# Architecture et conception détaillée

|  |
| --- |
|  |
|  |
| **9991af5552eee3d809694a30c5d05344**  **Plateforme de valorisation des produits agricoles de la province de Québec (ProdAgriviQ)** |
| Version 1.5 |
|  |
| **Judicaël OUIKON DONGONGO**  **Noreddine Ben Jillali** |
| 29 février 2016 |

**Historique des révisions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date**  (jj-mm-aaaa) | **Version** | **Description** | **Auteur** |
| 17-02-2016 | 1.0 | Introduction et architectures | Judicaël OUIKON DONGONGO  Noreddine Ben Jillali |
| 21-02-2016 | 1.2 | Conception détaillée | Judicaël OUIKON DONGONGO  Noreddine Ben Jillali |
| 25-02-2016 | 1.3 | Diagrammes | Judicaël OUIKON DONGONGO  Noreddine Ben Jillali |
| 26-02-2016 | 1.4 | Lecture et correction | Noreddine Ben Jillali |
| 27-02-2016 | 1.4.1 | Relecture et correction | Judicaël OUIKON DONGONGO |
| 29-02-2016 | 1.5 | Mise à jour et des diagrammes de ProdAgriviQ | Judicaël OUIKON DONGONGO  Noreddine Ben Jillali |

# ****Définitions****

|  |  |
| --- | --- |
| **Terme** | **Définition** |
| Entity | stéréotype des classes du modèle (**M**VC) [8] [9] |
| layer | stéréotype des composants de l'application ProdAgriviQ [8] [9] |
| Controler | stéréotype des classes du contrôleur (MV**C**) [8] [9] |
| IU | stéréotype des classes de la vue (M**V**C) [8] [9] |

# ****Abréviations/acronymes****

|  |  |
| --- | --- |
| **Abréviation / Acronyme** | **Définition** |
| PA | produit agricole |
| VPA | variété de produit agricole |
| ProdAgriviQ | Plateforme de valorisation des produits agricoles de la province du Québec |
| QC | Québec |
| UML | Unified Modeling Language |
| MVC | Modèle Vue Contrôleur |
| UI | User Interface |
| SRS | Software Requirement Specification |
| SGBD | Système de gestion de base de données |
| TCP/IP | Transmission Control Protocol / Internet Protocol |
| HTTP | Hypertext Transfer Protocol |
| JDBC | Java Database Connectivity |

Table des matières

|  |  |
| --- | --- |
| [1. Introduction](#ArchitectureetconceptiondétailléeProdAg)  [1.1. Objectif](#ArchitectureetconceptiondétailléeProdAg)  [1.2. Portée](#ArchitectureetconceptiondétailléeProdAg)  [1.3. Références](#ArchitectureetconceptiondétailléeProdAg)  [2. Partie prenantes de la conception et leurs préoccupations](#ArchitectureetconceptiondétailléeProdAg)  [2.1. Partie prenantes de la conception et leurs préoccupations](#ArchitectureetconceptiondétailléeProdAg)  [2.2. Vues de conception et liens avec les préoccupations de conception](#ArchitectureetconceptiondétailléeProdAg)  [3. Architecture logicielle](#ArchitectureetconceptiondétailléeProdAg)  [3.1. Vue d’ensemble de l’architecture logicielle](#ArchitectureetconceptiondétailléeProdAg)  [3.2. Vue d’architecture logique de ProdAgriviQ](#ArchitectureetconceptiondétailléeProdAg)  [3.3. Vue d’architecture physique de ProdAgriviQ](#ArchitectureetconceptiondétailléeProdAg)  [4.    Conception détaillée](#ArchitectureetconceptiondétailléeProdAg)  [4.1. Vue d’ensemble de la conception détaillée](#ArchitectureetconceptiondétailléeProdAg)  [4.2. Conception détaillé de l’élément 1 : Gestion de variétés de produits agricoles et de récoltes saisonnières](#ArchitectureetconceptiondétailléeProdAg)  [4.3. Conception détaillé de l’élément 2 : Inscription Fermier](#ArchitectureetconceptiondétailléeProdAg)  [4.4. Conception détaillé de l’élément 3 : Rechercher produit agricole](#ArchitectureetconceptiondétailléeProdAg) | 5  5  5  5  7  7  8  8  8  8  12  13  13  14  24  27 |

# 1. Introduction

## 1.1. Objectif

Le but du présent document est de fournir une description de la conception et de l’architecture logicielle de ProdAgriviQ en se basant sur les exigences techniques (versus fonctionnelles) fournies dans le document des spécifications du logiciel (SRS) [7]. Afin d'illustrer ces interactions, ce document contient la conception détaillée des éléments du logiciel ainsi que différentes vues représentant l’architecture logicielle.

## 1.2. Portée

ProdAgriviQ est une application Web  qui sera conçu et réalisé dans le cadre du cours de la vérification et assurance qualité de la maîtrise. Dues aux contraintes de délai (au plus 13 semaines), seuls 4 sous-systèmes de l’application seront conçus et implémentés. Le projet fait l’objet d'un contrat entre deux étudiants de ce cours.

À cette ère du Web, il n’existe pas une plateforme centralisée destinée à tous les fermiers pour valoriser les produits agricoles de la province de Québec. ProdAgriviQ sera développée et mise gratuitement à la disposition des fermiers et entreprises agroalimentaires du Québec.   Le but  ultime  est  de  mettre  à  la disposition  du  public  une  vue générale de l’ensemble des produits agricoles (PA) des fermiers du Québec.

