BUT Informatique
1A - Semestre 1
Introduction aux bases de données et SQL
(R1.05)

R. Fleurquin

Chapitre 0

Pourquoi ce module?

Objectifs, contenu et modalités

Comment développer une application logicielle?

Des Besoins

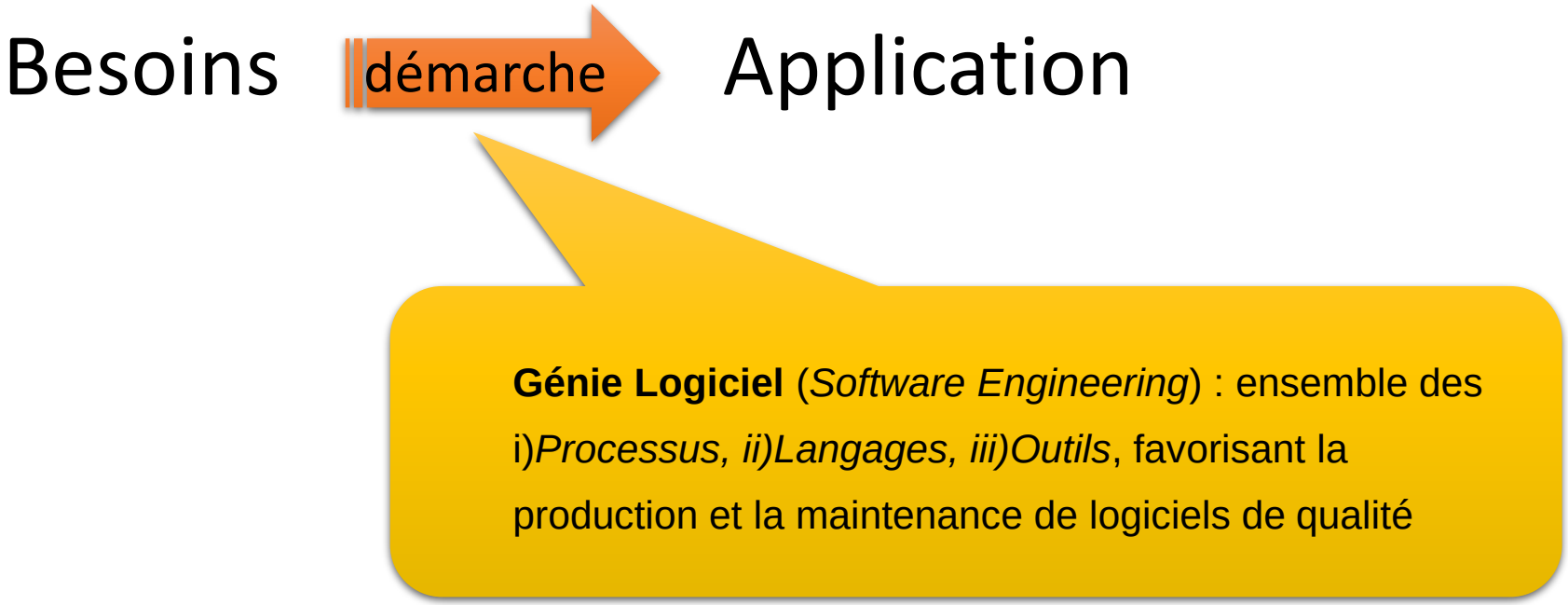
Une Application

De l'expérience?
Du talent?
De la magie?

De qualité ! (ISO 25010)

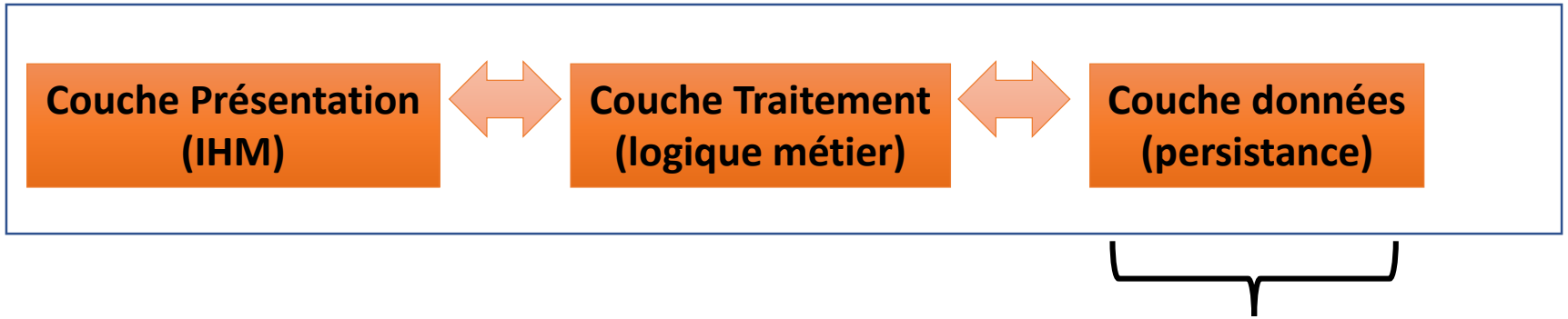
1. Capacité fonctionnelle
2. Facilité d'utilisation
3. Fiabilité
4. Performance
5. Maintenabilité
6. Portabilité

