

Manipulation de modèles comportementaux pour les lignes de produits

Encadrant : Mr. Ziadi Tewfik

Présenté par:
Racha Ahmad
Oussama El Abed

Plan

- 1ère Partie : Présentation du projet
 - I. Introduction
 - II. Approche LDP
 - III. Objectif du Projet
- 2ème PARTIE : Etude environnemental du projet
 - I. choix des outils
 - II. Analyse et réalisation
 - III. Les composants du plugin
- Conclusion
- Démonstration

Introduction

- **Logiciel efficace**
 - Extensible, Changeable
 - configurable pour une utilisation dans un contexte particulier
- Variabilité logicielle
 - Plusieurs versions de la même application
- une solution industrielle
 - Transposition du développement industriel au logiciel

Introduction

- **Logiciel efficace**
 - Extensible, Changeable
 - configurable pour une utilisation dans un contexte particulier
- **Variabilité logicielle**
 - Plusieurs versions de la même application
- **une solution industrielle**
 - Transposition du développement industriel au logiciel

Introduction



Dérivation

Notepad v1



Notepad v2



Notepad v3



Introduction

- **Logiciel efficace**
 - Extensible, Changeable
 - configurable pour une utilisation dans un contexte particulier
- **Variabilité logicielle**
 - Plusieurs versions de la même application
- **une solution industrielle**
 - Transposition du développement industriel au logiciel

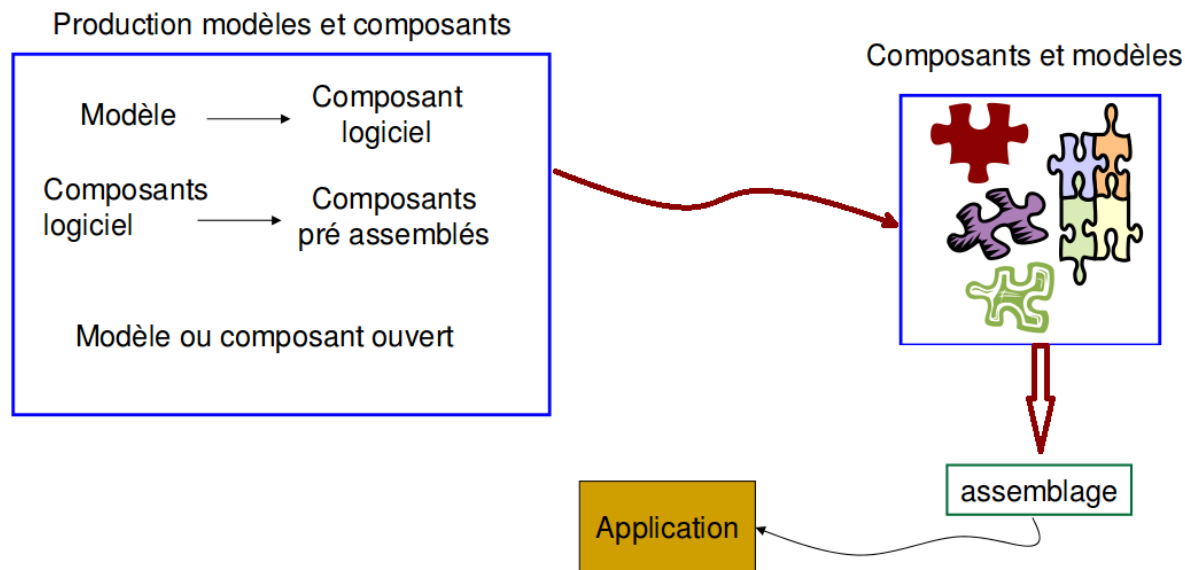
Approche LDP

- **Ligne de produits Logiciel (LdP)**
 - Ensemble de systèmes partageant un ensemble de propriétés communes et satisfaisant des besoins spécifiques pour un domaine particulier



Approche LDP

- Dimension 1 : Modélisation de la variabilité des Ldp.
- Dimension 2 : Dérivation automatique des produits.

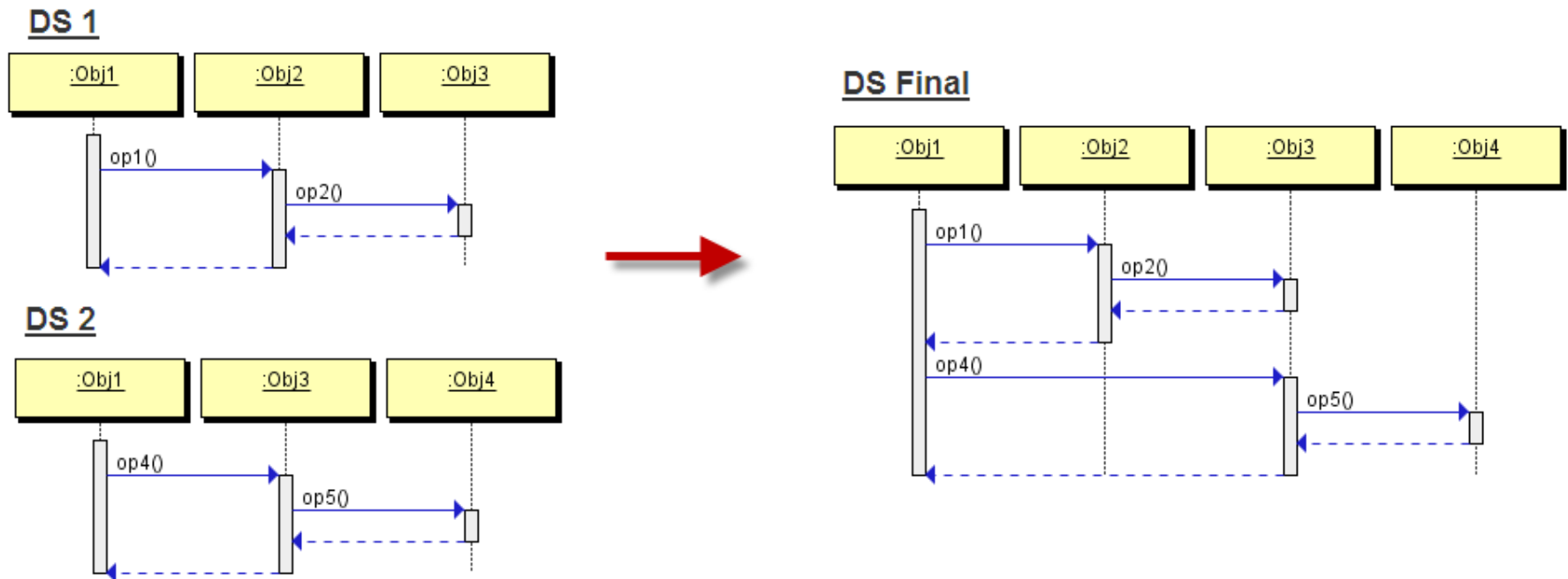


Objectif du Projet

- Appliquer l'approche LDP dans des modèle comportementaux où la variabilité spécifié est des diagrammes des séquences.
- Dérivation d'un diagramme de séquence final à partir des diagrammes de séquences des composants.
- Définir la combinaison relationnel entre ces composants

Objectif du Projet

- Appliquer l'approche LDP dans des modèle comportementaux où la variabilité spécifiée est des diagrammes des séquences.
- Dérivation d'un diagramme de séquence final à partir des diagrammes de séquences des composants.



Objectif du Projet

- Appliquer l'approche LDP dans des modèle comportementaux où la variabilité spécifié est des diagrammes des séquences.
- Dérivation d'un diagramme de séquence final à partir des diagrammes de séquences des composants.
- Définir la combinaison relationnel entre ces composants

Objectif du Projet

- Appliquer l'approche LDP dans des modèle comportementaux où la variabilité spécifiée est des diagrammes des séquences.
- Dérivation d'un diagramme de séquence final à partir des diagrammes de séquences des composants.
- Définir la combinaison relationnel entre ces composants
 - Exemple : la relation entre F1,F2,F3,F4