L'ensemble des activités du présent projet seront faites selon le profil basique de la norme ISO/CEI 2911 0 [8]. Cependant, l'étude des exigences ont été précédemment réalisées [7].

## 1.3. Références

### 1.3.1 Références

**Supports de cours**

[1] Claude Y Laporte, " Support de cours MGL805 - Vérification et assurance qualité de logiciels ", Hiver 2016

**[2]** Judicaël OUIKON DONGONGO et Noreddine Ben Jillali, **« Plan du ProjetH – ProdAgriviQ**», février 2016

**[3]** Judicaël OUIKON DONGONGO et Noreddine Ben Jillali, **« Contrat entre coéquipier – ProdAgriviQ** », février 2016

[4] Judicaël OUIKON DONGONGO et Noreddine Ben Jillali,  " Plateforme Web de valorisation des Produits Agricoles vivriers du Québec (ProdAgriviQ)", février 2016

[5] Judicaël OUIKON DONGONGO et Noreddine Ben Jillali,  " Composants logiciels - ProdAgriviQ", février 2016

**Document de vision**

**[6] Judicaël OUIKON-DONGONGO, et Gildas Victorien KABIWA TAGNE, « Document de vision de la Plateforme** Web de valorisation des Produits Agricoles vivriers du Québec (ProdAgriviQ)»,  version 1.5.1 du 13  octobre  2015

**Document de spécification des exigences**

[7]  Judicaël OUIKON-DONGONGO, et Gildas Victorien KABIWA TAGNE, « SRS de la Plateforme Web de valorisation des Produits Agricoles vivriers du Québec (ProdAgriviQ)», version 2.1 du 1 er décembre 2015

**Ouvrages spécifiques**

[8] Tilak Mitra, "Practical Software Architecture: Moving from System Context to Deployment (IBM Press) Kindle Edition, November 2015

[9] [Hassan Gomaa](http://www.informit.com/authors/author_bio.aspx?ISBN=9780201775952) , Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures, July 2004

[10] Jérôme Lafosse, Développements n-tiers avec Java EE (Architectures, GlassFish, JSF, JPA, JWS, EJB, JMS, SOAP, REST), ENI Edition, 2011

### 1.3.2 Références normatives

[8] ISO/CEI TR 29110-5-1-2:2011- RAPPORT TECHNIQUE

     Guide de gestion et d'ingénierie - Profil Basique Ingénierie du logiciel

     — Profils de cycle de vie pour très petits organismes (TPO) —

    Partie 5-1-2: Guide de gestion et d'ingénierie: Groupe de profils génériques:

Profil basique

# 2. Partie prenantes de la conception et leurs préoccupations

## 2.1. Partie prenantes de la conception et leurs préoccupations

Pour cette activité de conception, les parties prenantes sont :

* Noreddine, développeur et membre de l'équipe du projet;
* Judicaël, concepteur et chef de l'équipe du projet.

La préoccupation du développeur de l'application ProdAgriviQ est l’influence des pratiques de développement et les technologies utilisées lors de la réalisation des activités d'architecture et de conception. L'objectif principal est de séparer toute technologie de la mise en œuvre de la conception de l'application ProdAgriviQ.

Quant au concepteur de l'application ProdAgriviQ, sa préoccupation majeure est la place à accorder à UML dans la réalisation des tâches de cette activité. En effet, ce dernier souhaite que l'équipe utilise les artefacts de l'UML pour modéliser les architectures et autres concepts de la conception de l'application et ceci dans le but d'avoir une conception bien visuelle, bien construite et bien documentée. Un autre point soulevé par Judicaël est la prise en compte des exigences et contraintes techniques ou de conception telles qu’elles sont présentés dans le SRS. Il propose aussi de distinguer les cas d’utilisation fonctionnels des cas d’utilisation techniques. Pour le concepteur, les cas d’utilisation fonctionnels n’ont pas vraiment leur place dans l’architecture et la conception car ceux-ci sont largement décrits dans le SRS [5]. Seul les cas d’utilisation techniques découlant des besoins techniques tels l’accès aux données, gérer la concurrence et bien d’autre encore seront présentés les diagrammes de cas d’utilisation de ce document.

## 2.2. Vues de conception et liens avec les préoccupations de

## conception

Il s'agit dans cette étape de notre plan de projet de :

* développer le niveau d’abstraction que nous souhaitons atteindre. Ce niveau d'abstraction comporte des classes réutilisables qui sont en quelque sorte le socle de ProdAgriviQ ;
* appliquer les éléments de cette abstraction aux fonctionnalités de composants de ProdAgriviQ que nous avons considérés;

Nous allons donc concevoir la solution correspondant aux spécifications et contraintes techniques ainsi que fonctionnelles. Ces spécifications et contraintes proviennent à la fois du SRS de ProdAgriviQ d'une part et des préoccupations précédemment listées d'autre part. Le but est de concevoir une application Web maintenable et évolutive même à l'absence de l'équipe actuelle de conception ou de projet.

