


# LOG210 SÉANCE #02

## ANALYSE ET CONCEPTION DE LOGICIELS

1. Administration 
2. MDD: Catégories
3. DSS: Opérations système
4. RDCU: Contrôleur GRASP
5. Architecture en couches
6. Exercice

# ADMINISTRIVIA

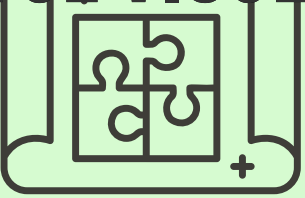
1. Accès G Suite ([ETSMTL.NET](https://ETSMTL.NET))
  - Google Classrooms
  - Google Drive
2. Invitation Lab 0 (GitHub Classroom)
  - ⚠ lien différent pour chaque groupe!
  - ne pas partager
3. Équipes (connaissez-vous vos coéquipiers?)
4. ~~Lien Zoom~~ Serveur Discord pour lab (Google Classrooms)



# ANALYSE ET CONCEPTION DE LOGICIELS

1. Administration
2. MDD: Catégories ← s20203
3. DSS: Opérations système
4. RDCU: Contrôleur GRASP
5. Architecture en couches
6. Exercice

# SURVOL VISUEL



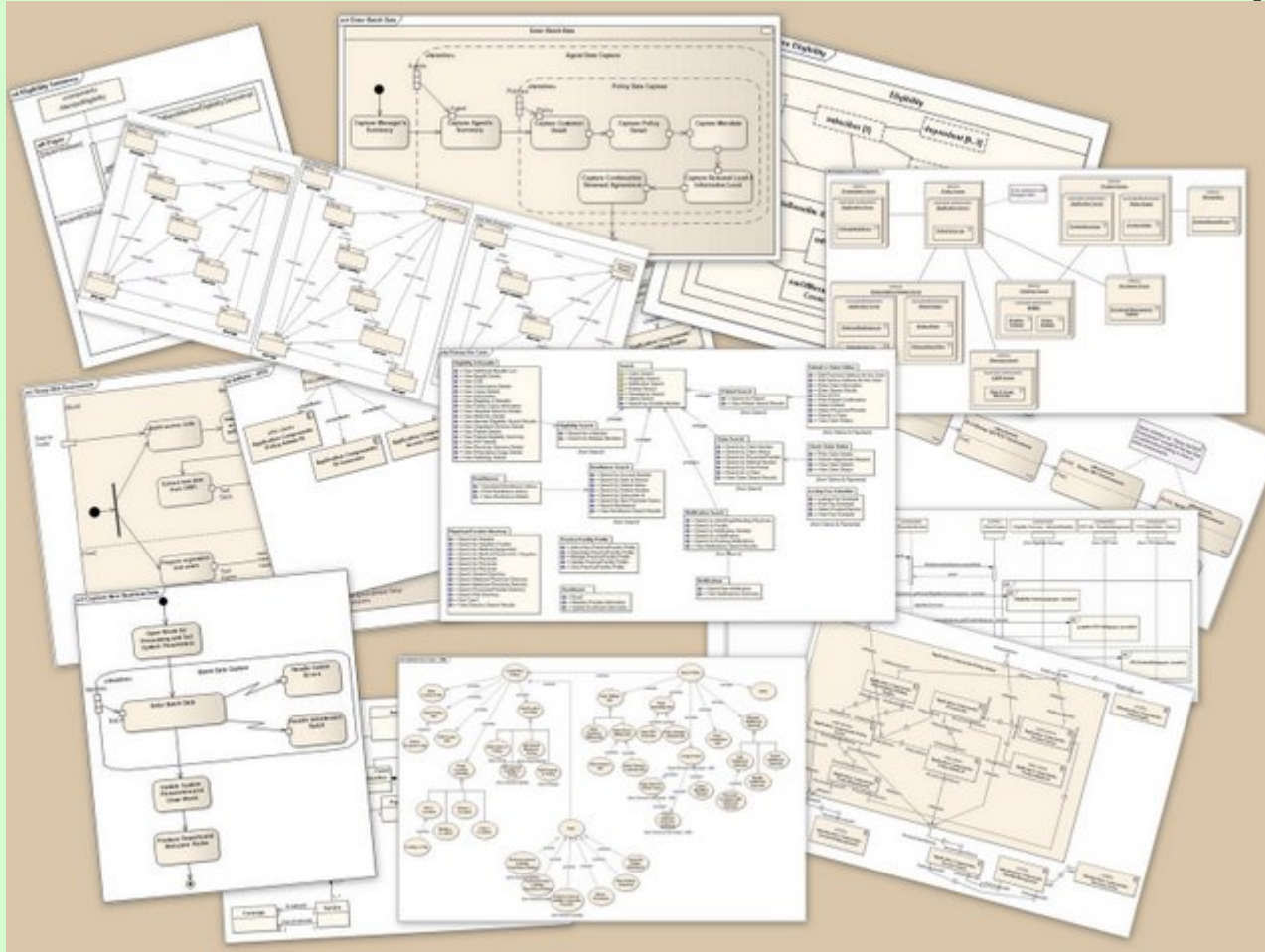
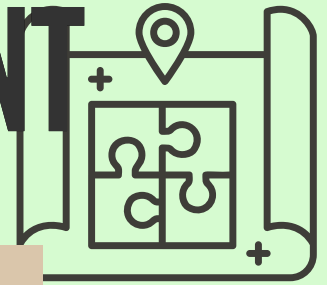
# MDD



