'essai

- Approche bottom-up
- Test de la soustraction
- Implémentation d'une classe soustraction
  - Test
- Intégration de cette classe dans Calculatrice
  - Test soustraction + test Addition => operation
- Refactoring de code : Ajout d'une classe operation
  - Vérification des tests unitaires précédent
- Nettoyage et Paramétrage des tests unitaires
  - Vérification des tests unitaires précédent

Exercice TDD

- Ajout de nouvelles fonctionnalités à la calculatrice
  - Operation * /
    - Création d'une fabrique d'operation
  - Operation SQRT (code Q)
    - Retour « »/ «ERREUR»
  - Operation PGCD Code (G)
  - Operation PPCM Code (P)
  - Ajout fonction Calculatrice.save()
    - Insert dans une BDD
  - Ajout fonction Calculatrice.Load(ID)
    - Lecture dans la BDD

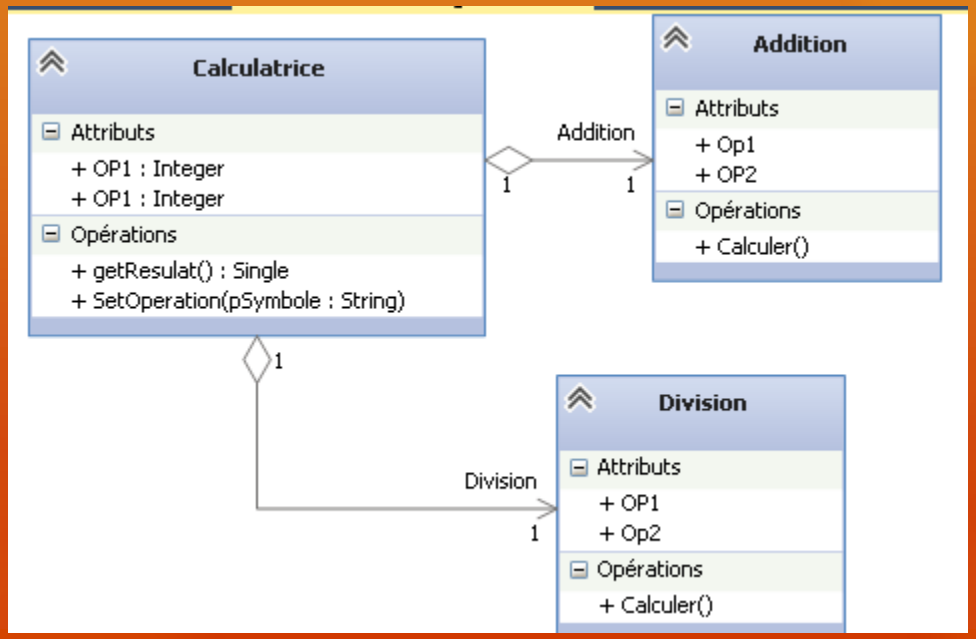
Exercice TDD 2



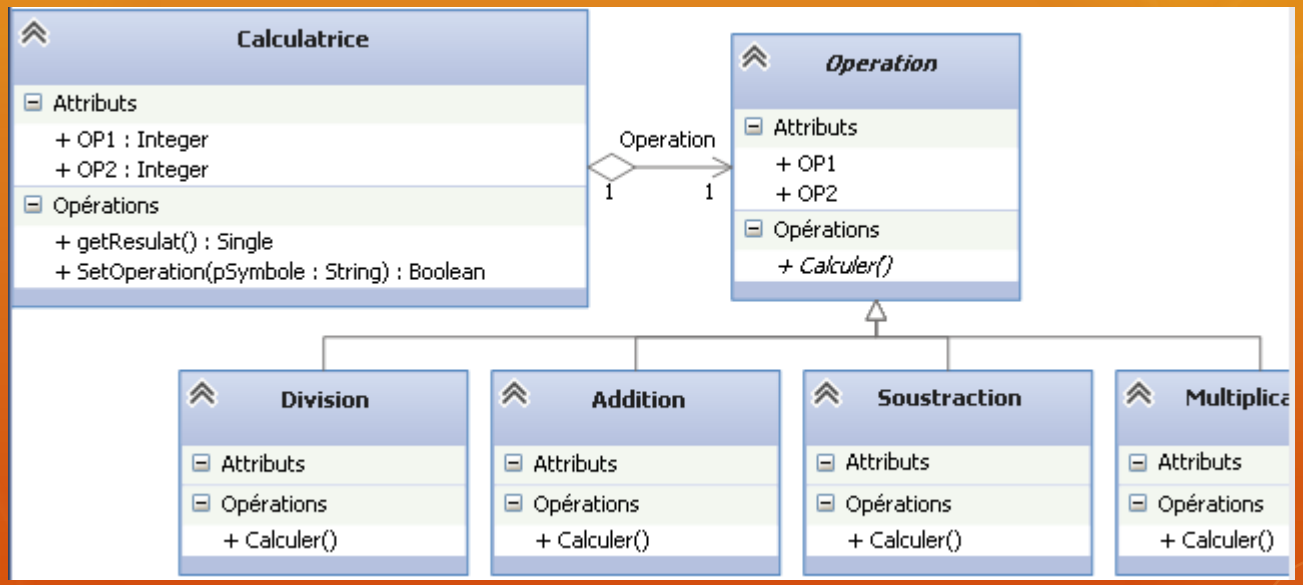
- Classe voiture
  - Attribut String Carburant (« E », « D », « G »)
    - ArgumentException si valeur différente
  - Attribut Int32 Poids
  - Methode getConso() -> Single
    - Essence (SI Poids < 1500; 5.5;7.2)
    - Diesel (SI Poids < 1700; 4.3;6.5)
    - GPL (SI Poids < 1800; 7.1;8.3)