F1

après

F2

Si condition1 alors F3

sinon F4

Choix des outils

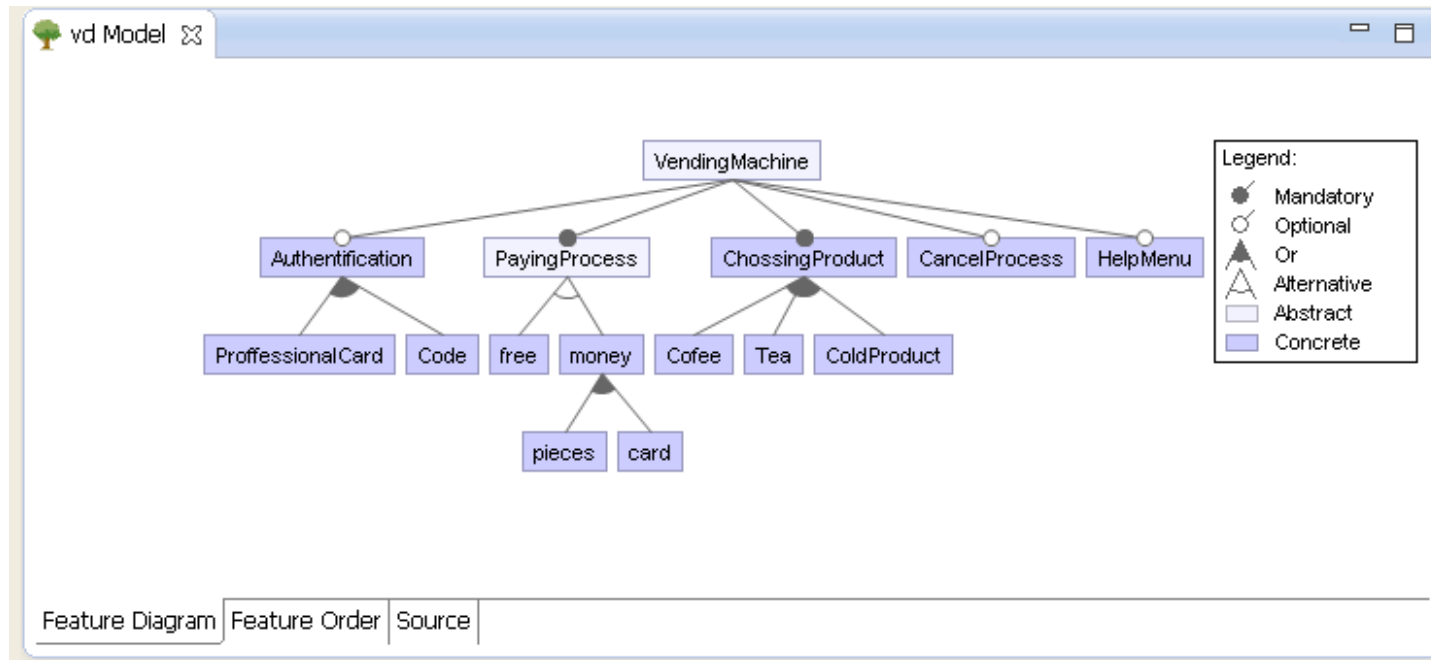
- **FeatureIDE**
- Xtext
- Sdedit

Choix des outils : FeatureIDE



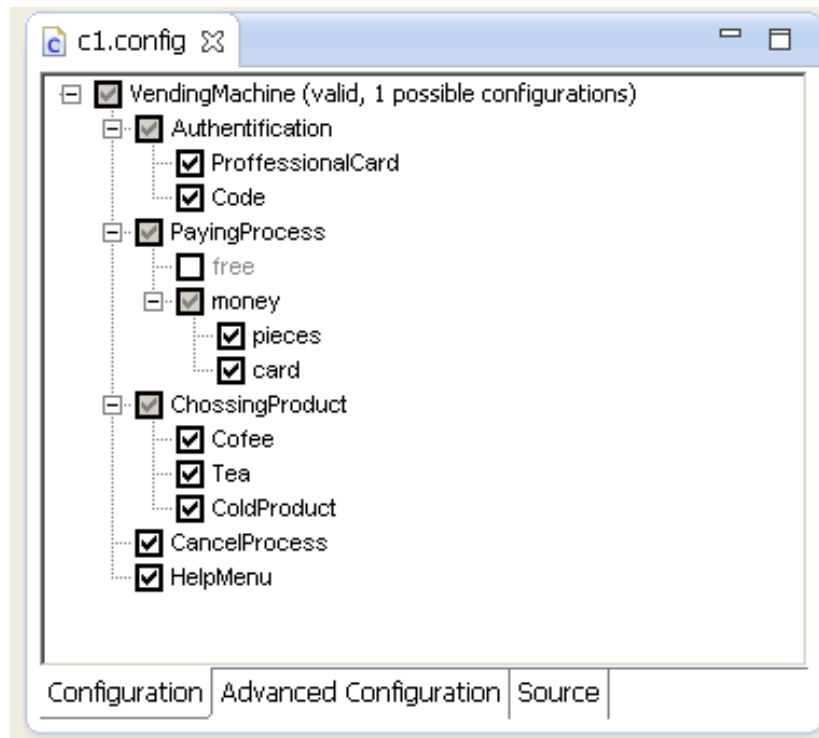
- FeatureIDE
 - IDE basé sur Eclipse
 - « feature-oriented »
 - développement des LDP: analyse de domaine, la mise en œuvre de domaine,
- « Feature » est une caractéristique d'un logiciel définie par les experts de domaine importante pour distinguer les différents produits.

