

E4D : ÉTUDE DE CAS

Durée : 5 heures

Coefficient : 5

CAS CAPDC

*Ce sujet comporte 16 pages dont 8 pages d'annexes.
Le candidat est invité à vérifier qu'il est en possession d'un sujet complet.*

Matériels et documents autorisés

- Règle à dessiner les symboles informatiques
- Lexique SQL sans commentaire ni exemple d'utilisation d'instruction

L'usage de la calculatrice n'est pas autorisé pour cette épreuve.

Liste des annexes

- Annexe 1 :* Extrait de cahier d'épandage de l'exploitation « La Hautière »
Annexe 2 : Schéma partiel du site informatique de la Chambre d'Agriculture
Annexe 3 : Extrait du schéma relationnel de la base de données « TraitementPhyto »
Annexe 4 : Diagramme partiel des classes métiers
Annexe 5 : Description textuelle des classes métiers
Annexe 6 : Description textuelle des classes techniques
Annexe 7 : Algorithme de la méthode *chargerLeTraitement()* de la classe *Passerelle*

Barème

Dossier 1	Informatisation du cahier d'épandage	25 points
Dossier 2	Évolution de l'architecture informatique à la Chambre d'Agriculture	15 points
Dossier 3	Développement de l'application « Registre phytosanitaire »	45 points
	Partie A : Exploitation de la base de données	20 points
	Partie B : Réalisation de l'application	25 points
Dossier 4	Évaluation des charges du projet AIM	15 points
	Total	100 points

CODE ÉPREUVE : ISE4D	EXAMEN : BREVET DE TECHNICIEN SUPÉRIEUR	SPÉCIALITÉ : INFORMATIQUE DE GESTION Option Développeur d'applications
SUJET	ÉPREUVE : ÉTUDE DE CAS	
Coefficient : 5	Code sujet : 08DA01N	Page : 1/16

Présentation du contexte

La Chambre d'Agriculture du Pas-de-Calais (CAPDC) assure des missions d'accompagnement pour le développement de l'agriculture dans son département.

Elle propose aux agriculteurs des prestations diverses :

- analyses de projets d'installation, de diversification, ... ;
- actions de sensibilisation et de promotion autour des métiers de l'agriculture ;
- formation et conseils sur la prise en compte de préoccupations environnementales (gestion des milieux naturels, élimination des déchets, ...).

Au titre de la protection des milieux naturels, les agriculteurs sont tenus de rendre compte des quantités d'engrais et de traitements (pesticides, désherbants, ...) apportés à leurs cultures. Ce contrôle conditionne les aides accordées aux agriculteurs dans le cadre de la politique agricole commune (PAC). Les agriculteurs doivent donc remplir régulièrement un cahier d'épandage pour les engrais et un registre phytosanitaire pour les traitements.

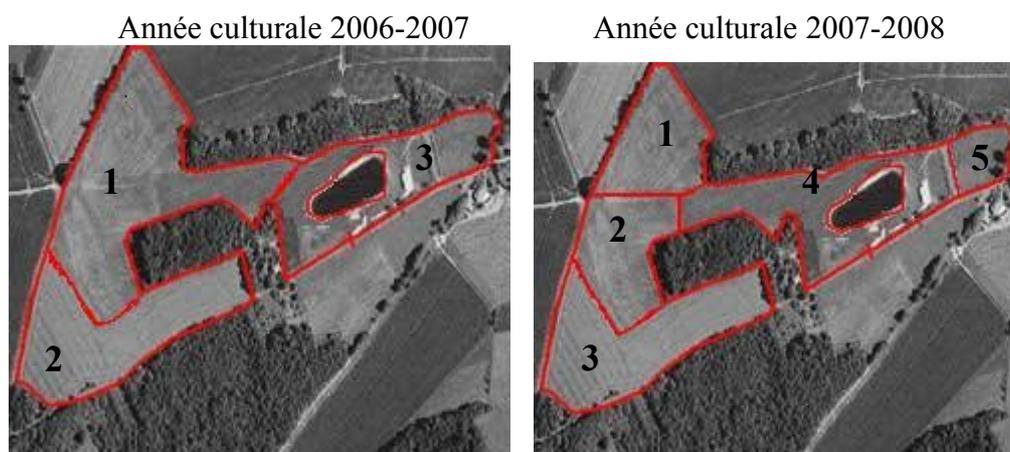
Le service informatique de la Chambre d'Agriculture a sollicité la SSII Sigeto pour collaborer au projet AIM (Agriculture Information Maîtrisée) dont l'objectif est de faciliter la gestion des échanges entre les agriculteurs et les autres acteurs du monde agricole.

Le logiciel AIM permettra de saisir, stocker, consulter et échanger des données relatives à l'exploitation agricole et aux activités de culture et d'élevage. Il constituera une aide à la déclaration, à la traçabilité et à la gestion technico-économique de l'exploitation.