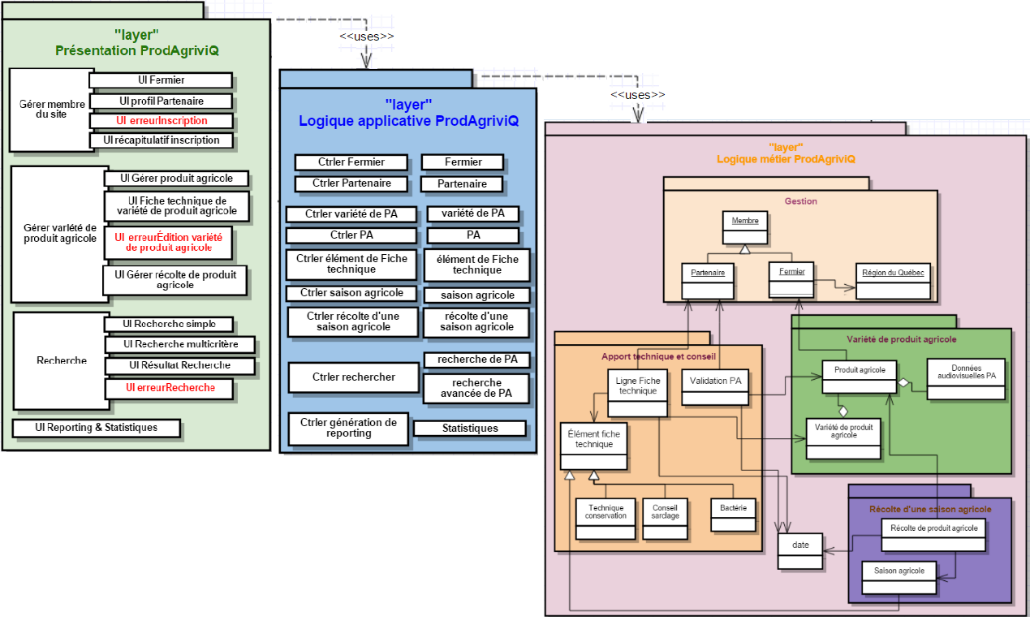
# 3.    Architecture logicielle

## 3.1. Vue d’ensemble de l’architecture logicielle

Dans cette section, les sous-systèmes de ProdAgriviQ listés dans l’annexe A du plan de projet [2] ainsi que leur dépendance sont identifiés et décrit suivant le paradigme MVC. Nous avons utilisé le modèle en couche qui est représenté en UML par le diagramme de package (architecture interne) et le diagramme de déploiement (architecture physique ou d'exploitation).

## 3.2. Vue d’architecture logique de ProdAgriviQ

### 3.2.1.     Vue d’ensemble



*Figure 1 : Architecture en couche des modules de ProdAgriviQ considérés pour ce projet de session*

### 3.2.2.     Contraintes de conception qui s’applique à cette vue

Nombreuses contraintes s’appliquent à cet architecture dont les plus « critiques » sont les contraintes techniques qui découlent des exigences non fonctionnelles (ENF) décrites dans le SRS [7]. Ces contraintes sont liées aux performances d’accès aux données, la sécurité du système, l’immensité ou la volumétrie de données, l’hétérogénéité des données (audiovisuelles et textuelles) et la disponibilité de l’application.

### 3.2.3.     Exigences et préoccupations de conception

Dans [5] nous avons décrit les composants de ProdAgriviQ et élaboré son diagramme de déploiement. Ces différents composants d’exploitation se retrouvent dans chacun des sous-systèmes ou modules de cette application web. Les exigences de la conception sont celles d’une architecture à composants qui sont les suivants :

* la base de données de ProdAgriviQ aura au moins deux instances dont une en lecture seule. On a par ailleurs retenu un moteur de base de données relationnel;
* chaque module de ProdAgriviQ à savoir la gestion des variétés de produits agricoles, l’inscription des Fermiers, recherche de PA et gestion des récoltes de PA d’une saison doit avoir un composant « noyau applicatif » [5] (contrôleur) et un composant « accès aux données » [5] (DAO) qui lui est propre ;
* Les technologies de Java Enterprise Edition seront utilisées pour développer composants « noyau applicatif » ou contrôleur, « sécurité », « authentification » et « accès aux données » ;
* Le composant « instance de base de données » sera implémenté en mySQL dans le cadre de ce projet de session.

### La principale préoccupation de conception concerne la configuration matérielle qui pourrait supporter l’exploitation de ProdAgriviQ. À cette ère de Web et pour une province aussi grande comme le Québec, quelle serait la configuration matérielle la plus adaptée ? Solliciter le service des fournisseurs de cloud serait-il la bonne réponse à cette préoccupation ? Quelle serait l’attitude des Fermiers et d’autres membres du site si la gestion de leurs données serait confiée faite par le service des fournisseurs de cloud qu’ils ne connaissent pas ?

### 3.2.4.     Description des éléments de la vue et leurs interfaces

Dans la vue, l'on retrouve les classes qui ne sont rien d'autre que les interfaces graphiques qui permettront aux utilisateurs d'atteindre leur fin avec l'application. Ces classes sont regroupées par composant logiciel de ProdAgriviQ tel que présentées dans le tableau suivant :

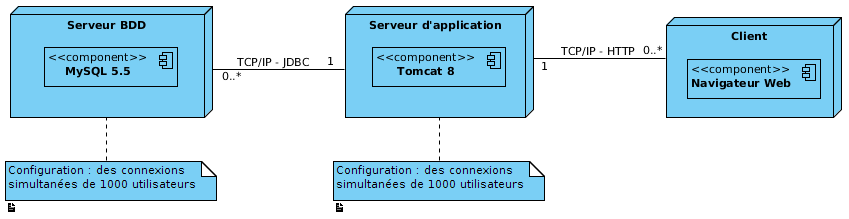
|  |  |
| --- | --- |
| **Composant de ProdAgriviQ** | **Élément de la vue** |
| Gérer membre du site | UI Fermier |
| UI récapitulatif inscription |
| UI erreur inscription |
| UI profil Partenaire |
| Gérer variété de produit agricole et récolte saisonnière | UI Gérer produit agricole |
| UI Fiche technique d'une variété de produit agricole |
| UI Fiche technique d'un produit agricole |
| UI Gérer les récoltes d'une saison agricole |
| UI erreur édition d'une variété de produit agricole |
| Recherche de produit agricole | UI recherche simple |
| UI recherche multicritère ou recherche avancée |
| UI Résultat de recherche |
| UI erreur de recherche |
| Statistiques | UI Reporting et statistiques |