Choix des outils : FeatureIDE



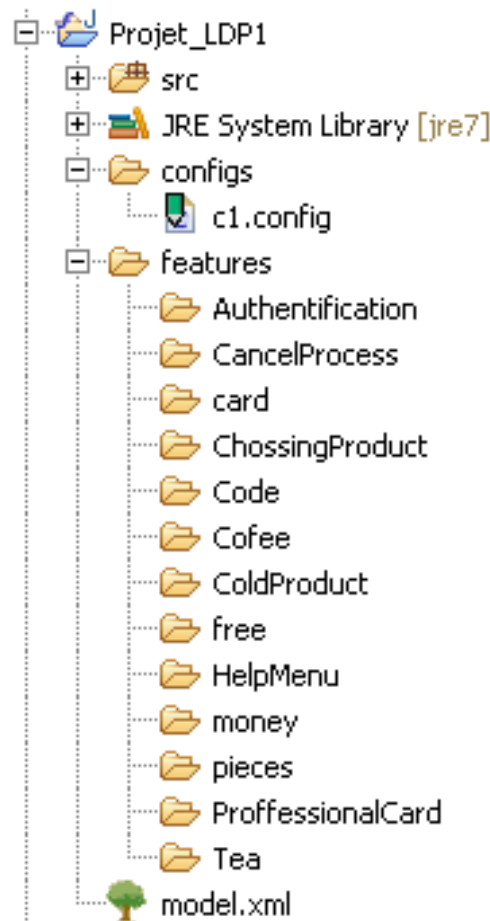
Editeur de Feature Modèle

Choix des outils : FeatureIDE



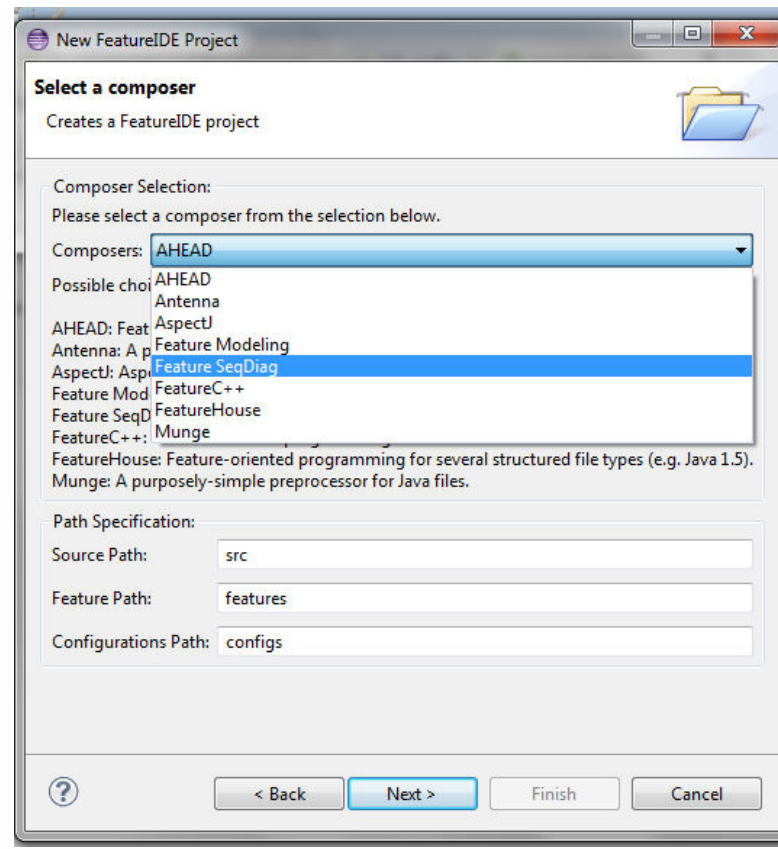
Le fichier de configuration

Choix des outils : FeatureIDE



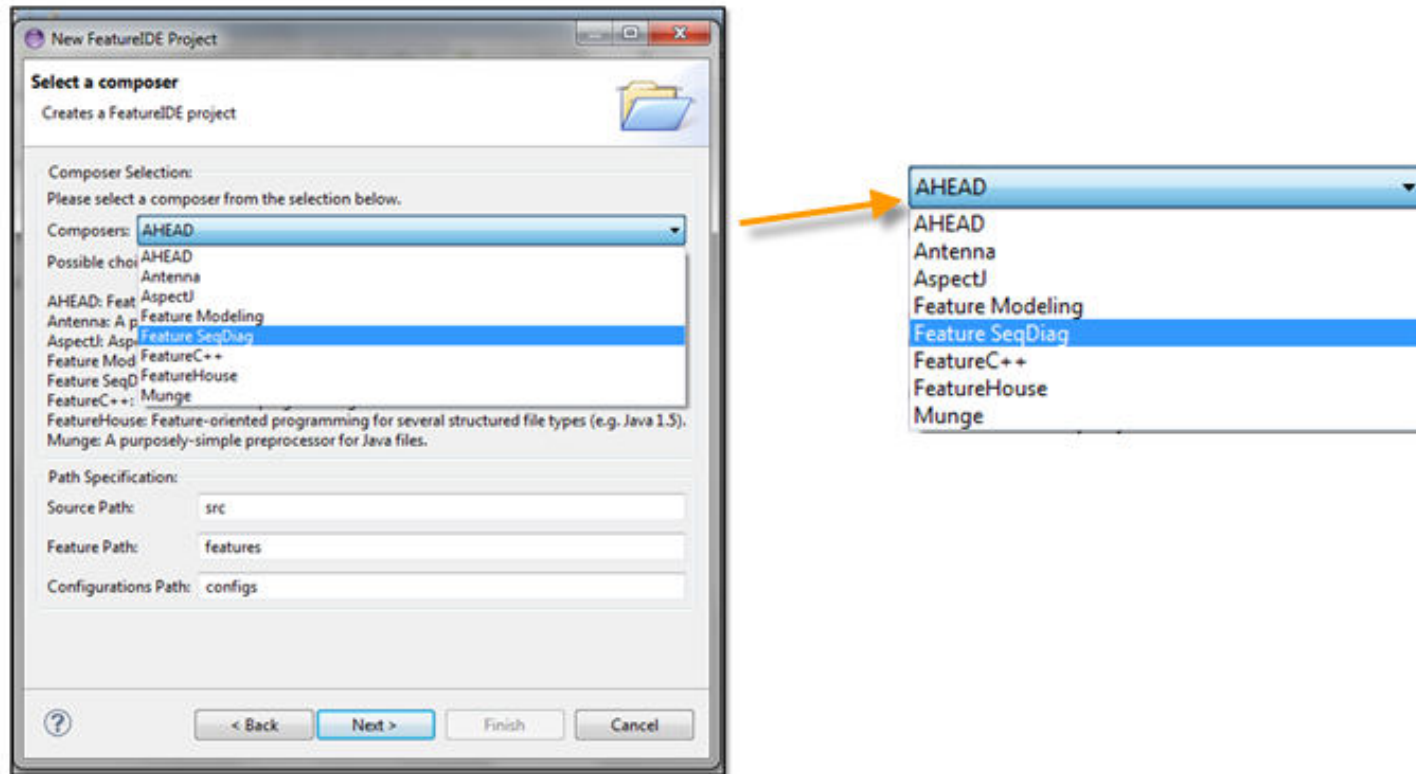
L'arborescence du projet

Choix des outils : FeatureIDE



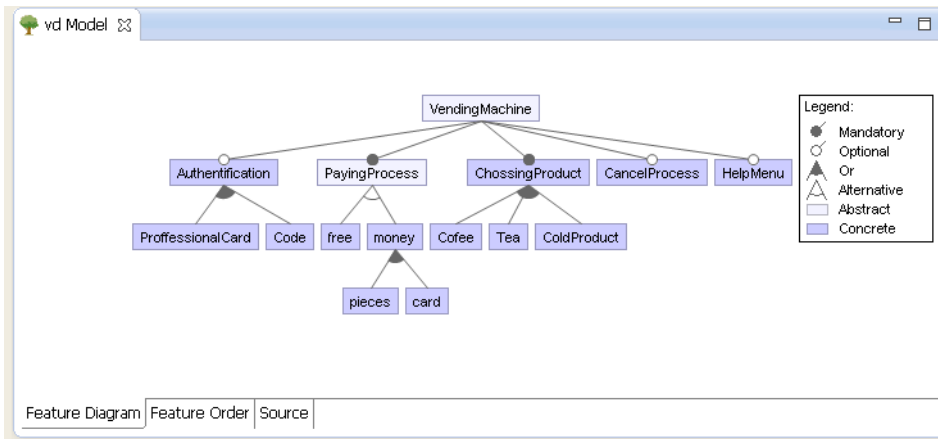
Menu du choix du composer du FeatureIDE

Choix des outils : FeatureIDE

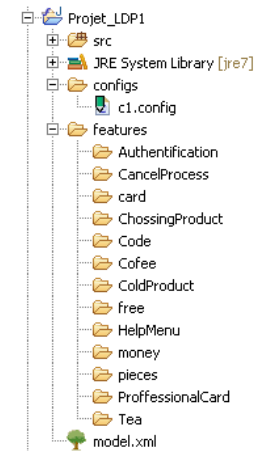


Menu du choix du composer du FeatureIDE

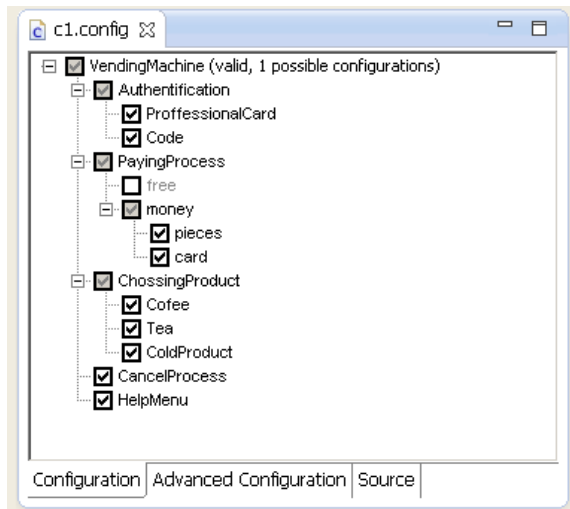
Choix des outils : FeatureIDE



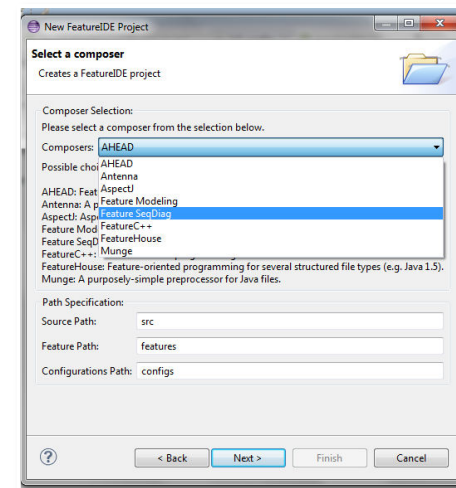
Editeur de Feature Modèle



L'arborescence du projet



Le fichier de configuration



choix du composer du FeatureIDE

Choix des outils

- **FeatureIDE**
- **Xtext**
- Sdedit

Choix des outils : Xtext

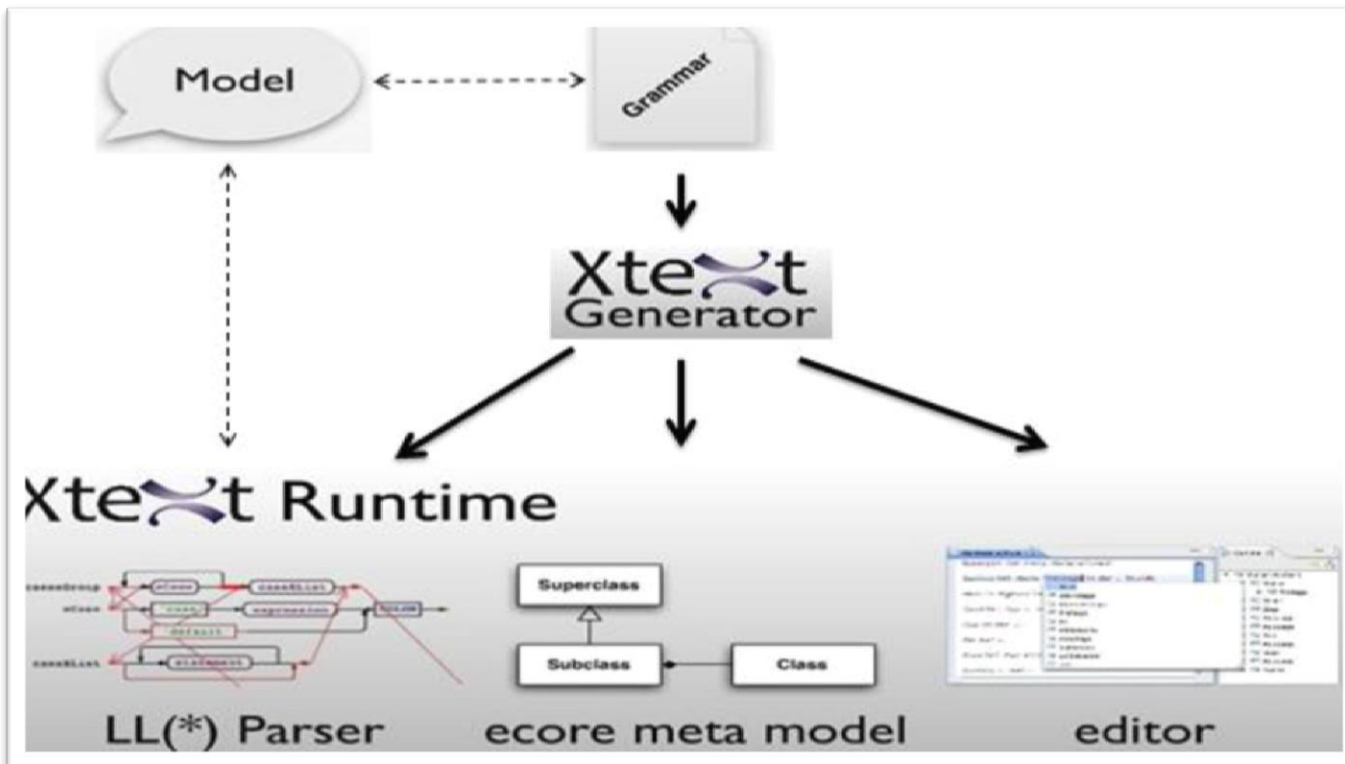


- une composante de TMF
- intégré dans Eclipse Modeling Framework : EMF

Choix des outils : Xtext

Xtext

- une composante de TMF
- intégré dans Eclipse Modeling Framework : EMF



Choix des outils

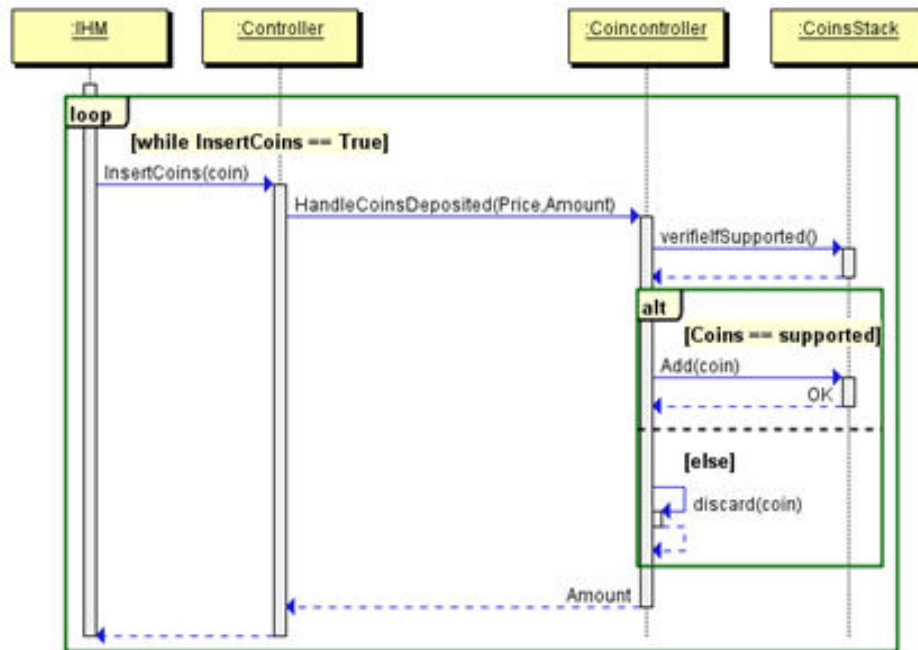
- **FeatureIDE**
- **Xtext**
- **Sdedit**