Le système d'information à concevoir mémorisera les données concernant l'épandage de produits fertilisants azotés apportés aux cultures des exploitations agricoles du département sur plusieurs années culturales. Chaque année culturale débute le 1^{er} juillet de l'année n et se termine le 30 juin de l'année n+1.

Chaque exploitation est composée de plusieurs îlots. Un îlot est un regroupement de parcelles contiguës, limité par des éléments facilement repérables et permanents (comme un chemin, une route, un ruisseau...) et stable d'une année culturale sur l'autre. Le découpage d'un îlot en parcelles est variable d'une année culturale à l'autre, en fonction des mises en cultures envisagées.

Exemple : Découpage d'un îlot



Pour une exploitation, on mémorise un code exploitation et les coordonnées de l'exploitant.

La collecte des informations pour une exploitation donnée s'effectuait jusqu'à présent sur un document papier appelé cahier d'épandage (cf. *annexe 1*).

Dans ce cahier d'épandage sont notées, pour chaque parcelle cultivée :

- La surface en hectares (ha) ;
- L'espèce cultivée (blé, orge, betteraves, ...) caractérisée par un code et un libellé ainsi que les rendements prévus et ceux réalisés ;
- Les informations concernant les apports en produits azotés. Pour chaque produit azoté, on indique notamment la quantité d'azote apportée par hectare et la date de l'apport. Les produits azotés peuvent être de nature minérale (engrais minéraux de synthèse) ou de nature organique (déjection animale, boues de station d'épuration, compost, ...). Un produit minéral a une teneur en azote qui lui est propre.
- L'apport de produits de nature organique donne lieu à l'enregistrement d'informations supplémentaires :
 - l'origine,
 - le délai d'enfouissement,
 - la présence d'un traitement anti-odeur,
 - la teneur en azote mesurée.

Travail à faire

1.1	Proposer un schéma entité-association représentant les informations nécessaires à l'informatisation de la collecte et du traitement des cahiers d'épandage des exploitations agricoles du département sur plusieurs années.
-----	---

Dossier 2 Évolution de l'architecture informatique à la Chambre d'Agriculture

Annexe à utiliser : 2

Une connexion haut débit SDSL et une plate-forme informatique (cf. *annexe 2*) sont installées dans les locaux de la chambre d'agriculture pour assurer le fonctionnement des nouvelles applications. Les serveurs « SRV-WEB », « SRV-SQL » et « SRV-MAIL » remplissent les rôles, respectivement, de serveur d'application Web, de serveur de bases de données (SGBDR) et de serveur de messagerie. Les bases de données situées sur le serveur « SRV-SQL » sont exploitées et mises à jour uniquement par le biais des applications Web.

Travail à faire

2.1	Donner la configuration IP - adresse IP, masque réseau et passerelle - du serveur « SRV-WEB ».
-----	--

Après étude, l'administrateur réseau décide de ne pas conserver le serveur « SRV-SQL » dans la zone démilitarisée, mais de le transférer dans l'enceinte du réseau local.

2.2	Justifier le déplacement du serveur « SRV-SQL » dans le réseau local.
-----	---

Avant transfert du serveur « SRV-SQL », les règles de filtrage du routeur pare-feu R1 en vigueur sur l'interface réseau 179.170.12.150 (côté DMZ) sont les suivantes :

N° règle	IP source	Port source	IP destinataire	Port destinataire	Décision
1	Toutes	Tous	179.170.12.151	80	Accepté
2	Toutes	Tous	179.170.12.152	25	Accepté
3	Toutes	Tous	179.170.12.152	110	Accepté
Défaut	Toutes	Tous	Toutes	Tous	Rejeté

Les règles de filtrage sur l'interface 192.168.0.254 (côté réseau local) sont les suivantes :

N° règle	IP source	Port source	IP destinataire	Port destinataire	Décision
Défaut	Toutes	Tous	Toutes	Tous	Rejeté

Protocole / application	Port utilisé	
HTTP	80	Une règle traduit un droit ou un refus d'accès. Par exemple, la règle n° 1 de l'interface 179.170.12.150 indique que tout paquet entrant sur cette interface, en provenance de n'importe quelle machine du réseau Internet et à destination de la machine d'adresse IP 179.170.12.151 et du port 80 (protocole HTTP), est accepté. Chaque paquet reçu sur une interface est traité de la manière suivante : <ul style="list-style-type: none">• en respectant l'ordre des règles de 1 à n, la 1^{ère} règle applicable est recherchée,• si une des règles est applicable, alors la décision associée à la règle est appliquée et le parcours des règles est arrêté,• si aucune règle n'est applicable, c'est la règle par défaut qui est appliquée.
POP	110	
SMTP	25	
SGBDR	5432	
Telnet	23	

Travail à faire

2.3	Proposer une nouvelle règle sur l'interface 192.168.0.254 qui permet au serveur « SRV-WEB » de communiquer avec le serveur « SRV-SQL », désormais dans le réseau local, avec 192.168.0.153 pour adresse IP. <i>Préciser l'ordre d'application de cette règle par rapport à la règle actuelle.</i>
-----	---

Le serveur « SRV-SQL », récemment acquis, dispose d'un contrôleur RAID assurant les solutions RAID1 et RAID5 et pilotant 4 disques de 300 Go.