### 3.2.5.     Raisonnement

Dans cette architecture logique de ProdAgriviQ, nous avons procédé par un regroupement des classes ayant de sémantique proche pour obtenir nos composants ainsi que les grands modules de ProdAgriviQ. Pour chaque module, nous avons défini  les classes de modèle, de vue et de contrôleur. L'on remarquera que les classes de vue utiliseront ou communiqueront avec les classes de contrôleur qui à leur tour communiqueront avec celles du modèle. Les classes de même niveau peuvent toutefois communiquer entre elles. Cependant, en ce qui concerne les classes du modèle autrement dit les classes de la couche ou layer Logique métier de ProdAgriviQ, nous avons distingué les classes qui sont réellement stables telles que Variété de produit agricole et produit agricole des classes susceptibles d'évoluer dans le temps, c’est le cas de la classe récolte de produit agricole.

## 3.3. Vue d’architecture physique de ProdAgriviQ

### 3.3.1.     Vue d’ensemble



*Figure 2 : Architecture physique de ProdAgriviQ considérés pour ce projet de session*

### 3.3.2.     Contraintes de conception qui s’applique à cette vue [7]

* **PAQ-CC-005** : Le modèle de données doit être flexible afin de faciliter son implémentation en utilisant à la fois les gestionnaires de base de données open source (MySQL).
* **PAQ-CC-006** : Aux phases de conception, de développement et de tests, le gestionnaire de base de données open source MySQL doit être utilisé pour le stockage et la  gestion des données ainsi que le requêtage.
* **PAQ-CC-007** : Le gestionnaire de base de données open source, MySQL doit être utilisé en production.
* **PAQ-CC-008** : Le protocole HTTP sera utilisé pour toute communication entre l’utilisateur et l’application Web ProdAgriviQ.
* **PAQ-CC-009** : Aucune communication directe permise entre ProdAgriviQ et toute autre application.

### 3.3.4.     Description des éléments de la vue et leurs interfaces

ProdAgriviQ adoptera une architecture multi-tiers [7] afin de séparer les grandes composantes qui s’interfaceront ensemble pour faire fonctionner le système. Cette application comportera 3 tiers :

* Le client : est un navigateur web qui s'en chargera de l'affichage des différentes pages web de ProdAgriviQ et qui communiquera avec l'application en se basant sur le protocole HTTP.
* Le serveur d'application : un serveur tomcat dans lequel l'application web ProdAgriviQ sera déployée, ce serveur permettra l'accès à un nombre important d'utilisateur simultanément et utilisera un serveur de base de données pour le stockage de données.
* Le serveur de base de données relationnelles : Dans un premier temps nous avons opté pour un système de gestion de base de données relationnelles pour le stockage des données de l'application ProdAgriviQ, ce serveur sera de type MySQL et communiquera à l'aide d'une interface JDBC avec le serveur d'application via le protocole TCP/IP.

### 3.3.5.     Raisonnement

Nous avons choisi cette architecture afin de séparer les grandes parties de notre système et pour avoir la possibilité de changer le type de serveur et SGBD en fonction de l'évolution de l'application et la quantité de données et d'accès gérés par ProdAgriviQ. Ainsi qu'une telle architecture nous offre une grande capacité d'évolution et une aisance lors de la maintenance.

# 4.    Conception détaillée

## 4.1. Vue d’ensemble de la conception détaillée

Dans les sections qui vont suivre, nous allons présenter les diagrammes de classes ainsi que ceux de séquence de chacun des composants de ProdAgriviQ qui seront implémentés. Nous avons réuni la gestion de variétés de produit agricole ainsi que celle de récolte saisonnière en un seul composant. Par respect du paradigme MVC, les classes de la Vue, ont un fond vert, celles de Contrôleur ont quant à elle un fond blanc et les classes du Modèle sont en jaune.