Choix des outils : Sdedit

- création de diagrammes de séquence UML
- Syntaxe simple, très utile

Choix des outils : Sdedit

- création de diagrammes de séquence UML
- Syntaxe simple, très utile



```
ihm: IHM[a]
controller: Controller[a]
coinController: Coincontroller[a]
coinsStack: CoinsStack[a]

[c:loop while InsertCoins == True ]
    ihm:controller.InsertCoins(coin)
    controller:Amount=coinController.HandleCoinsDeposited(Price,Amount)
    coinController:coinsStack.verifyIfSupported()

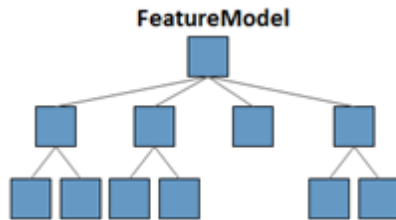
    [c:alt Coins == supported]
        coinController:OK=coinsStack.Add(coin)
        --[else]
        coinController:coinController.discard(coin)
    [/c]

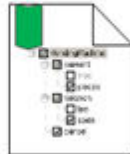
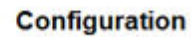
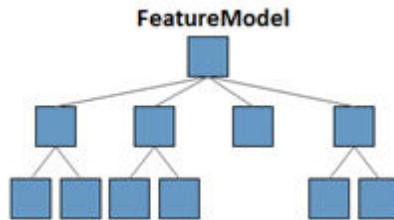
[/c]
```

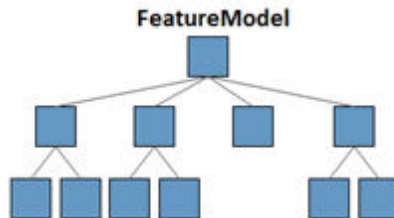
Analyse et Réalisation

- Étendre l'environnement FeatureIDE par un plugin
 - se spécialise dans la dérivation des diagrammes de séquence
 - **le composer FeatureSeqDiag**
 - Éditer les fichier de combinaison
 - **Le langage Sdcombin**
 - Éditer les fichier de l'application Sdedit
 - **Le langage Sdedit**
 - Visualiser les diagramme de séquence dérivé
 - **L'application sdedit-4.01**

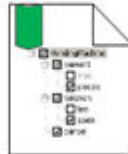
Analyse et Réalisation



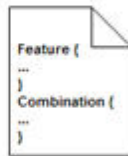




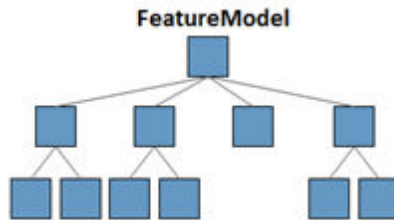
Configuration



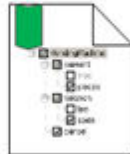
SysCombin.sdc



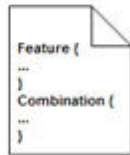
Analyse et Réalisation



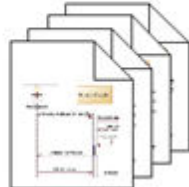
Configuration



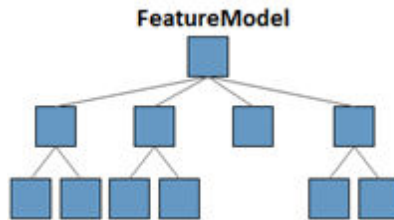
SysCombin.sdc



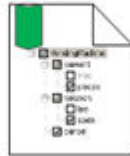
**Features Sedit
Files**



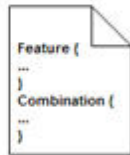
Analyse et Réalisation



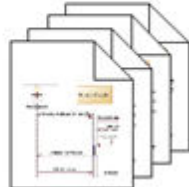
Configuration



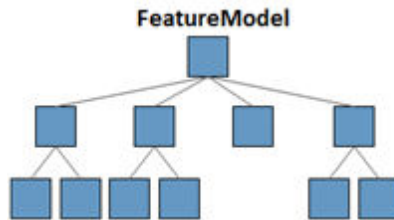
SysCombin.sdc



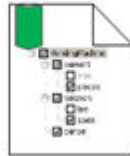
**Features Sedit
Files**



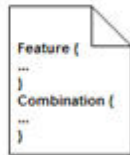
Analyse et Réalisation



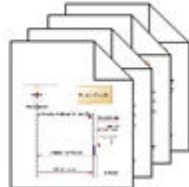
Configuration



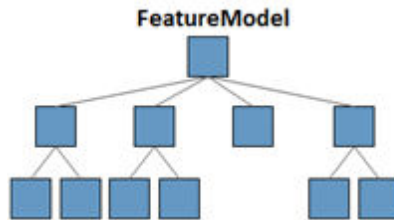
SysCombin.sdc



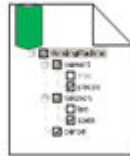
**Features Sedit
Files**



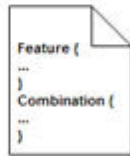
Analyse et Réalisation



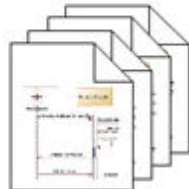
Configuration



SysCombin.sdc



Features Sdedit Files



SDGenerated Directory

Final SysCombin.sdc



Final Sdedit File

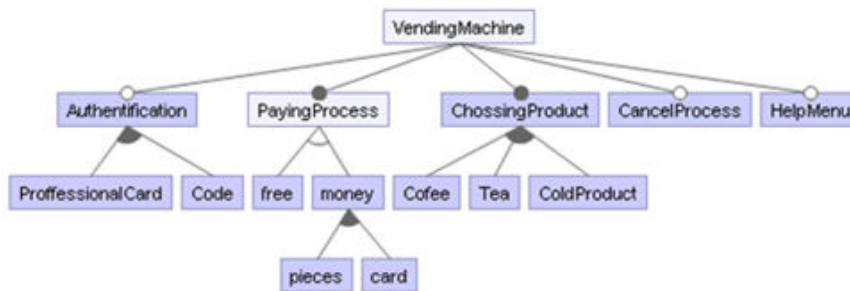


Analyse et Réalisation: SysCombin.sdc

- Le fichier sdcombin « SysCombin.sdc » contient
 - La déclaration des features interagit
 - la combinaison entre les features père et/ou feature feuille
 - Feature père : qui a des fils ou des autres père de fils
 - Feature feuille : les feuille de l'arbre Feature Model(FM)
- Après la dérivation, le nouveau fichier sdcombin doit contenir la combinaison entre que les feature fils

Analyse et Réalisation: SysCombin.sdc

Implémentation :



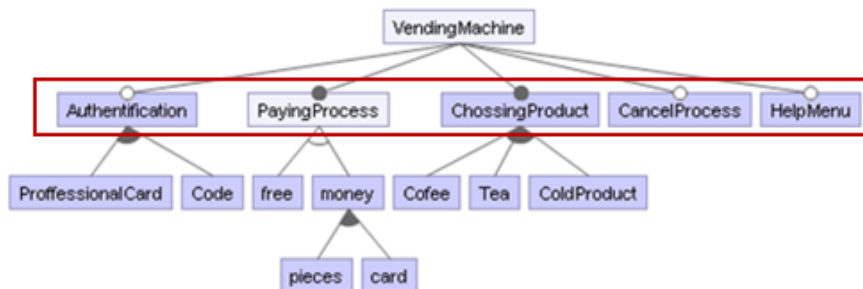
SysCombin.sdc

```
Features {
    Authentication
    Help
    PayingProcess
    CancelProcess
    ChoosingProduct
}

Combination {
    Authentication
    loop [" while UserDesire == TRUE "] {
        opt ["Help IS True"] {
            Help
        }
        PayingProcess
        alt ["CancelProcess == True"] {
            CancelProcess
        }
        else
        ChoosingProduct
    }
}
```

Analyse et Réalisation: sysCombin.sdc

Implémentation :



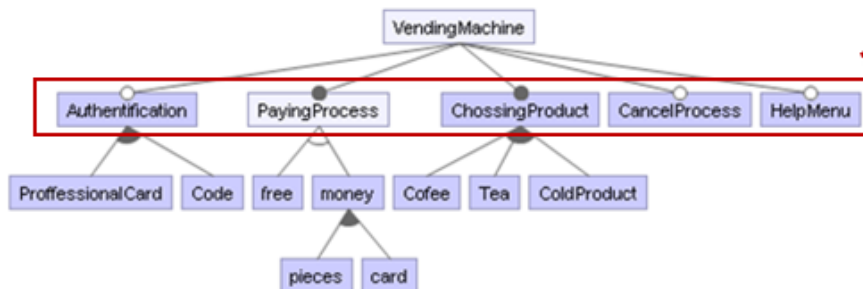
SysCombin.sdc

```
Features {
    Authentication
    Help
    PayingProcess
    CancelProcess
    ChoosingProduct
}

Combination {
    Authentication
    loop [" while UserDesire == TRUE "] {
        opt ["Help IS True"] {
            Help
        }
        PayingProcess
        alt ["CancelProcess == True"] {
            CancelProcess
        }
        else
        ChoosingProduct
    }
}
```