2.4	Calculer et comparer les capacités utiles de stockage du serveur « SRV-SQL » pour chaque solution RAID1 et RAID5. <i>Justifier les calculs.</i>
-----	---

Les Traitements Phytosanitaires

Les exploitants agricoles sont souvent amenés à épandre sur les cultures des produits afin de lutter contre les maladies éventuelles des espèces cultivées. On appelle cette opération un traitement phytosanitaire car il utilise des produits pesticides (organiques ou chimiques).

Un traitement phytosanitaire peut concerner une semence (il est alors mélangé à la semence avant le semis) ou une culture en champ (il est alors appliqué en une ou plusieurs pulvérisations sur cette culture).

À cet effet, les exploitants sont tenus de remplir un registre phytosanitaire, fourni par la Chambre d'Agriculture, enregistrant les différents épandages. Ce registre est actuellement un document papier. Il est envisagé de proposer aux agriculteurs qui le désirent une application spécifique. Cette application, actuellement en phase de test, permettra les saisies des informations par les agriculteurs eux-mêmes (via Internet) ainsi qu'un traitement centralisé de ces informations (contrôles, statistiques).

A) Exploitation de la base de données

La base de données utilisée pour l'application est présentée en *annexe 3*.

Une parcelle peut faire l'objet de traitements ; si le traitement est de type « semence », il est unique car le produit est mélangé au semis. En revanche, si le traitement est « en champ », celui-ci est constitué d'une ou plusieurs pulvérisations. Le champ *typeTraitement* indique la nature de ce traitement. Le champ *dosageTraitementSemence* enregistre le dosage (quantité par unité de surface) pour le traitement « en semence » ; le champ *dosage* de la table *Pulvérisation* précise le dosage de chaque pulvérisation.

Remarque : Le SGBDR utilisé dispose de la fonction *dateDiff(uneDateDebut, uneDateFin)* qui renvoie en nombre de jours le résultat de la soustraction de la date *uneDateDebut* à la date *uneDateFin*.

Exemple : *dateDiff('15-03-2008', '25-04-2008')* retourne la valeur 41.

Une nouvelle législation entraînant l'interdiction de pulvérisations dans les 30 jours qui précèdent la récolte, la Chambre d'Agriculture souhaite adresser un courriel d'information à l'ensemble des exploitants qui possèdent des parcelles ayant fait l'objet de pulvérisations moins de 30 jours avant la date de récolte prévue.

<i>Travail à faire</i>	
3.1	Écrire la requête SQL permettant d'obtenir les informations nécessaires pour ce traitement.
3.2	Écrire la requête SQL qui retourne le nom des exploitants et l'identifiant des parcelles qui ont fait l'objet de plus de quatre pulvérisations.
3.3	Écrire la requête SQL qui retourne le nom des exploitants ainsi que l'identifiant des parcelles qui n'ont fait l'objet d'aucun traitement.

Il est prévu de sécuriser l'application en évitant les saisies indésirables remettant en cause les règles de gestion métier. C'est ainsi que des déclencheurs (*triggers*) assureront l'intégrité des données.

Un déclencheur a été écrit :

```
CREATE TRIGGER trg_insert_pulverisation
BEFORE INSERT
ON Pulverisation
FOR EACH ROW
```

NEW désigne ici la ligne de la table Pulverisation en cours d'insertion.

```
DECLARE typeTr CHAR(1) // déclaration de la variable locale typeTr
SELECT typeTraitement INTO typeTr FROM traitement
WHERE traitement.id = NEW.idTraitement
IF typeTr = 's'
BEGIN
    RAISERROR // génère une erreur
END
```

<i>Travail à faire</i>	
3.4	Indiquer la règle de gestion prise en charge par le déclencheur.
3.5	Écrire le déclencheur qui permet de respecter la règle de gestion suivante : « <i>pour une parcelle, il ne peut y avoir qu'un seul traitement en semence puisque le produit phytosanitaire est mélangé au semis</i> ».

B) Réalisation de l'application

L'application permettra également de visualiser les informations saisies par les exploitants. Pour ce faire, l'emploi d'un langage de programmation objet est envisagé ; le diagramme partiel des classes métier est présenté en *annexe 4*, la description textuelle associée est présentée en *annexe 5*.