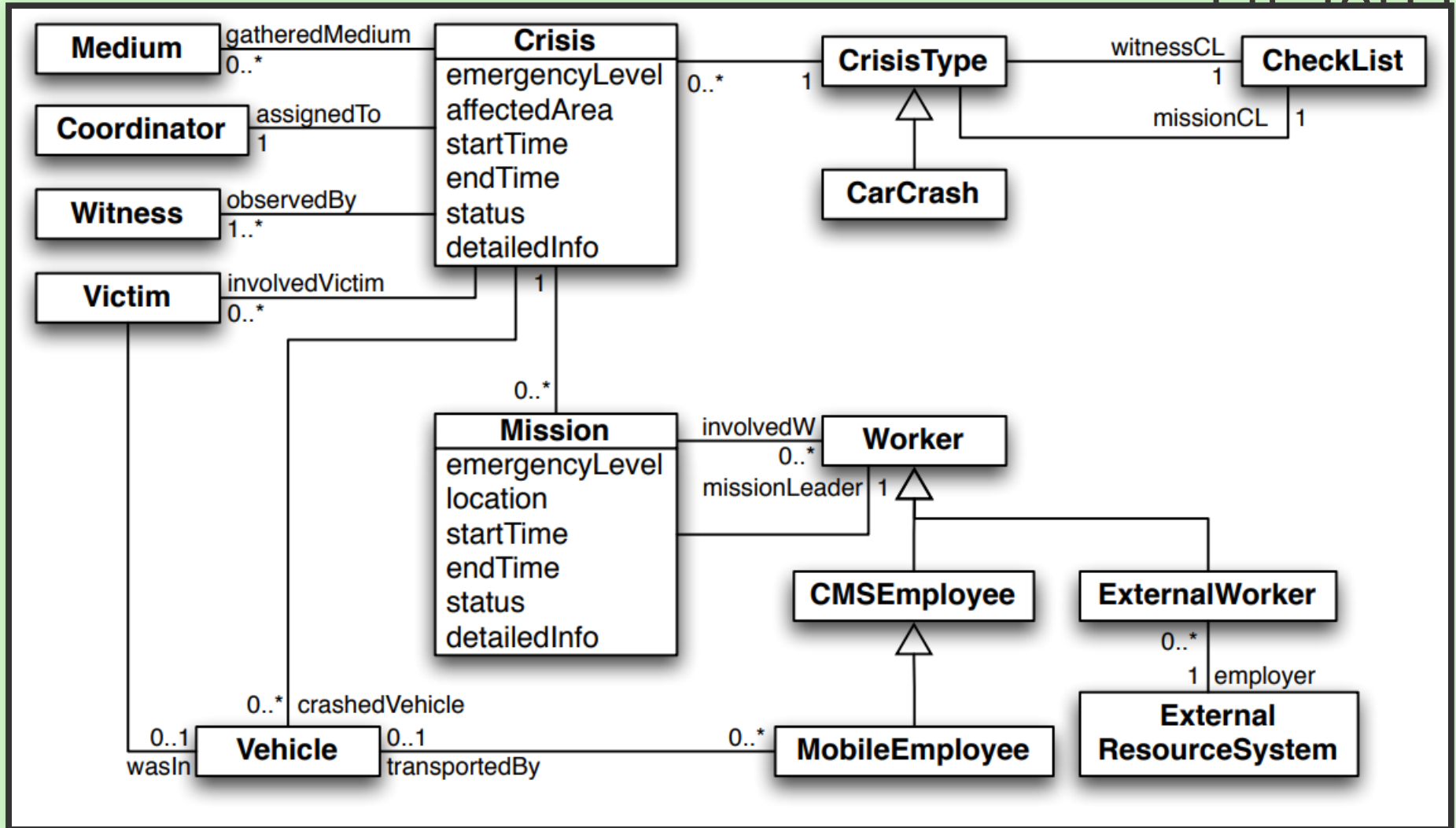
## IDENTIFIER CLASSES CONCEPTUELLES

1. Réutiliser modèle existant
2. Catégories de classes
3. Groupes nominaux

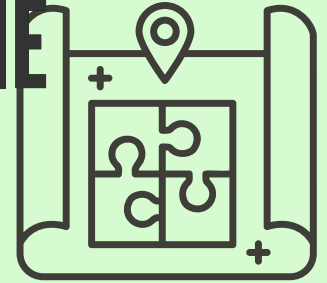
# MÉTHODE 1: MODÈLE EXISTANT



# MÉTHODE 1: MODÈLE EXISTANT



# MÉTHODE 2: CHERCHER PAR CATÉGORIE



Section 9.5 du livre.

transaction  
métier

descriptions  
d'entités

événements  
notables

lignes d'une  
transaction

catalogues

autres  
systèmes  
externes

produit ou  
service lié à une  
transaction

conteneurs

contenu d'un  
conteneur

documents  
financiers,  
contrats, etc.

où la transaction  
est-elle  
enregistrée

instruments  
financiers

lieu de la  
transaction  
ou service

Role de  
personne

objets  
physiques

plannings,  
manuels,  
etc.



# CATÉGORIE DE CLASSE

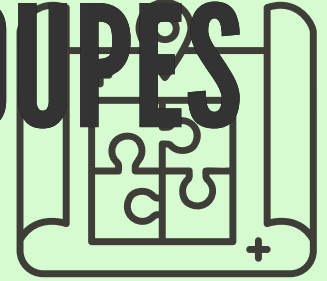


Cas d'utilisation « S'inscrire à un groupe-cours »

1. L'Étudiant commence une inscription.
2. L'Étudiant entre le sigle du cours.
3. Le Système affiche la liste des groupes-cours ainsi que l'horaire de chaque groupe-cours.
4. L'étudiant sélectionne le groupe-cours auquel il veut s'inscrire.
5. ...

*Catégories dans les **Notes de cours***

# MÉTHODE 3: IDENTIFIER LES GROUPES NOMINAUX



- Analyse linguistique du texte des spécifications (Cas d'utilisation, Récit utilisateur, etc.)
- Noms  $\rightarrow$  Classes conceptuelles ou attributs possibles
- Approche *simple*, mais *imprécise*

# IDENTIFIER LES GROUPE NOMINAUX



Cas d'utilisation « S'inscrire à un groupe-cours »

1. L'Étudiant commence une inscription.
2. L'Étudiant entre le sigle du cours.
3. Le Système affiche la liste des groupes-cours ainsi que l'horaire de chaque groupe-cours.
4. L'étudiant sélectionne le groupe-cours auquel il veut s'inscrire.
5. ...

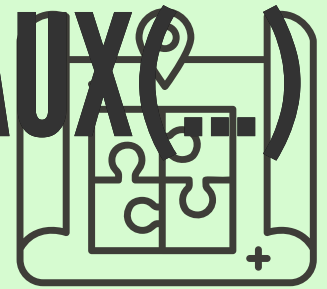
# IDENTIFIER LES GROUPES NOMINAUX



Cas d'utilisation «S'inscrire à un groupe-cours»

1. L'Étudiant commence une inscription.
2. L'Étudiant entre le sigle du cours.
3. Le Système affiche la liste des groupes-cours ainsi que l'horaire de chaque groupe-cours.
4. L'étudiant sélectionne le groupe-cours auquel il veut s'inscrire.
5. ...

# IDENTIFIER LES GROUPES NOMINAUX



Résultat: *Classes candidates*

- Beaucoup de directives (Chapitre 9)
- Classe vs Attribut?
- Attribut vs Association?