Analyse et Réalisation: sysCombin.sdc

Implémentation :



SysCombin.sdc

Features {

```
Authentication
Help
PayingProcess
CancelProcess
ChoosingProduct
```

}

Combination {

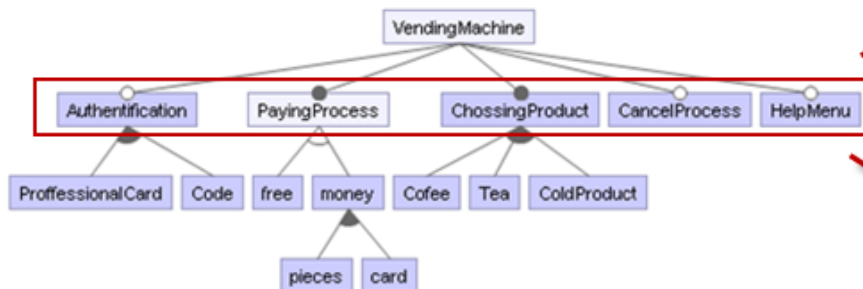
```
Authentication
loop [" while UserDesire == TRUE "] {
  opt ["Help IS True"] {
    Help
  }
  PayingProcess
  alt ["CancelProcess == True"] {
    CancelProcess
  }
  else
  ChoosingProduct
}
}
```

}

Analyse et Réalisation: SysCombin.sdc

Implémentation :

SysCombin.sdc



Features {

```
Authentication
Help
PayingProcess
CancelProcess
ChoosingProduct
```

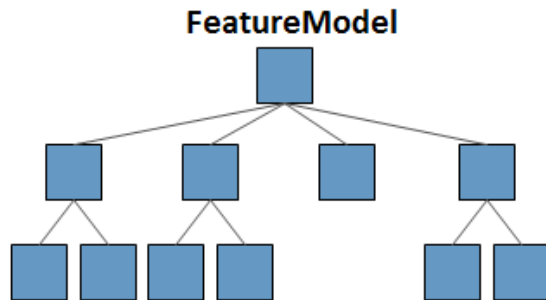
}

Combination {

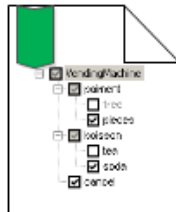
```
Authentication
loop [" while UserDesire == TRUE "] {
  opt ["Help IS True"] {
    Help
  }
  PayingProcess
  alt ["CancelProcess == True"] {
    CancelProcess
  } else
    ChoosingProduct
  }
}
```

}

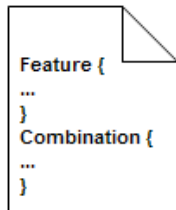
Analyse et Réalisation: Dérivation de la combinaison



Configuration



SysCombin.sdc



Dérivation

**Final
SysCombin.sdc**

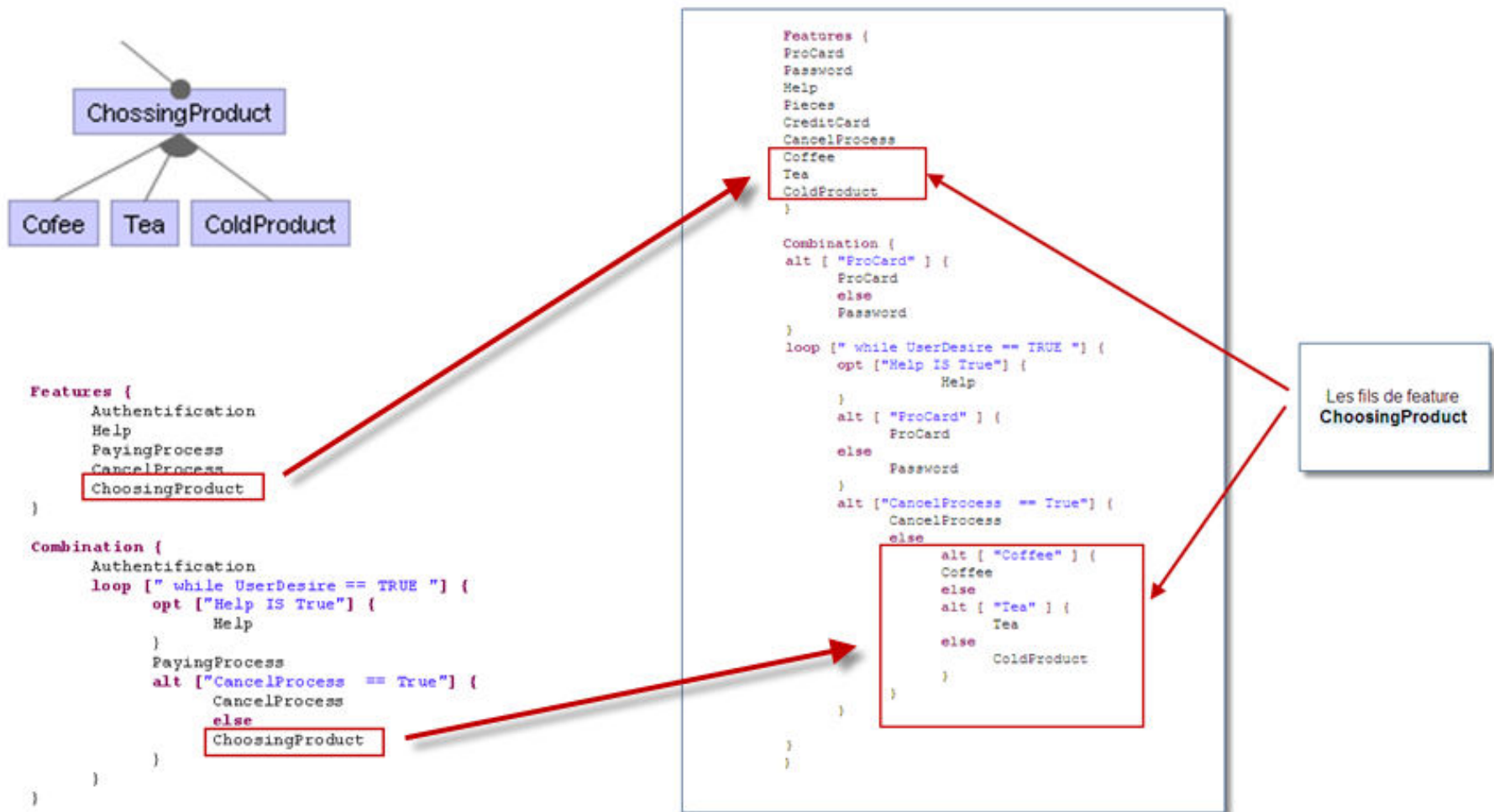
```
Feature {  
...  
}  
Combination {  
...  
}
```


Analyse et Réalisation: Config_NewSysCombin.sdc

Après la dérivation :

Analyse et Réalisation: Config_NewSysCombin.sdc

Après la dérivation :



Analyse et Réalisation: les fichiers Sdedit

- Le fichier sdedit « feature.sd » contient
 - le diagramme des séquence spécifié propre au feature
 - Une partie pour la déclaration des objets
 - Une autre partie pour définir les événements entre les objets
- Il faut le crée pour chaque feature son fichier
- Mettre le fichier dans le répertoire qui le même nom de feature dans le projet
 - si le feature abstract, il n'interagit pas dans les dérivations
 - Il n'a pas de dossier

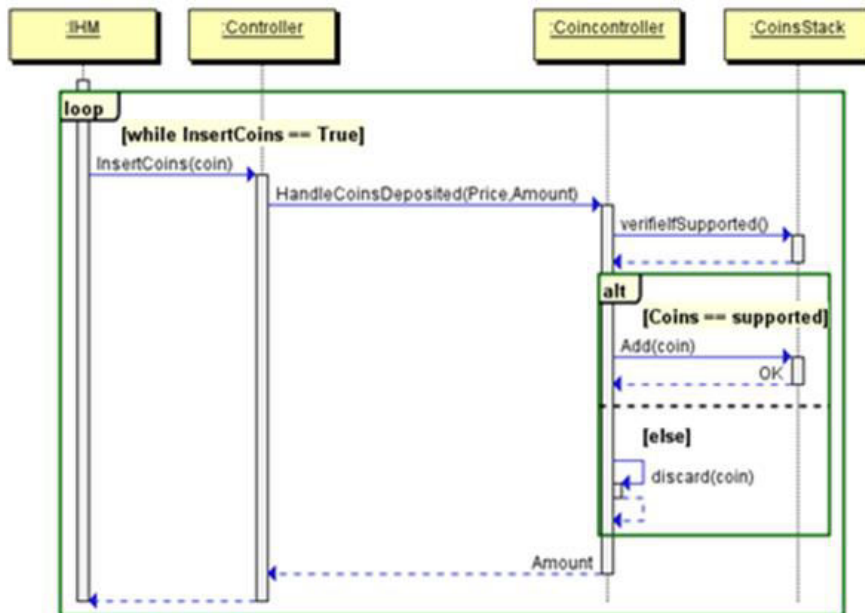
Analyse et Réalisation: les fichiers Sdedit

- Le fichier sdedit « feature.sd » contient
 - le diagramme des séquence spécifié propre au feature
 - Une partie pour la déclaration des objets
 - Une autre partie pour définir les événements entre les objets
- Il faut le crée pour chaque feature son fichier
- Mettre le fichier dans le répertoire qui le même nom de feature dans le projet
 - si le feature abstract, il n'interagit pas dans les dérivations
 - Il n'a pas de dossier