## 4.2. Conception détaillé de l’élément 1 : Gestion de variétés de

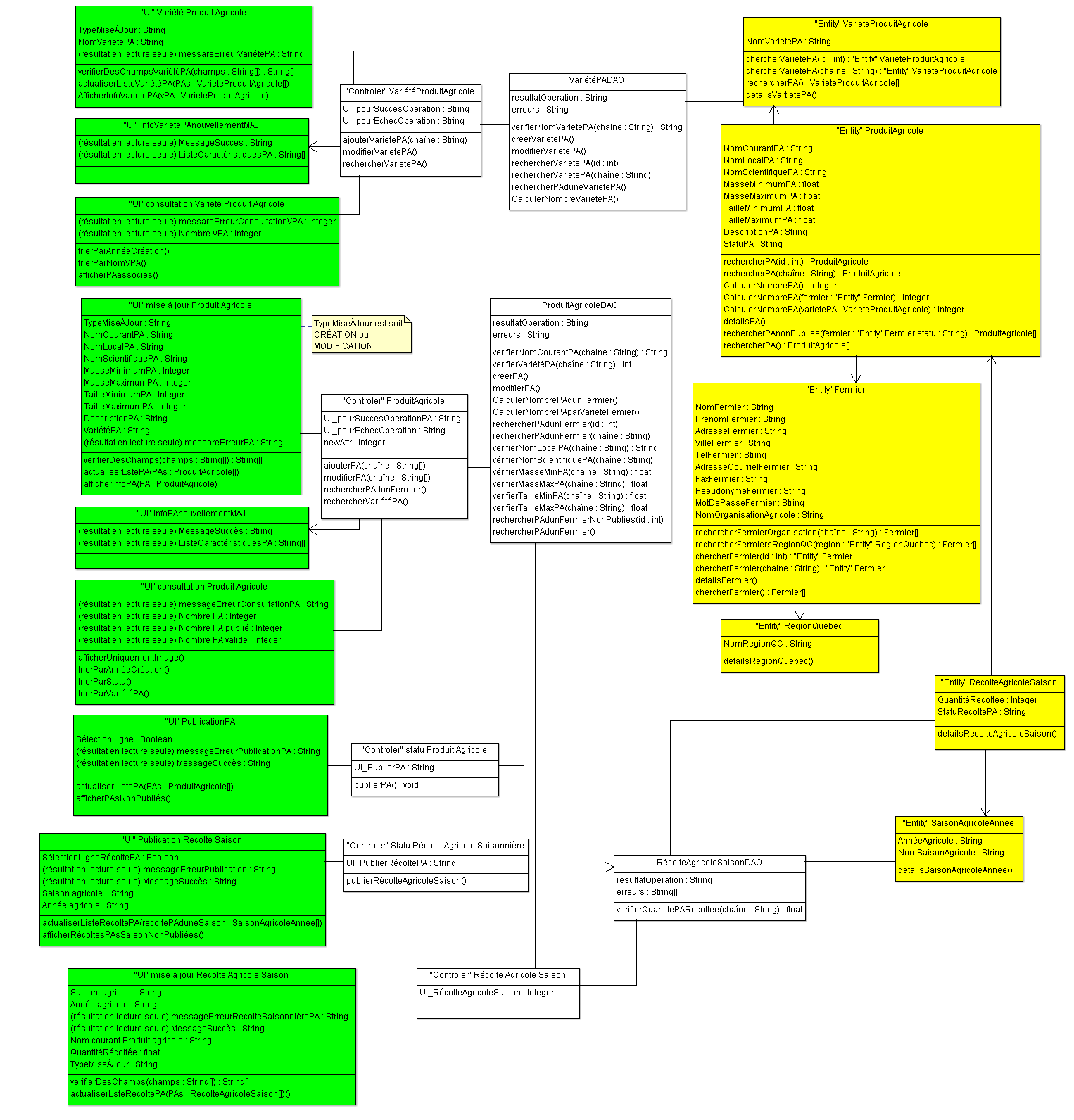
## produits agricoles et de récoltes saisonnières

Dans le cadre de ce projet de session, la gestion de variétés de produits agricoles couplée avec les récoltes agricoles d'une saison représente le principal module de ProdAgriviQ. Dans les sections qui suivent, nous présenterons les diagrammes de classes, de cas d’utilisation technique et fonctionnelle ainsi que celui de séquences des différents modules de cette application web. Le diagramme d'état de PA sera aussi présenté.

En ce qui concerne les cas d’utilisation techniques, ils ne sont pas spécifiques à un module ou à une application moderne spécifique. Ce sont en effet, des problèmes et contraintes qui concernent généralement tous types de logiciels et de systèmes. À cet effet, un seul diagramme sera établi et présenté à la section consacrée à la conception détaillée de la gestion des variétés de PA et de récoltes agricoles d’une saison. Cependant, ce même diagramme s’applique aussi aux autres modules de l’application.

### 4.2.1.     Vue structurelle

Les couleurs de fond des classes du diagramme ci-après permettent de distinguer les classes de vue (vert) de celles de contrôleur (blanc) et du modèle (jaune). Les classes de la vue accèdent aux données des entités à travers les classes de contrôleur.



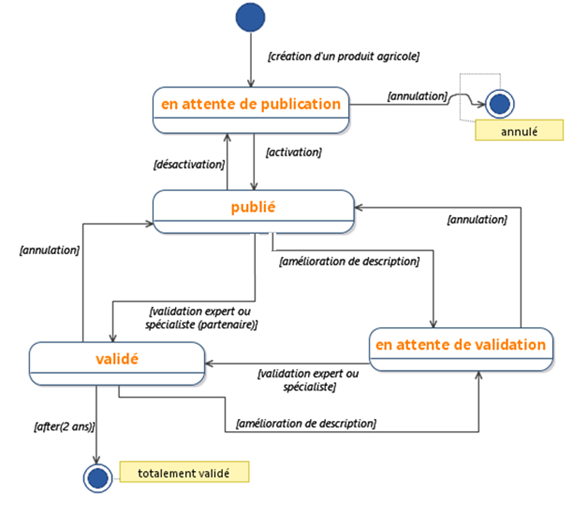
*Figure 3 : diagramme de classes du module gestion des variétés des produits agricoles et*

*récoltes d’une saison*

### 4.2.2.     Vue comportementale

**4.2.2.1. Diagramme d'états d'un objet "Produit Agricole"**

Le cycle de vie d'un objet PA est décrit par le diagramme d’états suivant :