<i>Travail à faire</i>	
3.6	Écrire le constructeur de la classe <i>TraitementSemence</i> .
3.7	Écrire les méthodes <i>quantitéAppliquée()</i> des classes : <ul style="list-style-type: none"> a) <i>TraitementSemence</i> b) <i>TraitementEnChamp</i>.

Pour charger en mémoire les instances des classes métier, une classe *Passerelle* est utilisée ; sa responsabilité est de récupérer les informations de la base de données « *TraitementPhyto* » et d'instancier les objets des classes métiers.

L'annexe 6 présente la description de cette classe.

<i>Travail à faire</i>	
3.8	Écrire la méthode <i>chargerLesPulvérisations()</i> de la classe <i>Passerelle</i> .

La méthode *chargerLeTraitement()* construit et retourne un objet de la classe *TraitementPhytosanitaire* à partir de la base de données, cet objet pouvant être de la classe *TraitementSemence* ou de la classe *TraitementEnChamp*.

L'algorithme de cette méthode figure en annexe 7 ; une partie reste à détailler.

<i>Travail à faire</i>	
3.9	Écrire les instructions manquantes de la méthode <i>chargerLeTraitement()</i> de la classe <i>Passerelle</i> . Le complément demandé est à reporter sur la copie.

Dossier 4	Évaluation des charges du projet AIM
------------------	---

Les applications développées par la Chambre d'Agriculture concernant l'épandage et le registre phytosanitaire constituent des modules dans le cadre d'un projet plus vaste, visant à créer une plate-forme d'échange d'information entre les agriculteurs et les autres acteurs du monde agricole : le projet AIM (Agriculture Information Maîtrisée). Le projet AIM a été évalué à 5 000 jours-hommes (JH), hors déploiement. La responsabilité du projet AIM incombe à la SSII Sigeto, retenue par la Chambre d'Agriculture.

La planification initiale a été établie par la maîtrise d'ouvrage :

Phase	Charge	Responsabilité
Conception fonctionnelle	1 000 JH	Chambre d'Agriculture et Sigeto
Conception et développement	2 100 JH	Sigeto
Tests	900 JH	Chambre d'Agriculture et Sigeto
Communication	400 JH	Chambre d'Agriculture
Formation	600 JH	Chambre d'Agriculture et Sigeto
Déploiement	5 625 JH	Prestataire

La planification initiale a été réalisée sur 36 mois. On retiendra une moyenne de 20 jours ouvrés par mois.

Le projet a prévu le déploiement de l'application sur 1 400 exploitations agricoles en 2007, sur les 15 000 potentielles de la Région.

La charge de déploiement est estimée à ½ JH par exploitation.

On souhaite parvenir à 75 % de couverture fin 2009.

<i>Travail à faire</i>	
4.1	Indiquer le nombre d'exploitations visées par le taux de couverture. Justifier la charge de la phase de déploiement notifiée ci-dessus.

La phase de déploiement peut avoir lieu à partir de la fin des phases de conception développement et test. En 2007 elle a consommé 625 JH.

<i>Travail à faire</i>	
4.2	Indiquer le nombre d'exploitations ayant bénéficié de la phase de déploiement fin 2007.
4.3	Dresser le bilan de cette phase. <i>Commenter.</i>

Entre janvier et avril 2008 le déploiement a été suspendu.

<i>Travail à faire</i>	
4.4	Proposer une planification mensuelle de la phase de déploiement et les ressources humaines minimales à mettre en œuvre chez le prestataire pour atteindre l'objectif de 75 % en 2009.

Annexe 1 : Extrait de cahier d'épandage de l'exploitation « La Hautière »