Analyse et Réalisation: Les fichiers Sdedit

- Le fichier sdedit « feature.sd » contient
 - le diagramme des séquence spécifié propre au feature
 - Une partie pour la déclaration des objets
 - Une autre partie pour définir les événements entre les objets
- Il faut le crée pour chaque feature son fichier
- Mettre le fichier dans le répertoire qui le même nom de feature dans le projet
 - si le feature abstract, il n'interagit pas dans les dérivations
 - Il n'a pas de dossier

Analyse et Réalisation: Feature.sd



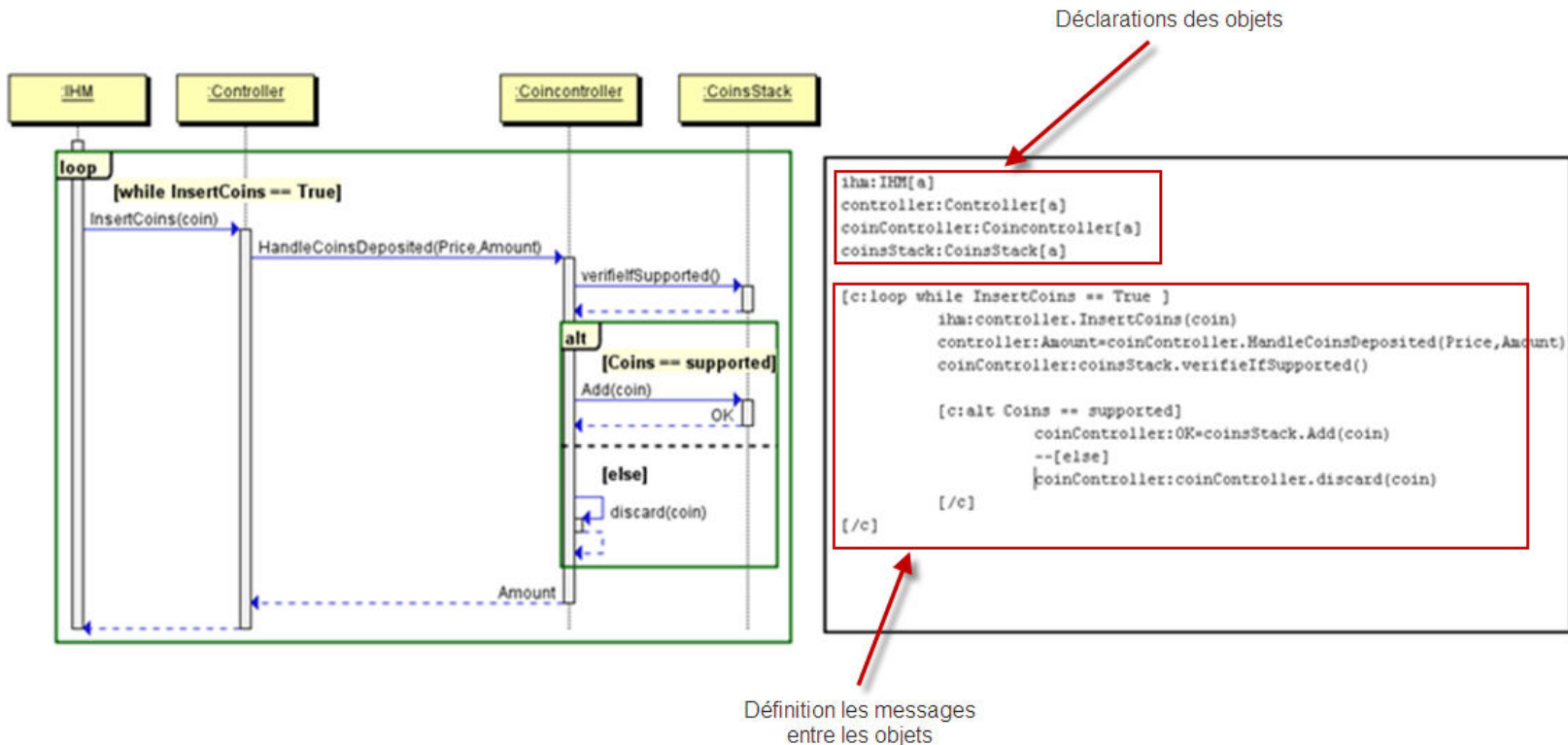
```
ihm:IHM[a]
controller:Controller[a]
coinController:Coincontroller[a]
coinsStack:CoinsStack[a]

[c:loop while InsertCoins == True ]
    ihm:controller.InsertCoins(coin)
    controller:Amount=coinController.HandleCoinsDeposited(Price,Amount)
    coinController:coinsStack.verifyIfSupported()

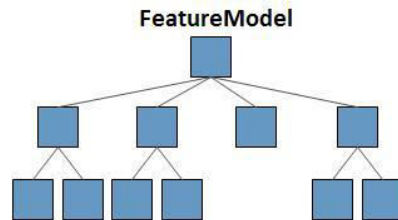
    [c:alt Coins == supported]
        coinController:OK=coinsStack.Add(coin)
        --[else]
        coinController:coinController.discard(coin)
    [/c]

[/c]
```

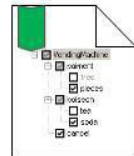
Analyse et Réalisation: Feature.sd



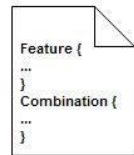
Analyse et Réalisation: Dérivation de la DS



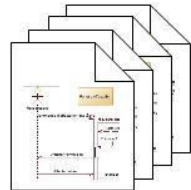
Configuration



SysCombin.sdc



**Features Sedit
Files**



Final Sedit File



Analyse et Réalisation: config_FinalSD.sd

- La partie déclaration des objets
 - rassemble toutes les objets déclarés dans toutes les fichier feature.sd
 - éviter la redondance des objets
- La partie définition des messages entre les objets
 - suivre la squelette fichier SysCombin.sdc
 - pour chaque feature interagit dans le fichier SysCombin.sdc on copie la 2ème partie du son fichier sdedit

Analyse et Réalisation: config_FinalSD.sd

Feature1.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]
```


```
obj1:obj2.message1()  
obj2:obj3.message2()
```

Feature2.sd

```
obj1:OBJ1[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

```
obj1:obj3.message3()  
obj3:obj4.message4()
```

Feature1 seq Feature2



FinalSD.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

```
obj1:obj2.message1()  
obj2:obj3.message2()  
obj1:obj3.message3()  
obj3:obj4.message4()
```

Analyse et Réalisation: config_FinalSD.sd

Feature1.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]
```

```
obj1:obj2.message1()  
obj2:obj3.message2()
```

Feature2.sd

```
obj1:OBJ1[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

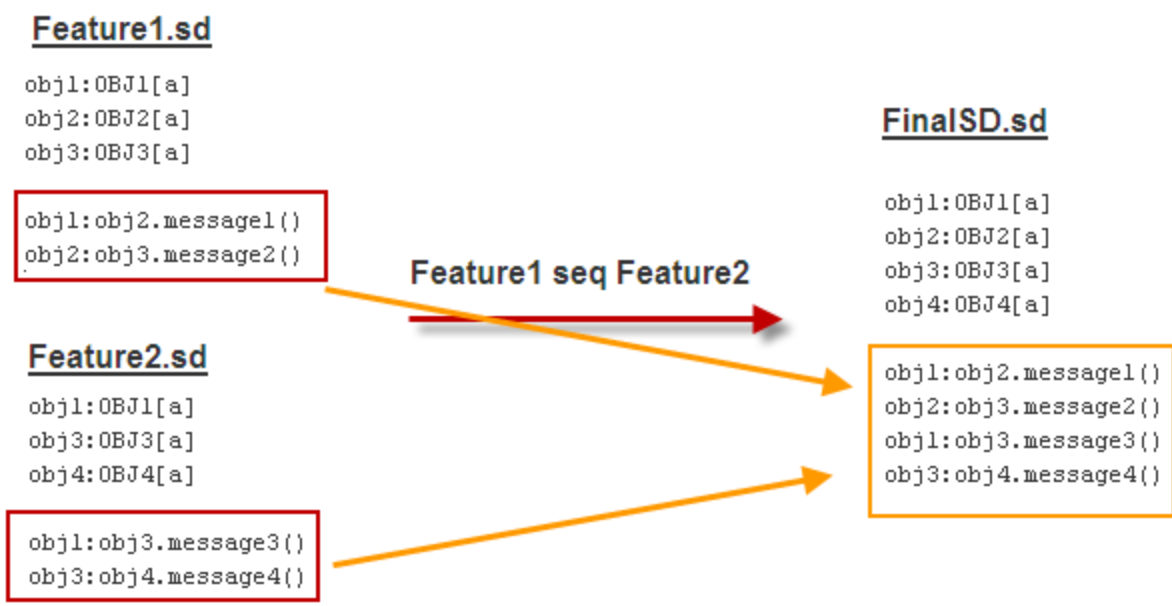
```
obj1:obj3.message3()  
obj3:obj4.message4()
```

Feature1 seq Feature2

FinalSD.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

```
obj1:obj2.message1()  
obj2:obj3.message2()  
obj1:obj3.message3()  
obj3:obj4.message4()
```