TDD :exemple

- Classe Calculatrice
- Saisie des opérandes :Setoperande1, SetOperande2
- Affichage du résultat (Par défaut 0)
  - Operande1 + Operande2 (Classe Addition)

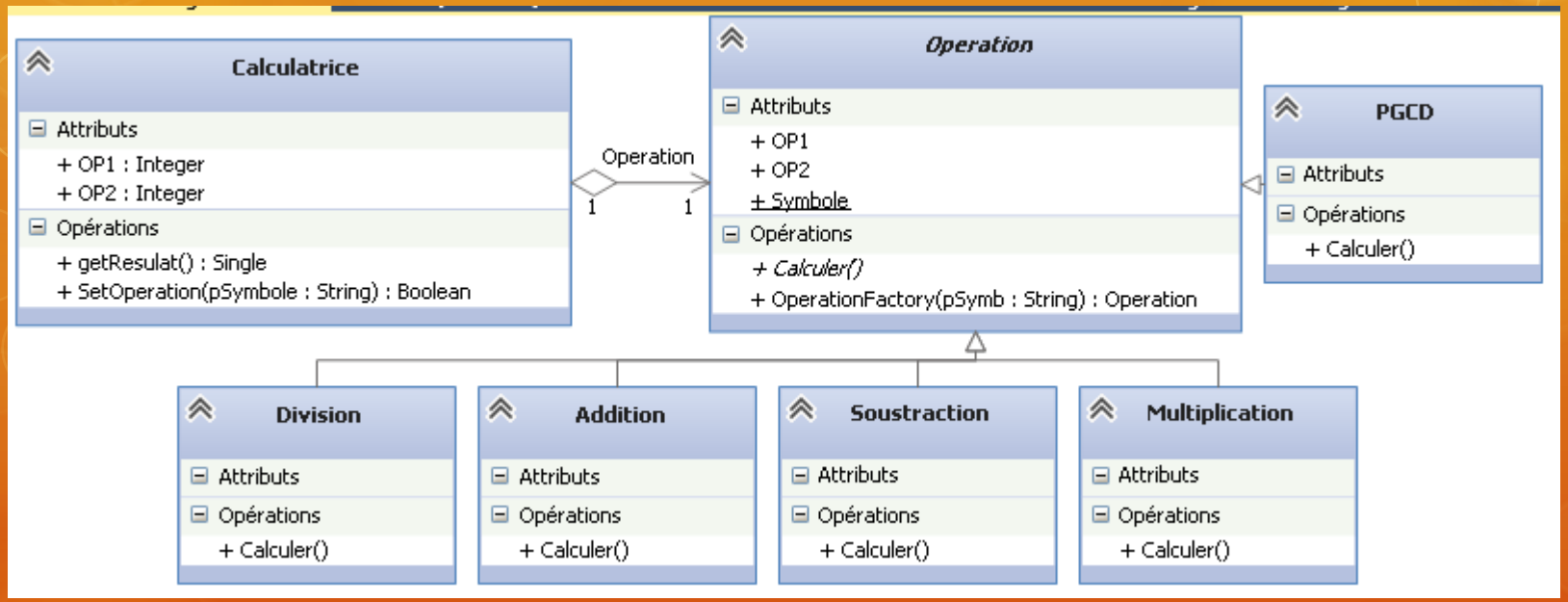


Calculatrice P2



Introduction d'une classe mère abstraite Opération
Suppression de l'attribut privé Symbole