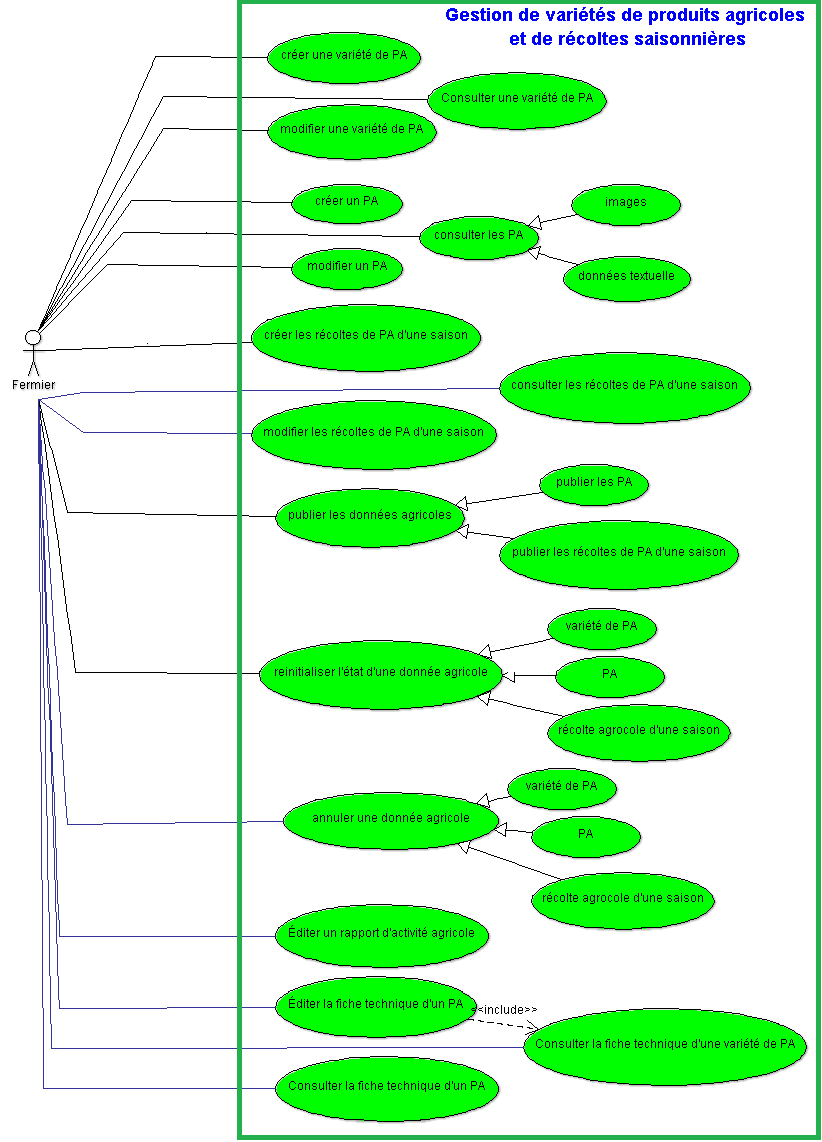


*Figure 4 : diagramme d’états d’une instance de Produit agricole*

Source : diagramme d’état de l’objet PA, SRS de ProdAgriviQ,version 2.1 du 1er Décembre 2015.

**4.2.2.2. Diagramme de cas d'utilisation de la Gestion de variété de PA et de**

**récoltes agricoles**

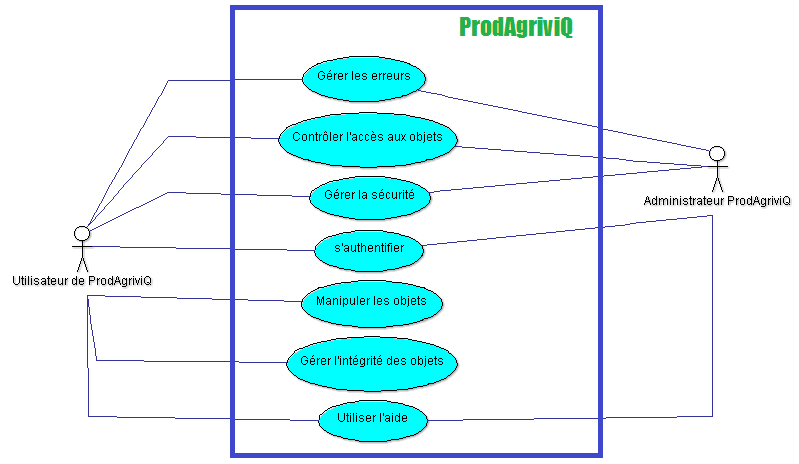
En phase d'analyse de ProdAgriviQ [7], l'analyste a défini et décrit les cas d'utilisation fonctionnels "gérer les variété de PA" et "gérer les récoltes de PA". Ce sont des cas d’utilisation fonctionnels composés. Nous proposons dans cette activité de conception, une décomposition de ces précédents cas d'utilisation ainsi qu'une harmonisation d'autres cas d'utilisation de l'analyse. Le résultat de ces diverses tâches est présenté sur le diagramme de cas d'utilisation suivant :

*Figure 5 : diagramme de cas d’utilisation fonctionnels de la gestion des variétés des PA*

En conception, le concepteur élabore les cas d’utilisation techniques qui sont des fonctionnalités techniques soutenant les fonctionnalités fonctionnelles vu en analyse.

Les acteurs qui bénéficient de ces fonctionnalités techniques du ProdAgriviQ sont pratiquement les mêmes que ceux de l’analyse que nous ajoutons l’administrateur. Ces acteurs sont des exploitants de cette application web. L’administrateur s’occupe de toutes les tâches de configuration, de déploiement et de supports d’une manière générale.