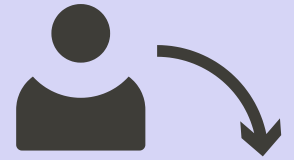
C'est de l'analyse (pas la conception)

# EXERCICE: MDD




# CLASSES CONCEPTUELLES

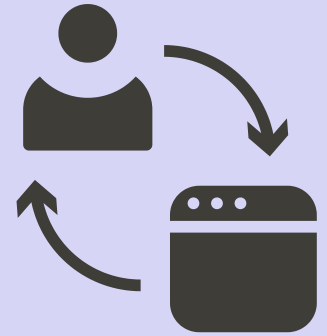
Google Classrooms



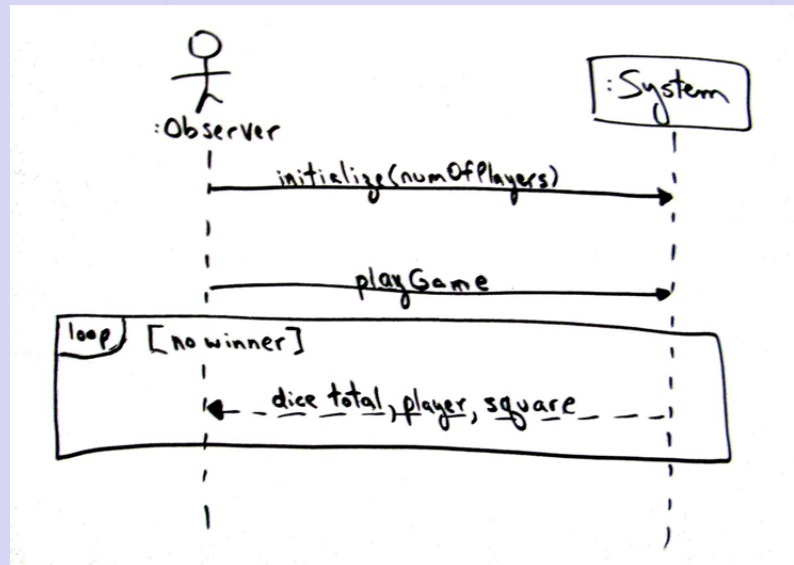
# ANALYSE ET CONCEPTION DE LOGICIELS

1. Administration
2. MDD: Catégories
3. DSS: Opérations système  S20203
4. RDCU: Contrôleur GRASP
5. Architecture en couches
6. Exercice

# DSS

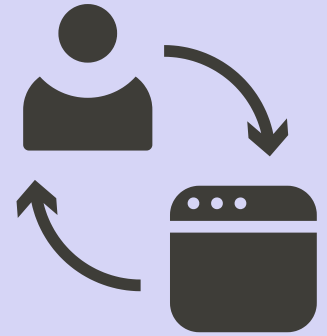


## DIAGRAMME SÉQUENCE SYSTÈME



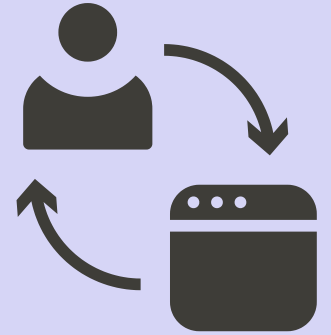


# DSS



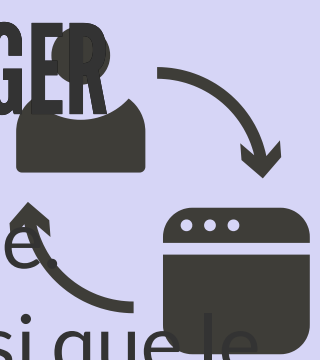
- Créé à partir d'un cas d'utilisation
- Modélise l'interaction
  - Opérations système
  - Messages de retour (au besoin)
- Notation UML
- But : définir l'API (haut niveau)
- Système est une « boîte noire »

# FAIRE UN DSS



- Identifier l'acteur principal
- Modéliser système comme boîte noire
- Proposer une opération système pour chaque évènement système
  - Types primitifs pour arguments
  - Messages de retour si nécessaire

# DSS: CU01 - AJOUTER UN LIVRE À ÉCHANGER

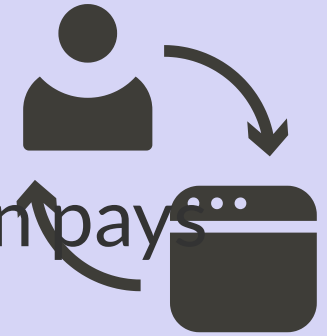


1. Le Client démarre un nouvel ajout de livre
2. Le Client entre le code ISBN du livre, ainsi que le code de sa condition.
3. Le Système enregistre le livre et présente sa description.

*Le Client répète les étapes 2 à 3 jusqu'à ce qu'il ait saisi tous les livres à échanger.*

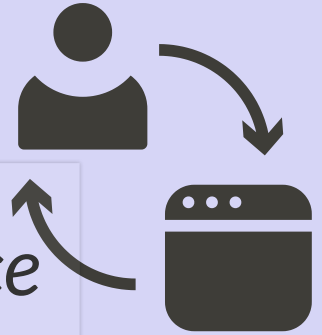
4. Le Système présente la liste de livres que possède le Client.

# DSS: «ATTAQUER UN PAYS»



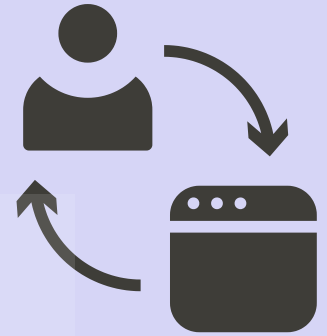
1. Le Joueur attaquant choisit d'attaquer un pays voisin du Joueur défenseur.
2. Le Joueur attaquant annonce combien de régiments il va utiliser pour son attaque.
3. Le Joueur défenseur annonce combien de régiments il va utiliser pour sa défense.
4. Les deux Joueurs jettent le nombre de dés selon leur stratégie choisie aux étapes précédentes.
5. Le Système compare les dés et élimine les régiments de l'attaquant ou du défenseur selon les règles et affiche le résultat.