Pour réussir il faut suivre une démarche rationnelle intégrant les « *meilleures pratiques* »



Les **3 tiers** d'un logiciel : offrir une *interface* permettant de réaliser des *traitements* en usant et capitalisant de *données*

Architecture classique d'un Logiciel



R1.05 s'intéresse au développement de la partie « persistante » => obtenir une ***base de données*** (ici dans le paradigme relationnel!)

Une *base de données relationnelle* c'est des *tables* qui contiennent des… données!

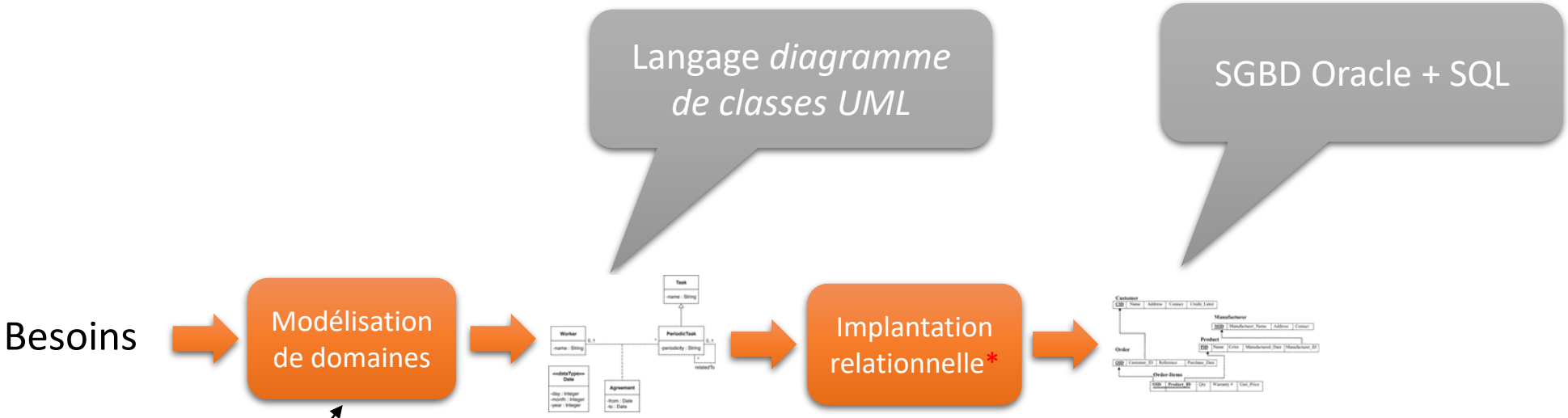
Étudiant

Nom	Age	Adresse	Apprenti	@Promotion
Fleurquin	19	« 3 rue Paul »	True	2A
Ridard	18	« 21 avenue Roi »	False	1A
…	…	…	…	…

Chaque ligne de la table stocke des informations ***typées*** et ***structurées*** en colonne sur une « entité » d'intérêt pour l'application (ici les informations sur un étudiant particulier).

La problématique de R1.05 : « Quelle démarche suivre pour, partant des besoins, développer ces tables puis les utiliser? »

. R1.05 : « Quelle démarche suivre pour développer ces tables? »