Le diagramme de cas d’utilisation techniques de ProdAgriviQ est le suivant :

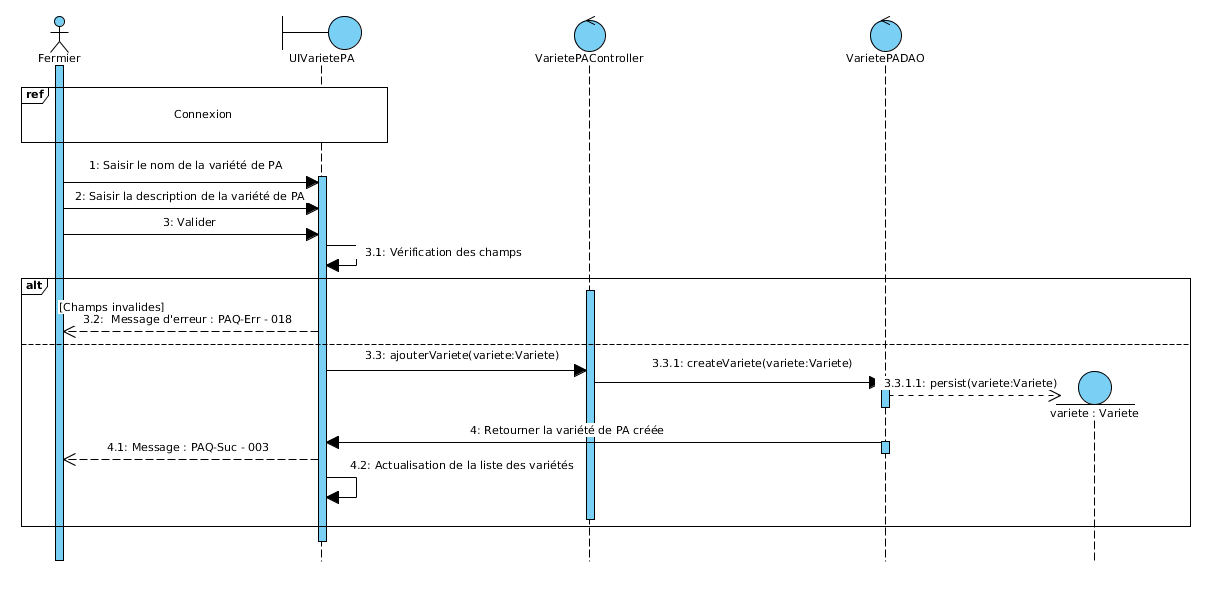


*Figure 6 : diagramme de cas d’utilisation techniques de la gestion des variétés des PA*

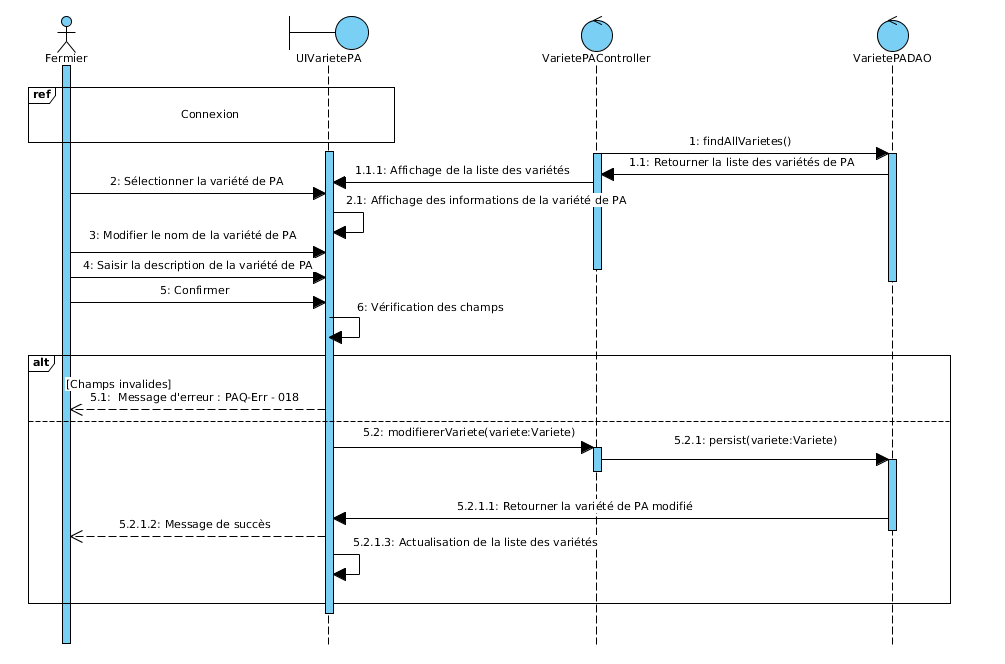
### 4.2.3.     Autres vues pertinentes

Dans chacun des diagrammes de séquence des sections suivantes, les interactions entre les instances de vue, de contrôleur et de modèle sont mises en évidence.  Ces diagrammes montrent aussi qu'il n'y a aucune interaction entre l'interface utilisateur (UI ou dialogue ou encore page web) et les objets de la base de données de ProdAgriviQ et ce par respect du paradigme MVC.