CAHIER D'EPANDAGE

Année culturale 2006/2007

Renseignements sur la parcelle		Renseignements sur les apports d'azote								Renseignements sur les effluents				
Référence de la parcelle : numéro îlot-numéro de parcelle dans l'îlot surface	Espèce cultivée implantée Rendement prévu Rendement réalisé	Date des divers apports	Libellé, nature du produit épandu (minéral ou organique)	Quantité de produit épandu / ha	Quantité totale épandue	Teneur en azote minérale estimée	Apport total d'azote organique	Apport total d'azote minéral	Dose d'azote	Origine de l'effluent E=exploitation, T=tiers	Délai d'enfouissement immédiat / 12h / 24h	Teneur en azote mesurée kg / t	Traitement anti-odeur (oui / non)	
				kg / ha	t	kg / t	kg	kg	kg / ha					
Sur la parcelle 1 de l'îlot 1 (5 ha), culture de betterave avec un objectif de 60 tonnes. Lors du premier épandage, réalisé en septembre, il y a eu apport de 30 tonnes de fumier / ha, la teneur moyenne de ce fumier a été mesurée à 5,5 kg d'azote par tonne. La fertilisation est complétée en avril par un second épandage avec un apport d'ammonitrate 27 dosé à 55 unités par hectare et en juin par un apport de fumier de 20 tonnes par hectare avec une teneur d'azote mesurée de 6,5 kg par tonne.														
1-1 5 ha	Betteraves 60 t 65 t	1	5-sept.	Fumier	30 000 kg / ha	150 t		825		165	E	24	5,5	Non
		2	15-avr.	ammo 27	200 kg / ha	1,02 t	270		275	55				
		3	3-juin	Fumier	20 000 kg / ha	100 t		650		130	T		6,5	Oui
		4												
							sous-total	1475	275					
Sur la parcelle 2 de l'îlot 1 pour une surface 7,5 ha de blé (objectif 80 quintaux), la fertilisation a été réalisée en 3 apports d'ammonitrate 27 : 50 kg / ha en février, 90 kg / ha en mars et à nouveau 50 kg / ha en avril.														
1-2 7,5 ha	Blé 80 q 90 q	1	15-févr.	ammo 27	185 kg / ha	1,38 t	270		373	50				
		2	9-mars	ammo 27	330 kg / ha	2,5 t	270		675	90				
		3	20-avr.	ammo 27	185 kg / ha	1,38 t	270		373	50				
		4												
							sous-total	0	1 421					
L'îlot 2 (une seule parcelle) est un pâturage de 2 ha. Elle a reçu en mars un apport de 40 t de purin par ha, dosé à 0,4 kg d'azote (mesuré), en avril un apport de 13/8/24 pour une dose de 45 kg / ha, suivi de deux autres passages avec de l'ammonitrate 27 à la dose de 45 kg / ha														
2-1 2 ha	Pâturage	1	5-mars	Purin	40 000 kg / ha	80 t		32		16	E		0,4	Oui
		2	7-avr.	13/8/24	350 kg / ha	0,7 t	130		90	45				
		3	2-mai	ammo 27	170 kg / ha	0,33 t	270		90	45				
		4	8-juin	ammo 27	170 kg / ha	0,33 t	270		90	45				
							sous-total	32	270					