Analyse et Réalisation: `config_FinalSD.sd`

Feature1.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]
```

```
obj1:obj2.message1()  
obj2:obj3.message2()
```

Feature2.sd

```
obj1:OBJ1[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

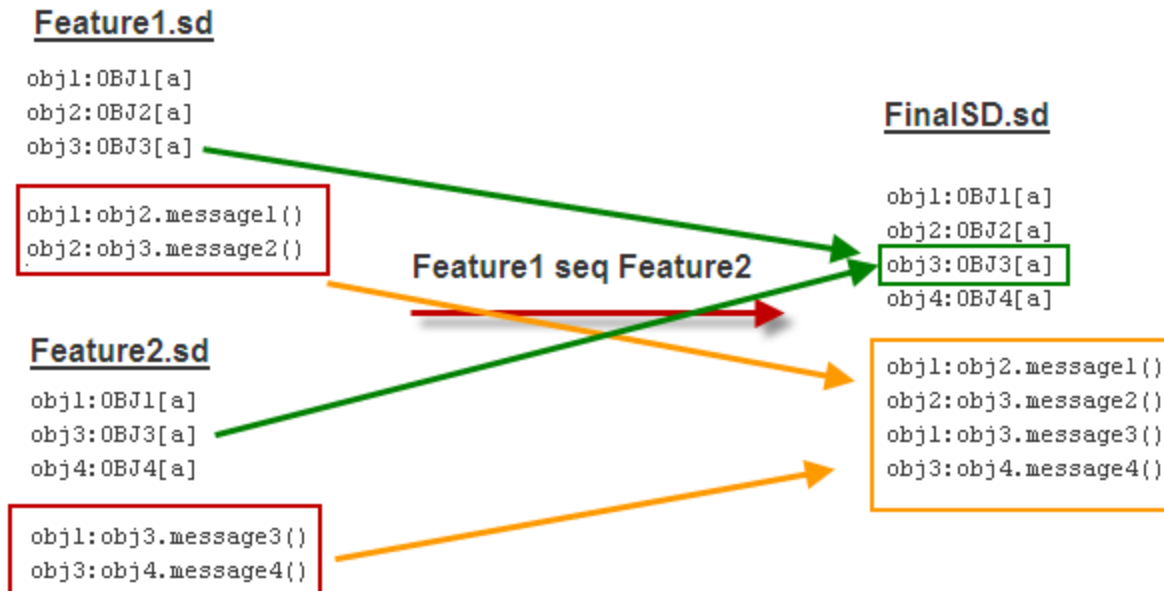
```
obj1:obj3.message3()  
obj3:obj4.message4()
```

Feature1 seq Feature2

FinalSD.sd

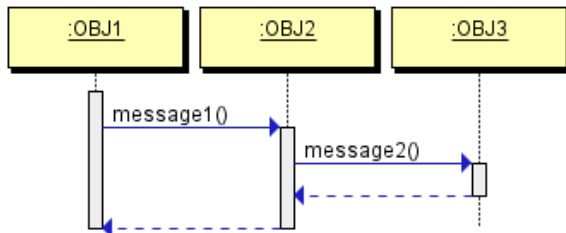
```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

```
obj1:obj2.message1()  
obj2:obj3.message2()  
obj1:obj3.message3()  
obj3:obj4.message4()
```

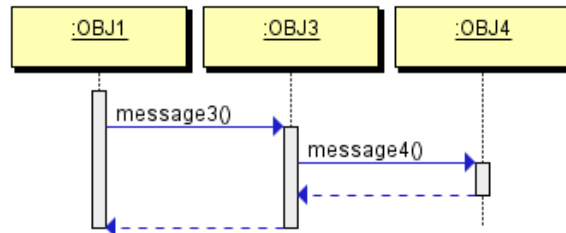


Analyse et Réalisation: config_FinalSD.sd

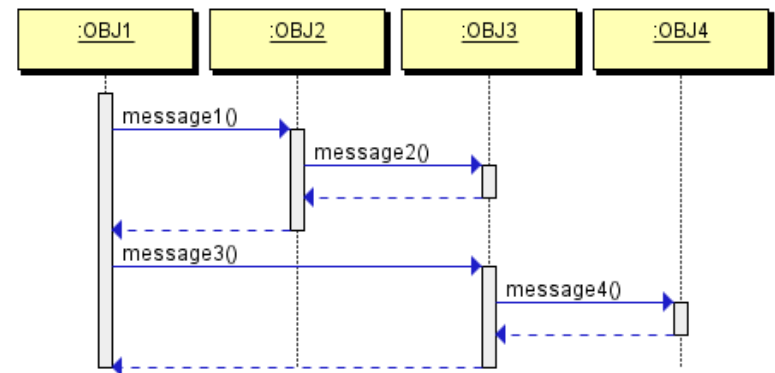
Feature1.sd



Feature2.sd



FinalSD.sd



Analyse et Réalisation: config_FinalSD.sd

Feature1.sd

```
obj1:OBJ1[a]
obj2:OBJ2[a]
obj3:OBJ3[a]

obj1:obj2.message1()
obj2:obj3.message2()
.
```

Feature2.sd

```
obj1:OBJ1[a]
obj3:OBJ3[a]
obj4:OBJ4[a]

obj1:obj3.message3()
obj3:obj4.message4()
```

alt ["condition"] { Feature1 else Feature2 }



FinalSD.sd

```
obj1:OBJ1[a]
obj2:OBJ2[a]
obj3:OBJ3[a]
obj4:OBJ4[a]

[c:alt condition]
    obj1:obj2.message1()
    obj2:obj3.message2()
--[else]
    obj1:obj3.message3()
    obj3:obj4.message4()

[/c]
```

Analyse et Réalisation: `config_FinalSD.sd`

Feature1.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]  
  
obj1:obj2.message1()  
obj2:obj3.message2()  
.
```

Feature2.sd

```
obj1:OBJ1[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]  
  
obj1:obj3.message3()  
obj3:obj4.message4()
```

alt ["condition"] { Feature1 else Feature2 }

FinalSD.sd

```
obj1:OBJ1[a]  
obj2:OBJ2[a]  
obj3:OBJ3[a]  
obj4:OBJ4[a]
```

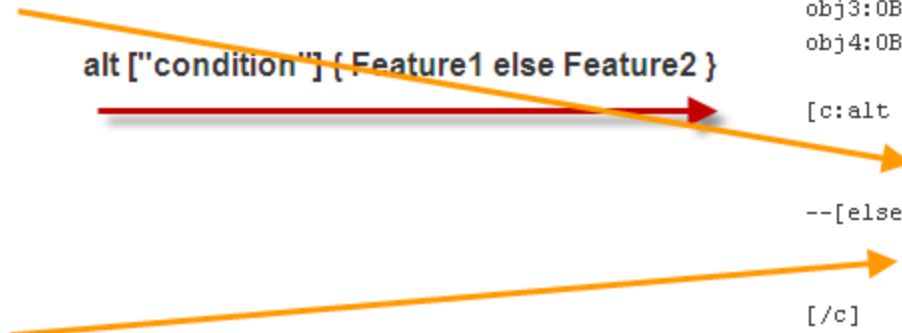
[c:alt condition]

```
obj1:obj2.message1()  
obj2:obj3.message2()
```

--[else]

```
obj1:obj3.message3()  
obj3:obj4.message4()
```

[/c]



Analyse et Réalisation: config_FinalSD.sd

Feature1.sd

```
obj1:OBJ1[a]
obj2:OBJ2[a]
obj3:OBJ3[a]

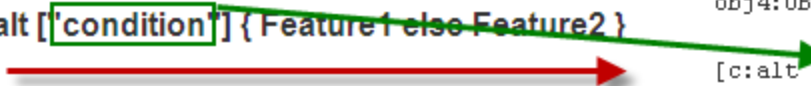
obj1:obj2.message1()
obj2:obj3.message2()
.
```

Feature2.sd

```
obj1:OBJ1[a]
obj3:OBJ3[a]
obj4:OBJ4[a]

obj1:obj3.message3()
obj3:obj4.message4()
```

alt ['condition'] { Feature1 else Feature2 }



FinalSD.sd

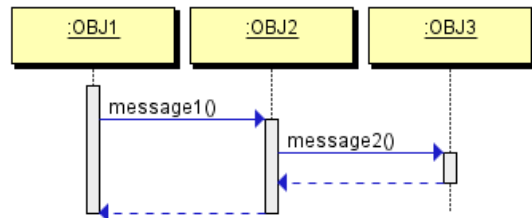
```
obj1:OBJ1[a]
obj2:OBJ2[a]
obj3:OBJ3[a]
obj4:OBJ4[a]

[c:alt condition]
    obj1:obj2.message1()
    obj2:obj3.message2()
--[else]
    obj1:obj3.message3()
    obj3:obj4.message4()

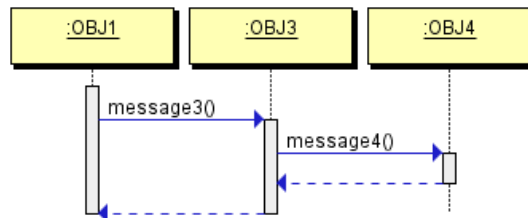
[/c]
```