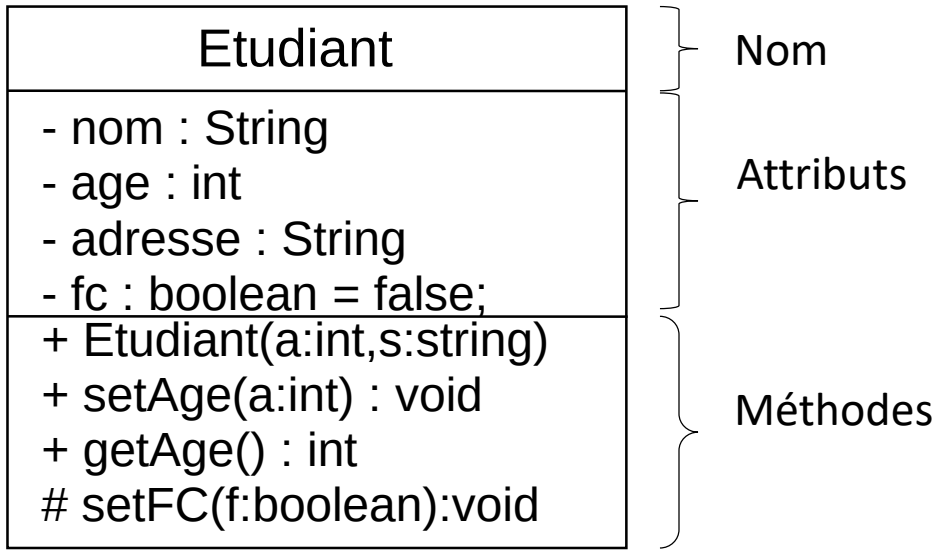


Période 1 => Comprendre le paradigme cible et Apprendre à modéliser en UML
Responsable : R. Fleurquin, 45'C+3hTD par semaine

Période 2 => Apprendre à implanter en SQL un diagramme UML puis à « manipuler » (requêtage) les bases de données relationnelles
Responsable : M.T. Pham, 45'C+1h30TD+1h30 TP par semaine

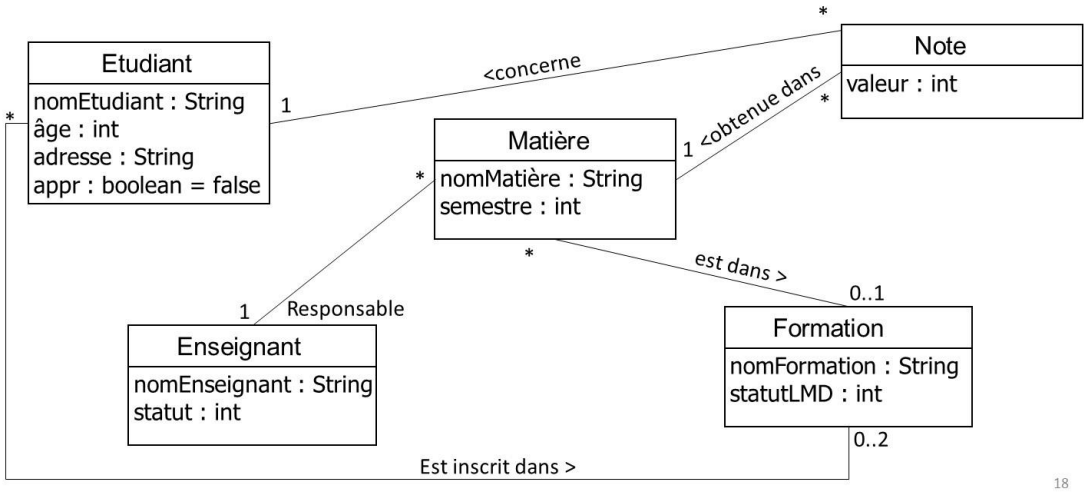
Nous développerons dans le paradigme Objet

- Il existe plusieurs « approches » (paradigmes) de développement.
- Nous utilisons le paradigme « Objet ».
- Le paradigme objet consiste à concevoir le code comme un assemblage de briques élémentaires appelées ***classes***.
- Ces classes ont la bonne idée de beaucoup « ressembler » aux objets de notre univers quotidien : ils ont des caractéristiques (***attributs***) et savent faire des choses (***méthodes***).
- Cette ressemblance est voulue! Elle garantit plein de bonnes propriétés aux codes ainsi structurés…
  - Par exemple les tables que nous dériverons de nos diagramme de classes respecteront plus facilement les formes normales (des règles qui évaluent la qualité structurelle d'une base de données relationnelle).



Nous développerons avec UML

- Le standard de référence est UML (OMG, 754 pages, une douzaine de langages!)
- Dans R1.05 nous travaillerons avec 2 des langages d'UML : le *diagramme de classes* et le *diagramme d'objets*
- Nous n'utiliserons qu'une partie seulement de la puissance de ces 2 langages et en particulier les concepts langagiers qui suivent :
  - Classe
  - Attribut
  - Association binaire (réflexive ou non)
  - Classe association (attribut porté)
  - Association n-aire (pour les plus à l'aise!)
  - Relation de généralisation
  - Contraintes textuelles



Évaluation, Mise en pratique

- 2 périodes : P1 (6 semaines) puis P2 (7 semaines)
- 2 notes par période (1CC et un CT)
- En période 1 : 1CC (en cours de période)+1CT(semaine 42, 1H30 avec doc)
- En période 2 : voir avec M.T. Pham
- Attention : une SAÉ associée (S1.04, *Création d'une base de données*) qui débutera vers la semaine 40
- Le plan prévu (en gros) :
  - 2 semaines sur la découverte du paradigme relationnel et la normalisation
  - 2 semaines sur l'apprentissage d'UML diagramme de classes
  - 2 semaines sur la modélisation de domaines
- Coefficient UE1.4 : 36% (40% pour la SAE S1.04)