# DSS: «ATTAQUER UN PAYS»

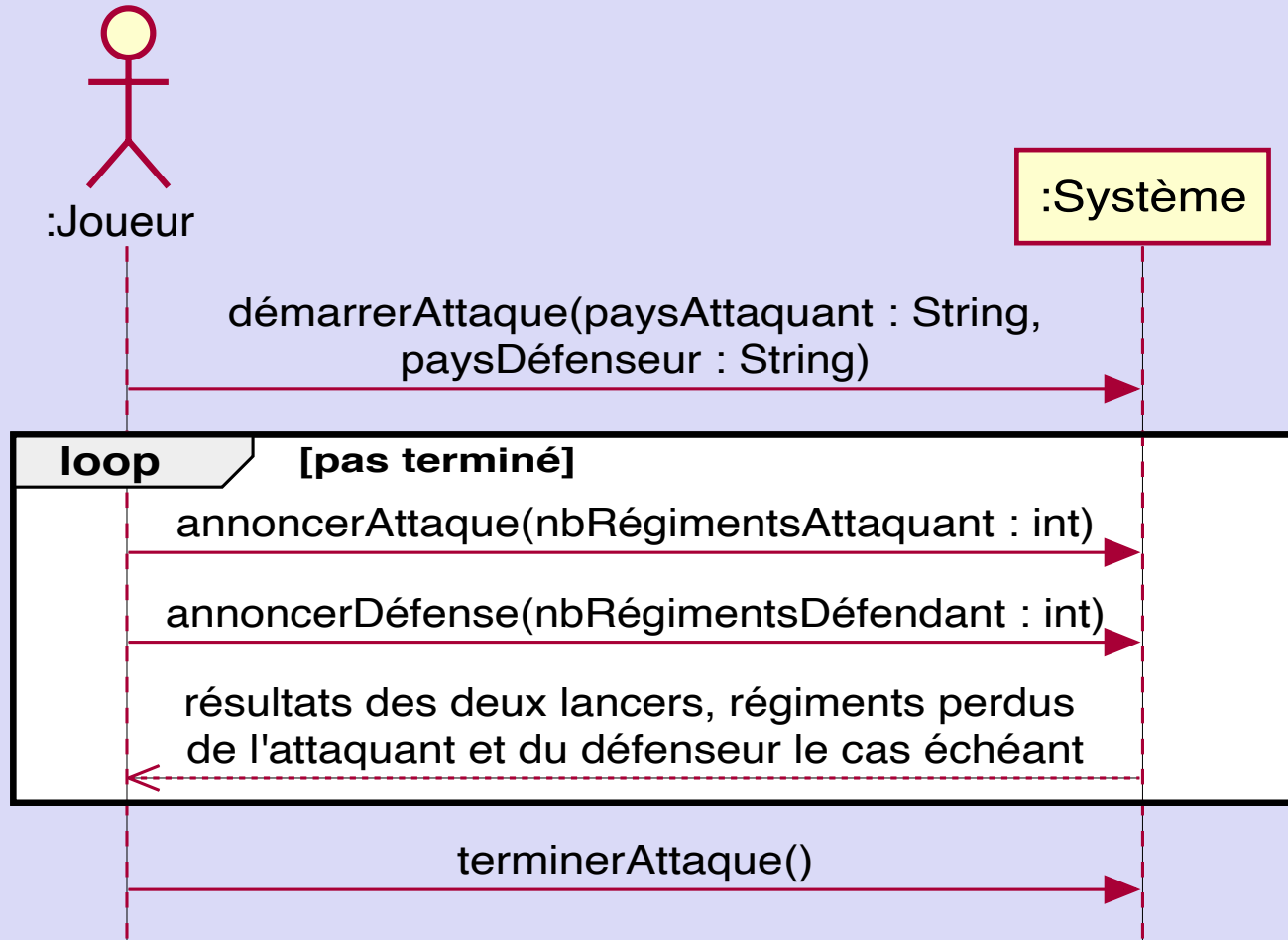


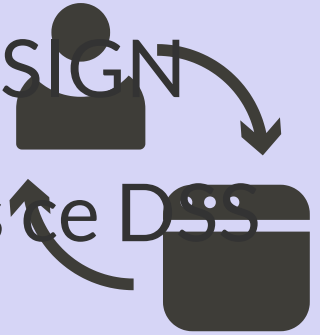
*Les Joueurs répètent l'étape 2 jusqu'à ce que l'attaquant ne puisse plus attaquer ou ne veuille plus attaquer.*

# DSS: «ATTAQUER UN PAYS»



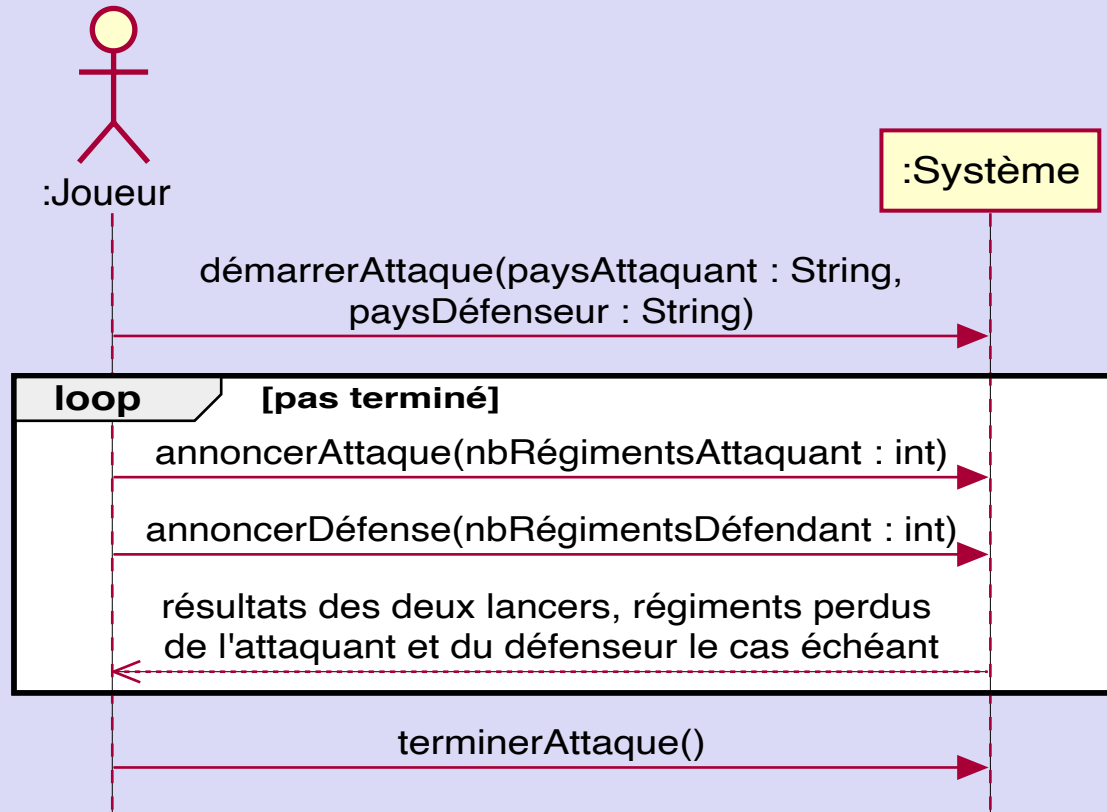
## DSS pour *Attaquer un pays*





Vrai/Faux: Il y a 5 opérations système dans ce DSS

### DSS pour *Attaquer un pays*



## ANALYSE ET CONCEPTION DE LOGICIELS

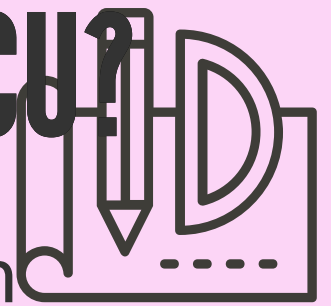


1. Administration
2. MDD: Catégories
3. DSS: Opérations système
4. RDCU: Contrôleur GRASP ← S20203
5. Architecture en couches
6. Exercice