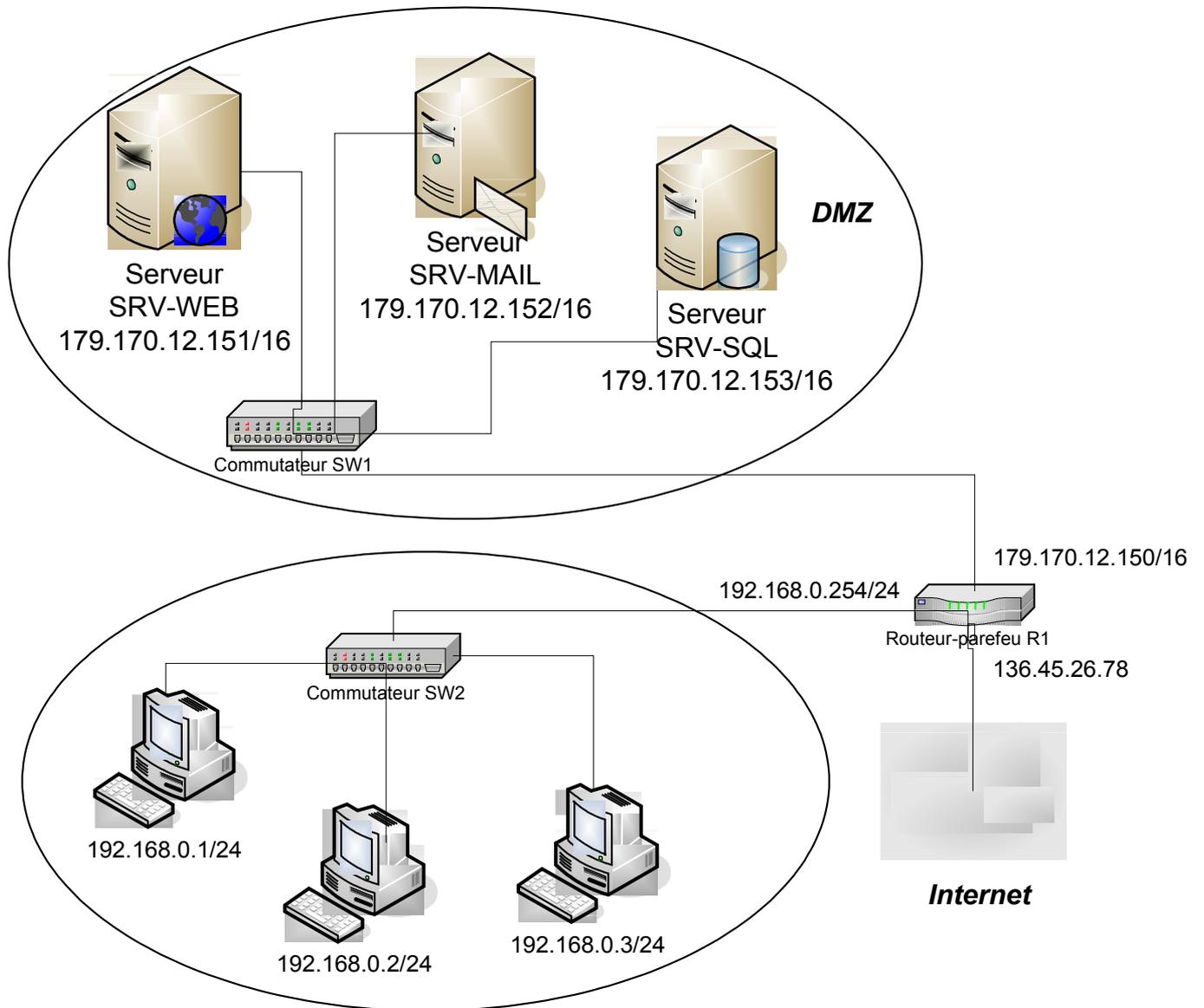
Total des surfaces épandues en ha

organique	minéral
12	33,5

Total des apports d'azote en kg

organique	minéral
1 507	1 966

Annexe 2 : Schéma partiel du site informatique de la Chambre d'Agriculture



Annexe 3 : Extrait du schéma relationnel de la base de données « TraitementPhyto »

Le dossier 3 est indépendant du dossier 1 : l'extrait du schéma relationnel ne représente pas une solution au dossier 1.

PRODUITPHYTOSANITAIRE(id, libelle)

Clé primaire : id

EXPLOITATION(id, nomExploitant, melExploitant)

Clé primaire : id

PARCELLE(id, dateSemis, dateRecoltePrevue, surface, idExploitation, idEspeceCultivee)

Clé primaire : id

Clé étrangère : idExploitation en référence à id de EXPLOITATION

Clé étrangère : idEspeceCultivee en référence à id de ESPECECULTIVEE

TRAITEMENT(id, typeTraitement, dosageTraitementSemence, idParcelle, idProduitPhytosanitaire)

Clé primaire : id

Clé étrangère : idParcelle en référence à id de PARCELLE

Clé étrangère : idProduitPhytosanitaire en référence à id de PRODUITPHYTOSANITAIRE

PULVERISATION(id, datePulverisation, dosage, idTraitement)

Clé primaire : id

Clé étrangère : idTraitement en référence à id de TRAITEMENT

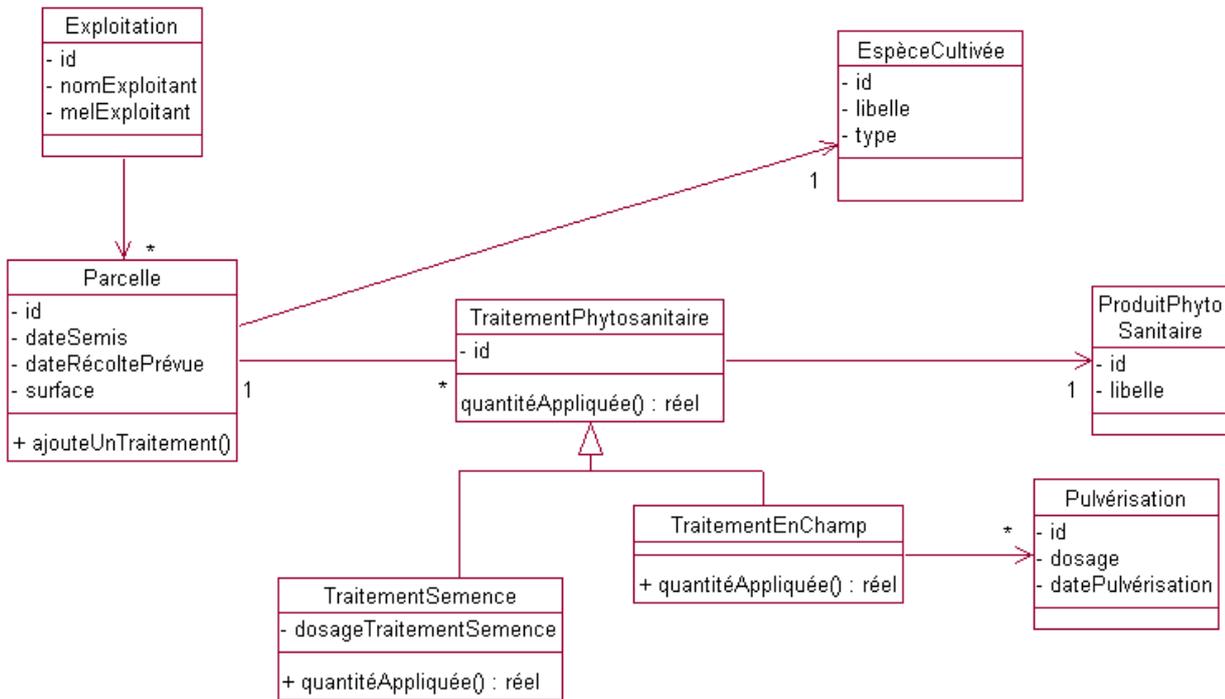
ESPECECULTIVEE(id, libelle, type)