Calculatrice P3



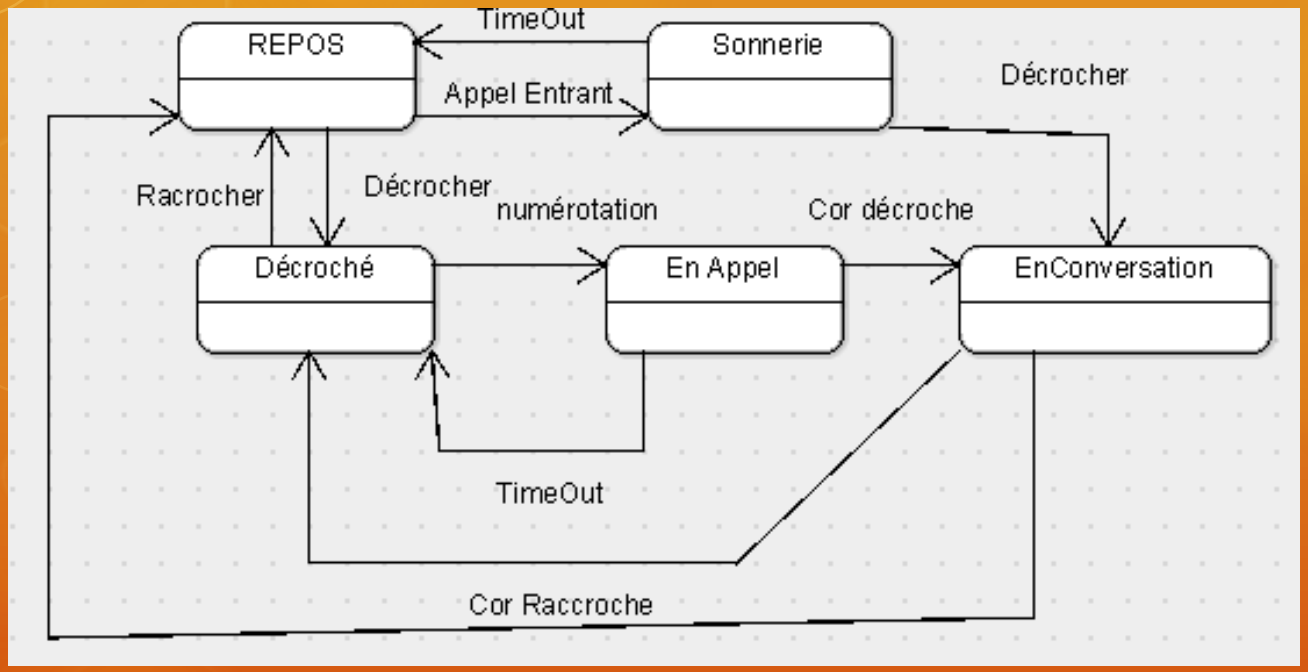
Introduction d'une nouvelle opération PGCD (symbole P)
Ajout d'une Operation Factory

Gestion des médias d'une bib(1)

- Un média a plusieurs états:
  - Lorsqu'on le créé il est nouveau
  - Un média nouveau peut être rendu disponible ou a Supprimer
  - Un média disponible peut être prêté
    - Lors de son retour il sera remis en disponible ou « à réparer »
  - Un Média « à réparer » peut être réparer pour être rendu disponible ou rejeter pour être déclaré « A supprimer »
  - Un média « Prêté » peut être déclaré perdu il sera alors déclaré « A supprimer »
  - Les médias « A supprimer » supprimer de la base par l'opération clean()

Gestion des médias d'une bib(2)

- Ajout de l'opération A Réparer et Réparer
  - Lors de son retour il sera remis en disponible ou « à réparer »
  - Un Média « à réparer » peut être réparer pour être rendu disponible ou rejeter pour être déclaré « A supprimer »



Phone
Etat : Integer
raccrocher() dechrocher() numeroter() decrocherCorres() raccrocherCorres() TimeOut()

DP Etat MCII