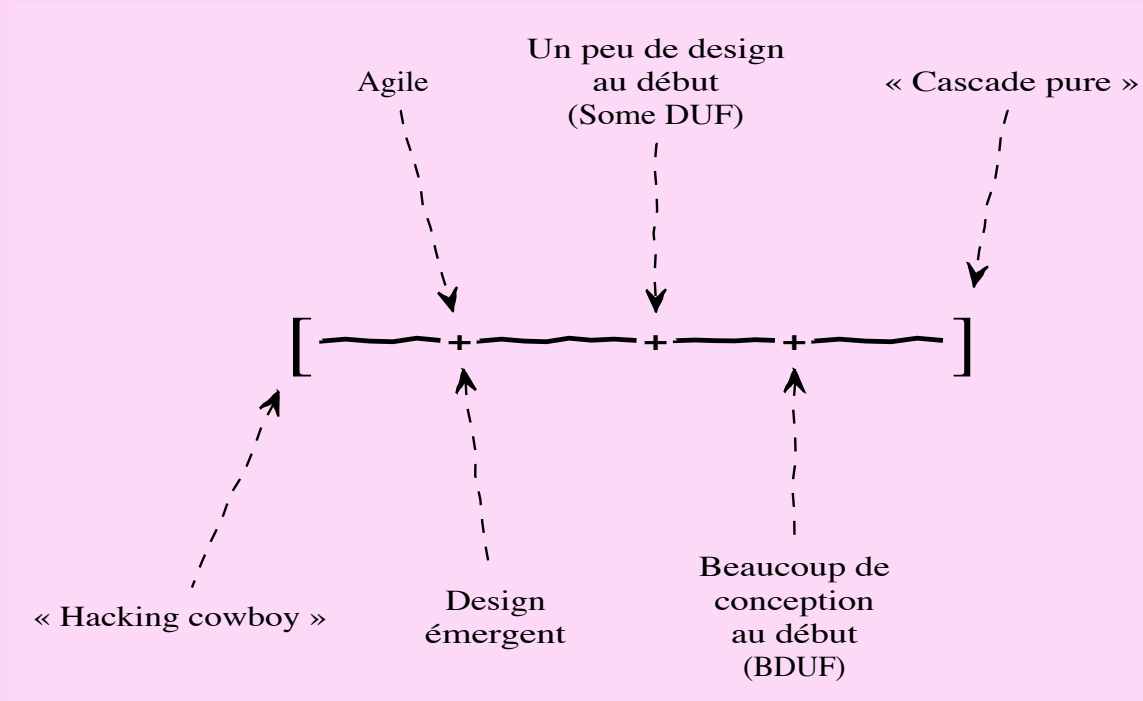




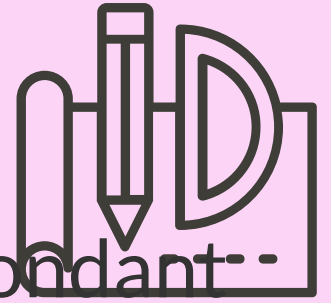
# : POURQUOI FAIRE UNE RDCU ?



Pour apprendre à faire une solution  
modulaire et intuitive

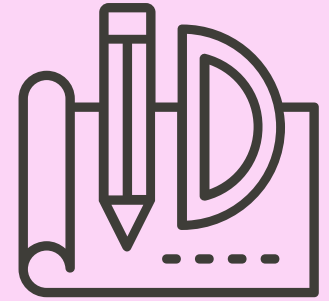


# RDCU: ASPECTS



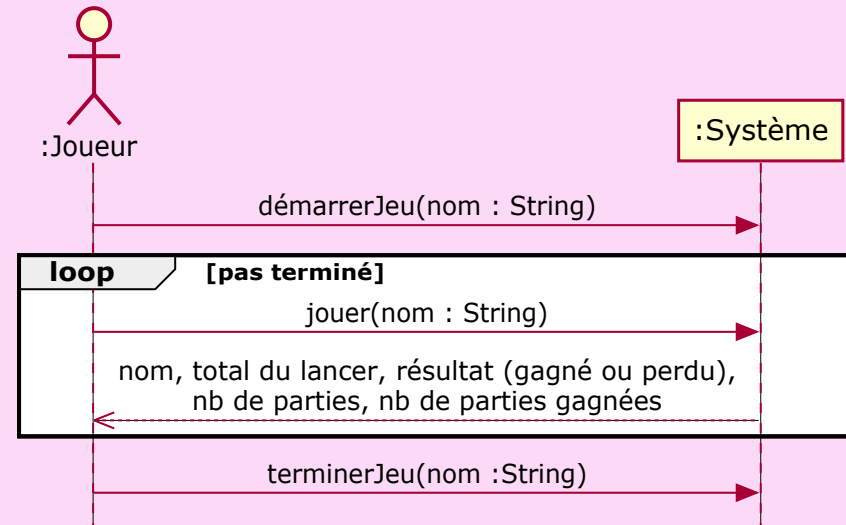
- Proposer des **classes logicielles** correspondant aux **classes conceptuelles**
  - Inspirées du MDD
  - Pour réduire le *décalage des représentations*
- Utiliser les principes GRASP
  - **Faible Couplage et Forte cohésion** (LOG121!)
  - **Contrôleur** (*architecture en couches*)

# RDCU - JEU DE DÉS

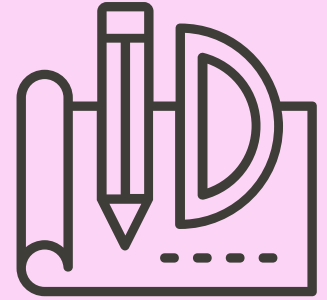


## Opérations système du DSS

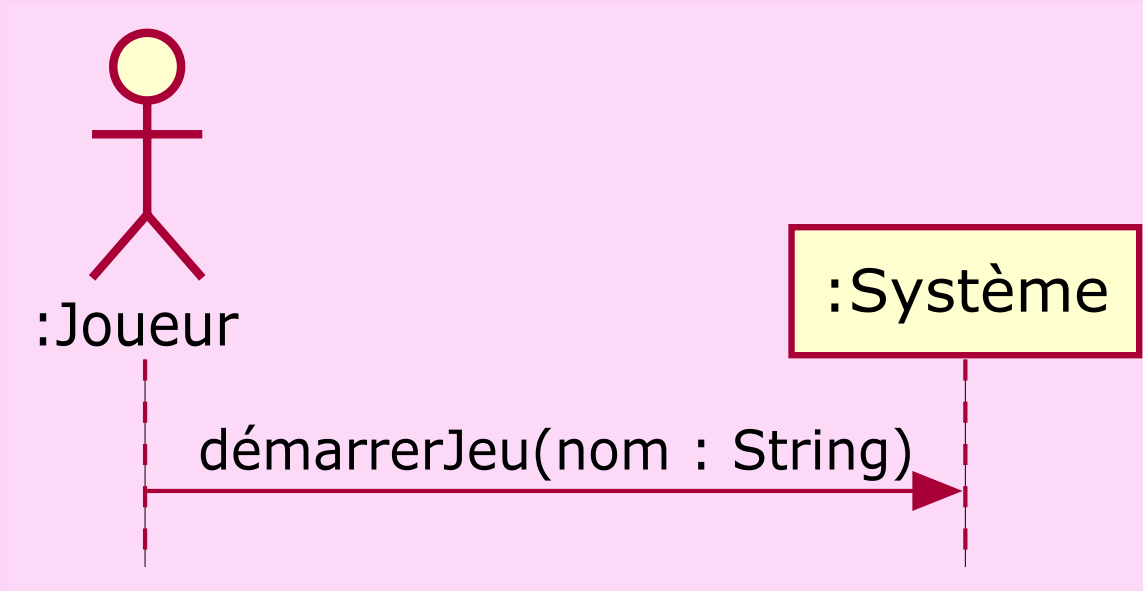
DSS pour un scénario adapté de *Jouer aux dés*  
(Ch. 1 de Larman)