Clé primaire : id

Dans la relation TRAITEMENT :

- Si le traitement est en semence (produit mélangé au semis), le champ *typeTraitement* prend la valeur 's' et le champ *dosageTraitementSemence* prendra une valeur.
- Si le traitement est en champ (plusieurs pulvérisations), le champ *typeTraitement* prend la valeur 'c' et le champ *dosageTraitementSemence* prendra la valeur nulle. Dans ce cas seulement, plusieurs occurrences de la relation PULVERISATION pourront être associées à ce traitement.

Annexe 4 : Diagramme partiel des classes métiers



Remarque : Les constructeurs, accesseurs et paramètres des méthodes ne sont pas présentés sur ce diagramme.

Annexe 5 : Description textuelle des classes métiers

Classe Exploitation

Attributs privés :

id : Chaîne
 nomExploitant : Chaîne
 melExploitant : Chaîne
 lesParcelles : Collection de Parcelle

Méthodes publiques :

Fonction getLesParcelles() : Collection de Parcelle
 // accesseur sur l'attribut privé lesParcelles

FinClasse

Remarque : La classe technique Collection est décrite en annexe 6.

Classe EspèceCultivée

Attributs privés :

id : Chaîne
 libellé : Chaîne
 type : Chaîne

FinClasse

Classe ProduitPhytosanitaire

Attributs privés :
id : Chaîne
libellé : Chaîne

FinClasse

Classe Parcelle

Attributs privés :
id : Chaîne
dateSemis : Date
dateRécoltePrévue : Date
surface : Réel
lEspèceCultivée : Espèce
lesTraitementsPhytosanitaires : Collection de TraitementPhytosanitaire

Méthodes publiques :
Fonction getSurface() : Réel
// accesseur sur l'attribut surface
Fonction getLesTraitementsPhytosanitaires() : Collection de TraitementPhytosanitaire
// accesseur sur l'attribut lesTraitementsPhytosanitaires
Procédure ajouteUnTraitement(unTraitementPhytosan : TraitementPhytosanitaire)
// permet d'ajouter un traitement à la parcelle courante

FinClasse

Classe TraitementPhytosanitaire

Attributs privés :
id : Chaîne
laParcelleCultivée : Parcelle
leProduitPhytosanitaire : ProduitPhytosanitaire

Méthodes publiques :
TraitementPhytosanitaire(unIdTraitementPhytosan : Chaîne, uneParcelleCultivée : Parcelle,
unProduitPhytosanitaire : ProduitPhytosanitaire)
// constructeur de la classe
Fonction abstraite quantitéAppliquée() : Réel
Fonction getLaParcelleCultivée() : Parcelle
// accesseur sur l'attribut laParcelleCultivée

FinClasse

Classe **TraitementSemence** hérite de **TraitementPhytosanitaire**

Attributs privés :

dosageTraitementSemence : Réel // indique la quantité par unité de surface, par ex. 200 (kg/ha)

Méthodes publiques :

TraitementSemence(unIdTraitementPhytosan : Chaîne, uneParcelleCultivée : Parcelle,
unProduitPhytosanitaire : ProduitPhytosanitaire, unDosage : Réel)

// constructeur à écrire

Fonction quantitéAppliquée() : Réel

// la quantité est calculée en effectuant le produit de la surface de la parcelle par le dosage

// du traitement concerné

FinClasse

Remarque : Le constructeur s'utilise de la manière suivante :

Tr : TraitementSemence

Tr ← new TraitementSemence(< arguments du constructeur >)

Classe **TraitementEnChamp** hérite de **TraitementPhytosanitaire**

Attributs privés :

lesPulvérisations : Collection de Pulvérisation

Méthodes publiques :

TraitementEnChamp(unIdTraitementPhytosan : Chaîne, uneParcelleCultivée : Parcelle,
unProduitPhytosanitaire : ProduitPhytosanitaire,
desPulvérisations : Collection de Pulvérisation)