Analyse et Réalisation: `config_FinalSD.sd`

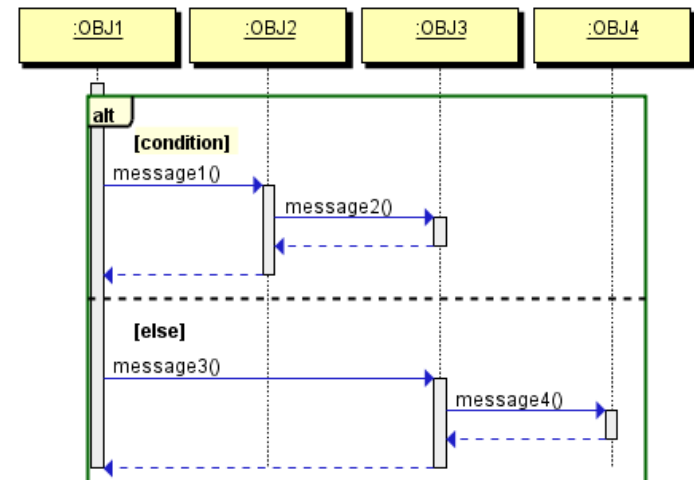
Feature1.sd



Feature2.sd



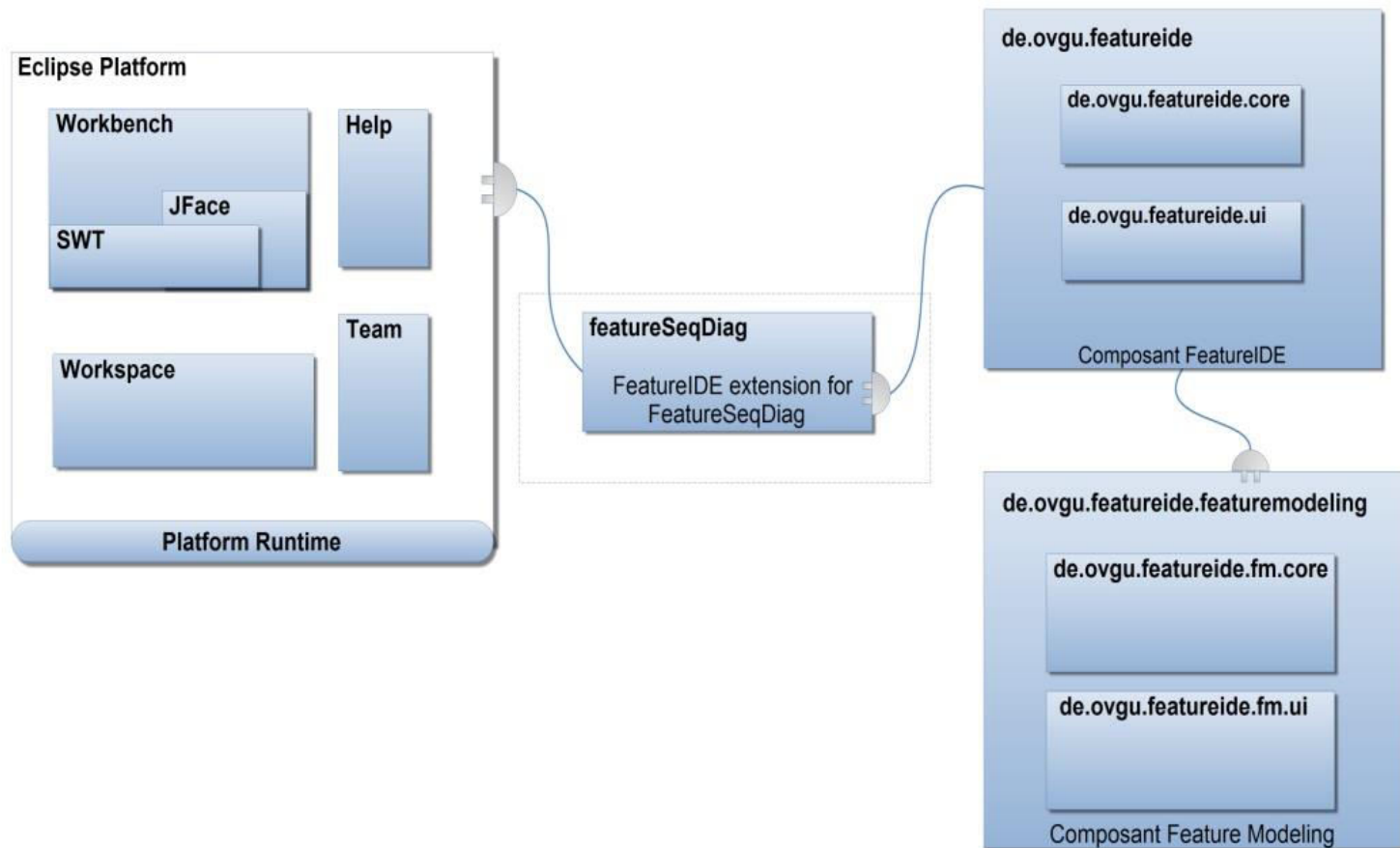
FinalSD.sd



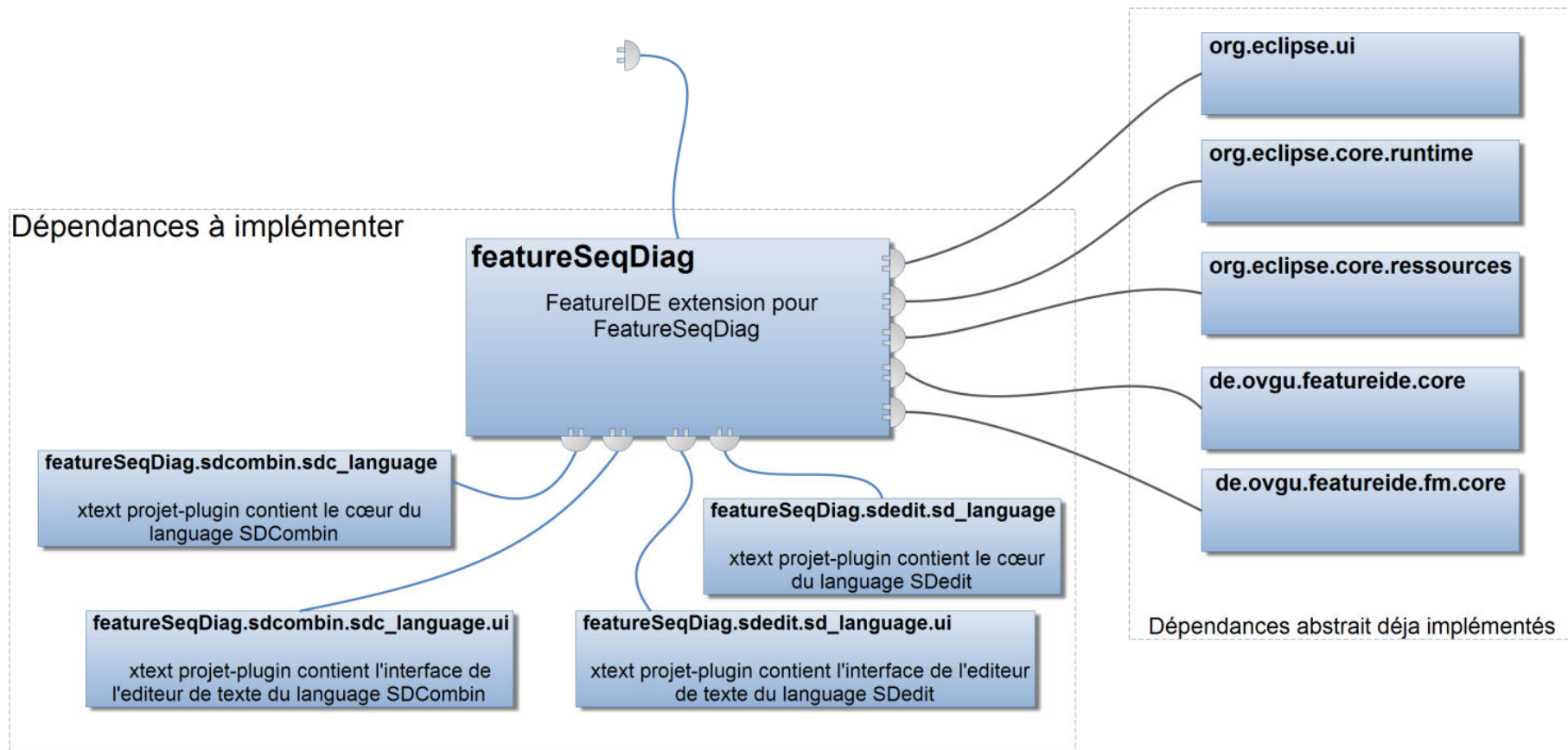
Les composants du plugin

- Le plugin qui se spécialise dans la dérivation des diagrammes de séquence
 - **FeatureSeqDiag**
- Éditeur des fichier de combinaison
 - **featureSeqDiag.sdcombin.sd_language**
 - **featureSeqDiag.sdcombin.sd_language.ui**
- Éditeur des fichier de l'application Sdedit
 - **featureSeqDiag.sdedit.sd_language**
 - **featureSeqDiag.sdedit.sd_language.ui**
- Visualiser les diagramme de séquence dérivé
 - **L'application sdedit-4.01**

Les composants du plugin



Les composants du plugin



Conclusion

- la modélisation et la gestion de la variabilité dans les systèmes à logiciel prépondérant , par exemple les lignes de produits logiciels est une tâche critique
- traiter les modèles de lignes de produits logiciels où la variabilité est spécifiée dans des modèles comportementaux (diagrammes de séquence).
- implémenter deux mini DSL (Domain Specific Language)
- proposer un algorithme de dérivation pour les MC qu'on a intégré dans l'environnement FeatureIDE.

Conclusion

- la modélisation et la gestion de la variabilité dans les systèmes à logiciel prépondérant , par exemple les lignes de produits logiciels est une tâche critique
- traiter les modèles de lignes de produits logiciels où la variabilité est spécifiée dans des modèles comportementaux (diagrammes de séquence).
- implémenter deux mini DSL (Domain Specific Language)
- proposer un algorithme de dérivation pour les MC qu'on a intégré dans l'environnement FeatureIDE.

Conclusion

- la modélisation et la gestion de la variabilité dans les systèmes à logiciel prépondérant , par exemple les lignes de produits logiciels est une tâche critique
- traiter les modèles de lignes de produits logiciels où la variabilité est spécifiée dans des modèles comportementaux (diagrammes de séquence).
- implémenter deux mini DSL (Domain Specific Language)
- proposer un algorithme de dérivation pour les MC qu'on a intégré dans l'environnement FeatureIDE.

Conclusion

- la modélisation et la gestion de la variabilité dans les systèmes à logiciel prépondérant , par exemple les lignes de produits logiciels est une tâche critique
- traiter les modèles de lignes de produits logiciels où la variabilité est spécifiée dans des modèles comportementaux (diagrammes de séquence).
- implémenter deux mini DSL (Domain Specific Language)
- proposer un algorithme de dérivation pour les MC qu'on a intégré dans l'environnement FeatureIDE.

Démonstration