# RDCU - JEU DE DÉS

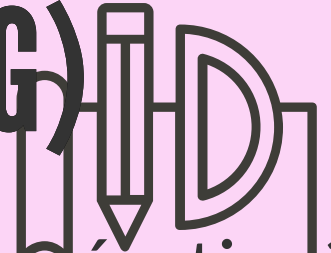


Première opération système:

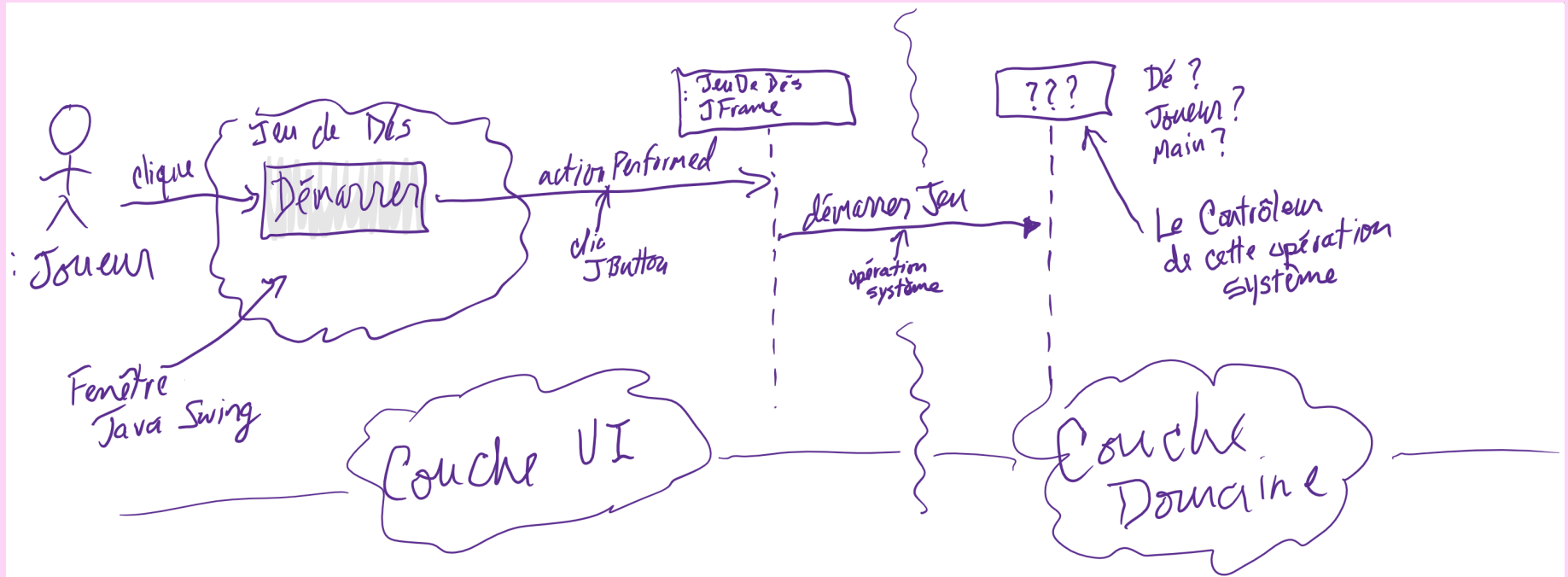


Qui envoie l'opération? Qui la reçoit?

# RDCU (SOLUTION JAVA SWING)



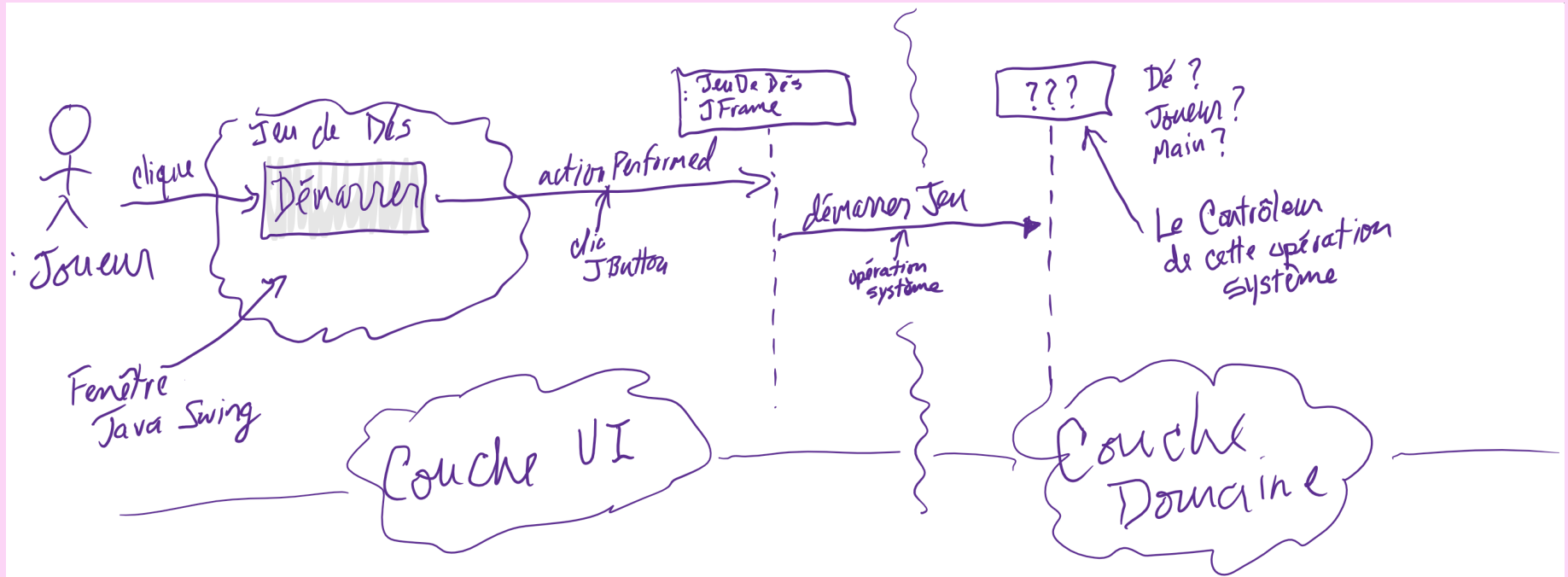
Opération démarrerJeu - qui envoie cette opération?