// constructeur à écrire

Fonction quantitéAppliquée() : Réel

// la quantité est calculée en effectuant le cumul des quantités pulvérisées

Fonction getLesPulvérisations() : Collection de Pulvérisation

// accesseur sur l'attribut lesPulvérisations

FinClasse

Classe **Pulvérisation**

Attributs privés :

id : Chaîne

dosage : Réel // indique la quantité par unité de surface, par ex. 200 (kg/ha)

datePulvérisation : Date

Méthodes publiques :

Pulvérisation(unId : Chaîne, uneDate : Date, unDosage : Réel)

// constructeur de la classe Pulvérisation

Fonction getDosage() : Réel

// accesseur sur l'attribut dosage

FinClasse

Remarque : Le constructeur s'utilise de la manière suivante :

pv : Pulvérisation

pv ← new Pulvérisation(<arguments du constructeur>)

Description textuelle des classes techniques

Classe Passerelle

Méthodes publiques à portée classe :

Fonction `getProduit(unIdProduit : Chaîne) : ProduitPhytosanitaire`

// retourne l'objet de la classe ProduitPhytosanitaire dont l'identifiant est passé en paramètre

Fonction `getParcelle(unIdParcelle : Chaîne) : Parcelle`

// retourne l'objet de la classe Parcelle dont l'identifiant est passé en paramètre

Fonction chargerLesPulvérisations(unIdTraitement : Chaîne) : Collection de Pulvérisation

// Instancie et retourne une collection d'objets de la classe Pulvérisation correspondant aux pulvérisations

// concernant le traitement d'identifiant unIdTraitement, à partir des données lues dans la base de données

Fonction chargerLeTraitement(unIdTraitement : Chaîne) : TraitementPhytosanitaire

// Instancie et retourne un objet de la classe TraitementPhytosanitaire correspondant au traitement

// d'identifiant unIdTraitement, à partir des informations lues dans la base de données

FinClasse

Classe Collection de <nom de la classe>

Méthodes publiques

Fonction `cardinal() : Entier`

// Renvoie le nombre d'objets de la collection

Fonction `obtenirObjet(unIndex : Entier) : Objet de la classe`

// Retourne l'objet d'index unIndex, le premier objet de la collection a pour index 1

Procédure `ajouter(unObjet : Objet de la classe)`

// Ajoute un objet à la collection

FinClasse

Pour instancier une collection :

`uneCollection : Collection de <classe>`

`uneCollection ← new Collection() de <classe>`

Pour parcourir par itération les éléments d'un objet Collection

Pour chaque <objet> dans <collection> faire

// instructions avec <objet>

`FinPour`

Classe JeuEnregistrements

Méthodes Publiques

`JeuEnregistrements(chaîneSQL : Chaîne)` *// Constructeur, positionne le curseur sur le premier
// enregistrement*

Procédure `suivant()`

// avance le curseur sur l'enregistrement suivant

Fonction `fin() : Booléen`

// indique si la marque de fin est atteinte

Fonction `getValeur(nomChamp : Chaîne) : Variant`

// renvoie la valeur du champ nomChamp de l'enregistrement courant. Variant est un type

//générique pouvant contenir tout type de valeur. On peut utiliser l'opérateur « + » pour concaténer

// des valeurs ou variables de type chaîne ou variant.

Procédure `fermer()`

// ferme le curseur et libère les ressources

FinClasse

Remarque : Le constructeur s'utilise de la manière suivante :

`jeu : JeuEnregistrements`

`jeu ← new JeuEnregistrements(<chaîne SQL >)`

Annexe 7 : Algorithme de la méthode chargerLeTraitement() de la classe Passerelle

```
Publique Statique Fonction Passerelle
:: chargerLeTraitement(unIdTraitement : Chaîne) : TraitementPhytosanitaire

Var
    texteReq : Chaîne
    jeuTraitements : JeuEnregistrements
    leProduit : ProduitPhytosanitaire
    laParcelle : Parcelle
    traitementAReturner : TraitementPhytosanitaire // objet à retourner

Début

    traitementAReturner ← null

    texteReq ← " Select * From Traitement tp Where tp.id = '"
    texteReq ← texteReq + unIdTraitement + "'"

    jeuTraitements ← new JeuEnregistrements(texteReq)
        // Exécution de la requête SQL

    si NON jeuTraitements.fin() alors
        // Il y a un traitement associé

        leProduit ← getProduit(jeuTraitements.getValeur("idProduitPhytosanitaire"))
        laParcelle ← getParcelle(jeuTraitements.getValeur("idParcelle"))

        -- Partie à compléter sur la copie (cf. question 3.9) --

        // instructions créant un objet de la classe TraitementSemence
        // ou de la classe TraitementEnChamp, retourné par la fonction

    fsi

    jeuTraitements.fermer()

    retourner traitementAReturner // objet retourné

Fin
```