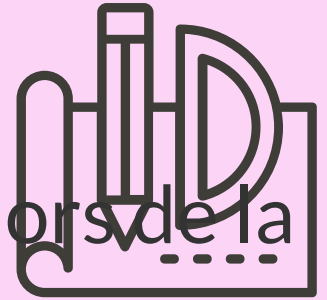
# RDCU (SOLUTION JAVA SWING)



Opération démarrerJeu - qui reçoit cette opération?



# PRINCIPE CONTRÔLEUR GRASP



**Problème:** Quel est le premier objet en dehors de la couche présentation **qui reçoit** et coordonne (« contrôle ») les opérations système ?

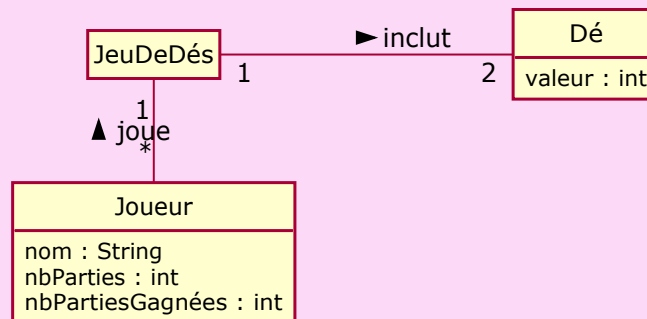
# PRINCIPE CONTRÔLEUR GRASP



**Solution:** Affectez une responsabilité à la classe qui correspond à l'une de ces définitions:

1. Elle représente le **système global**, un « **objet racine** », un **équipement** ou un **sous-système**.
2. ...

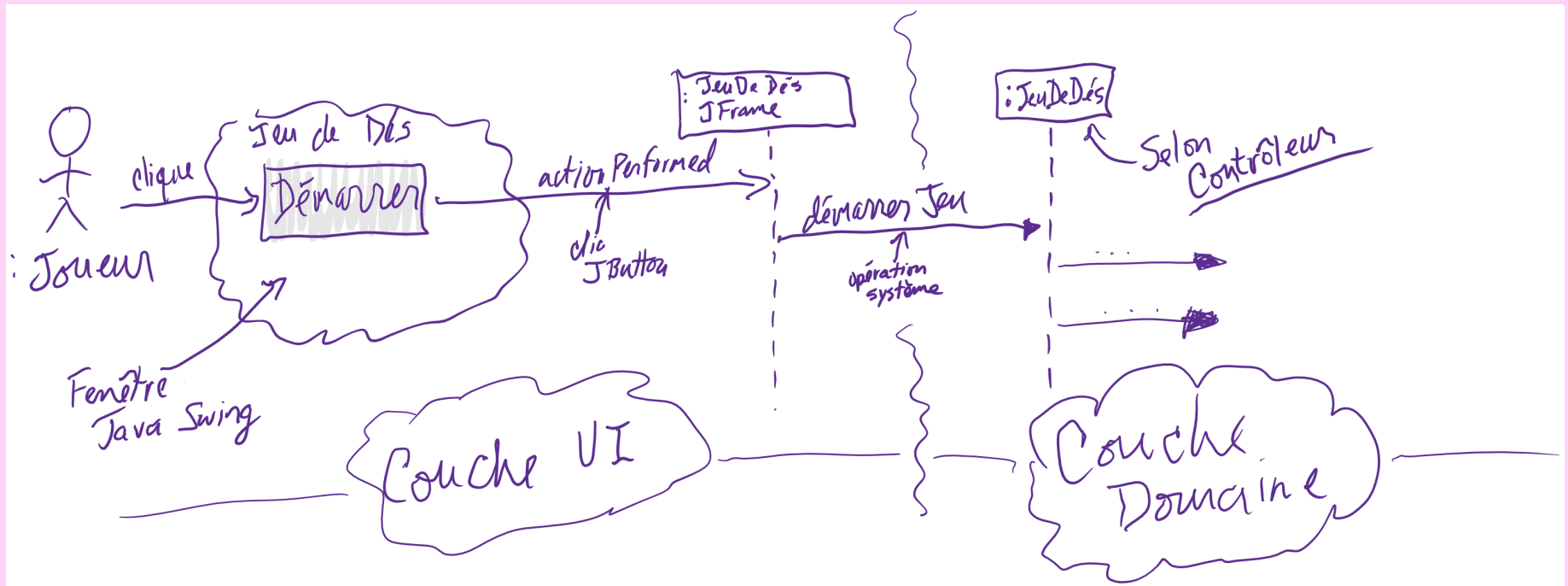
Modèle du domaine (adapté du Jeu de dés du Ch. 1 de Larman)



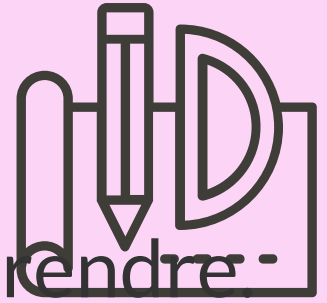


# RDCU - CONTRÔLEUR

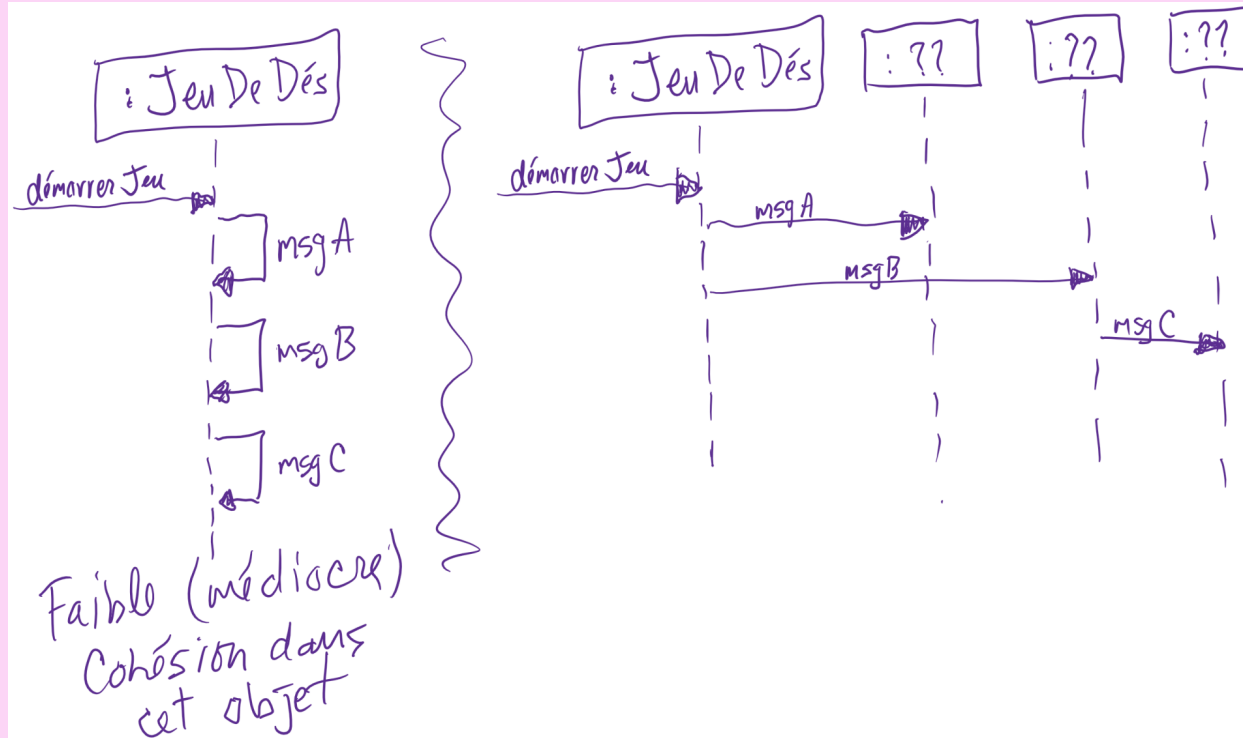
JeuDeDés est le contrôleur GRASP (inspiré du MDD)

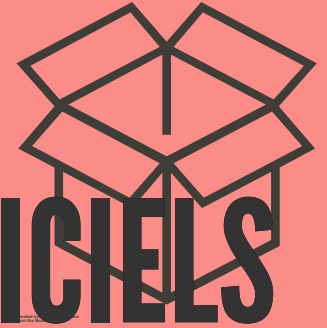


# RDCU - DÉTAILLÉ




Faire un design modulaire et facile à comprendre.  
Affecter les responsabilités aux bonnes classes.





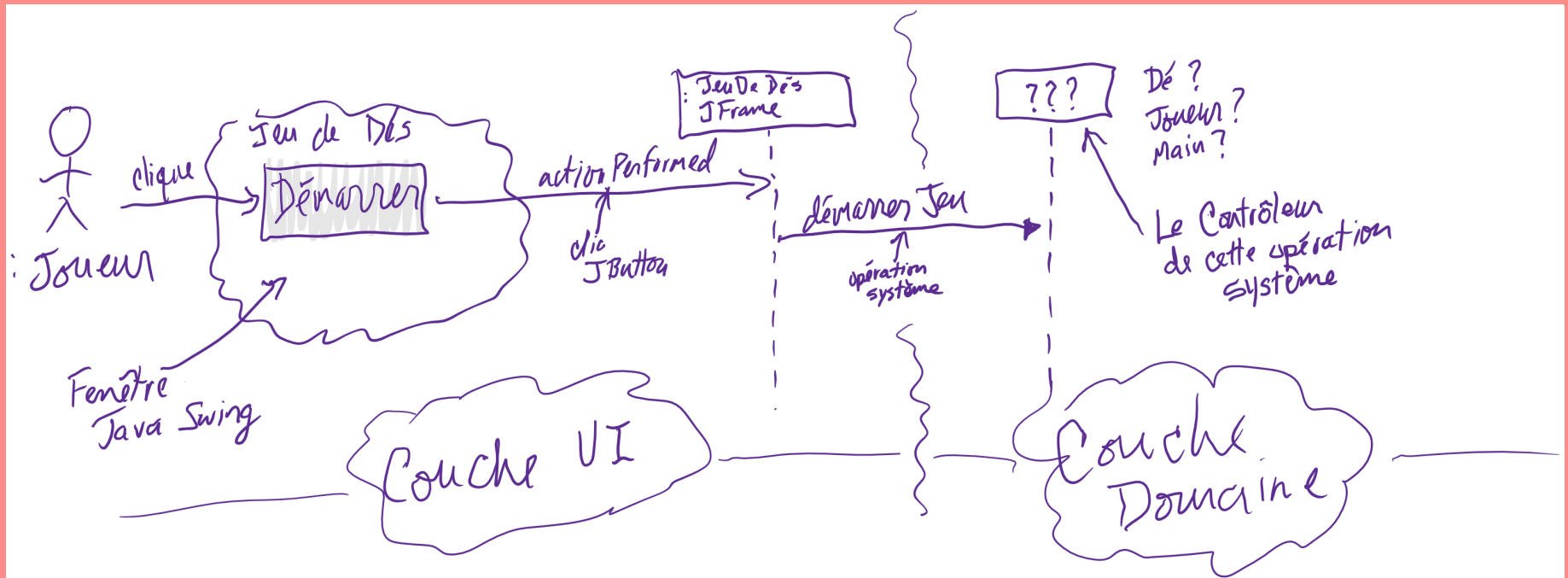
# ANALYSE ET CONCEPTION DE LOGICIELS

1. Administration
2. MDD: Catégories
3. DSS: Opérations système
4. RDCU: Contrôleur GRASP
5. Architecture en couches  S20203
6. Exercice

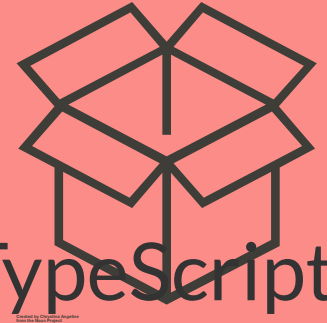
# ARCHITECTURE LOGIQUE



## Solution avec Java/Swing:



# ARCHITECTURE LOGIQUE



Solution avec Navigateur/HTTP/ExpressJS/TypeScript:



ExpressJS et JeuDeDés


# COMPRENDRE ET RESPECTER L'ARCHITECTURE EN COUCHES DU LABORATOIRE



<https://log210-cfuhrman.github.io/log210-valider-architecture-couches/>



# ANALYSE ET CONCEPTION DE LOGICIELS

1. Administration
2. MDD: Catégories
3. DSS: Opérations système
4. RDCU: Contrôleur GRASP
5. Architecture en couches
6. Exercice 

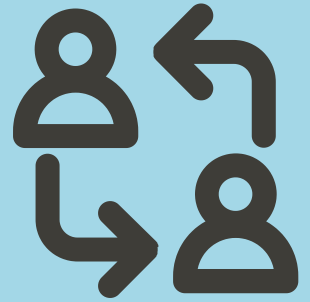
# EXERCICE MISE EN PLATEAU

<https://docs.google.com/document/d/1nGiQOeRyrY-Tx-J82z3qa0vAoUsF1rvHMNjyPGXH2mY/edit?usp=sharing>



# SÉANCE #02

## RÉTROACTION: PAGE D'UNE MINUTE



Created by Prithvi  
from the Noun Project

1. Quels sont les deux [trois, quatre, cinq] plus importants [utiles, significatives, surprenantes, dérangeantes] choses que vous avez apprises au cours de cette session?
2. Quelle (s) question (s) reste (s) en tête dans votre esprit?
3. Y a-t-il quelque chose que tu n'as pas compris?

<https://1drv.ms/u/s!An6-F73ulxAOhVyiCB46jTelNVLs>



# RÉFÉRENCES ET IMAGES

1. mapping by Vectors Point from the Noun Project
2. user interaction by mynamepong from the Noun Project
3. Prototyping by ProSymbols from the Noun Project