**4.2.3.1. Diagramme de séquence de création d'une variété de PA**

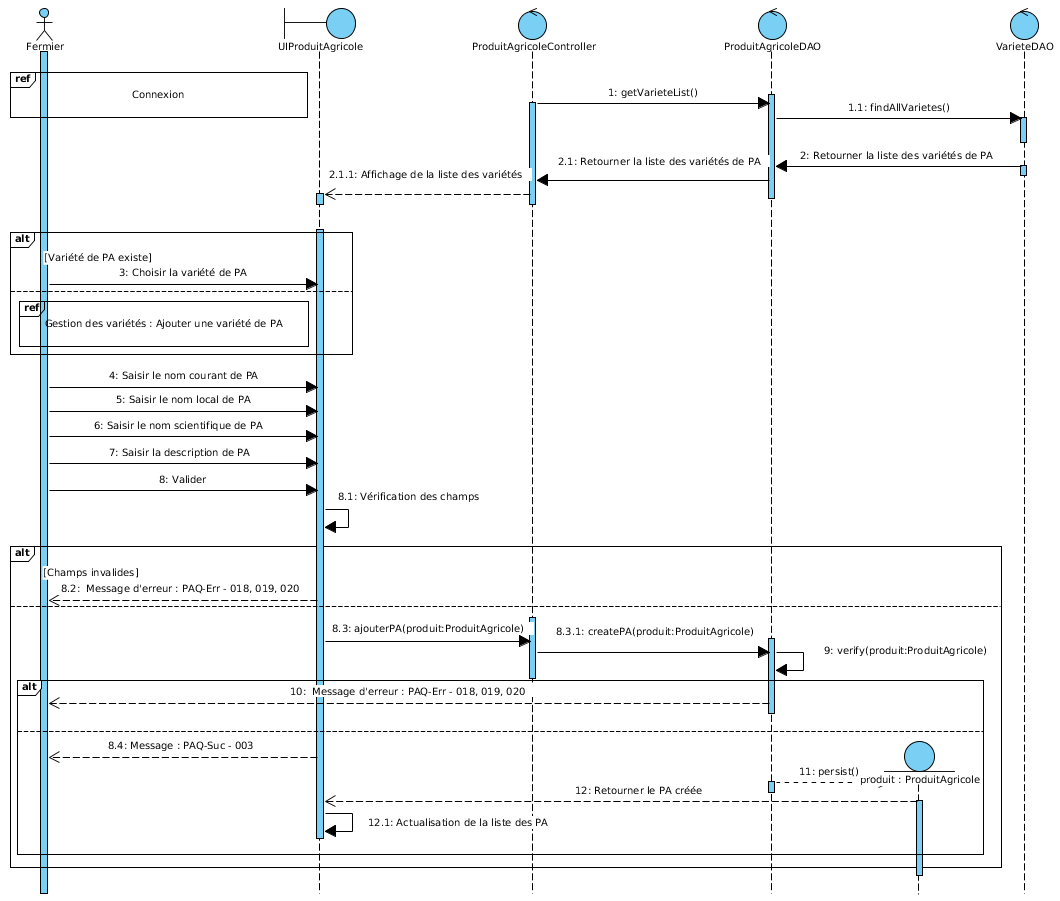
*Figure 7 : diagramme de séquence de création d’une variété des PA*

**4.2.3.2. Diagramme de séquence de modification d'une variété de PA**



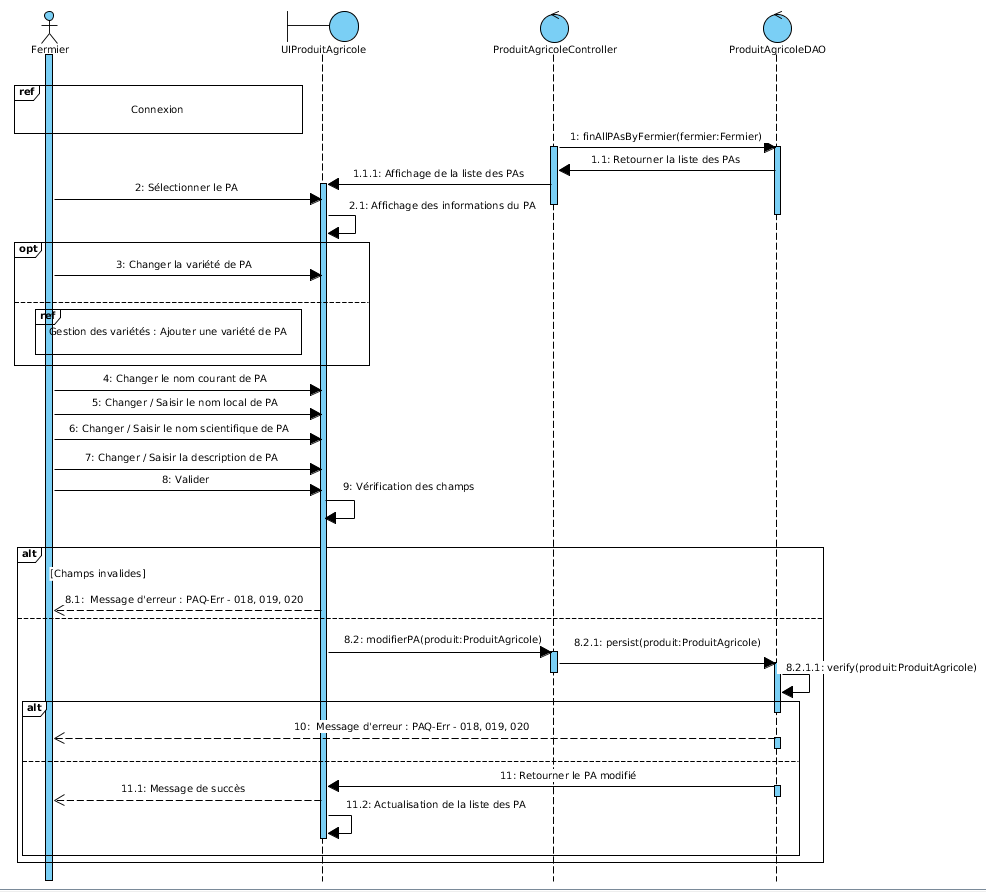
*Figure 8 : diagramme de séquence de modification d’une variété des PA*

**4.2.3.3. Diagramme de séquence de création d'un PA**



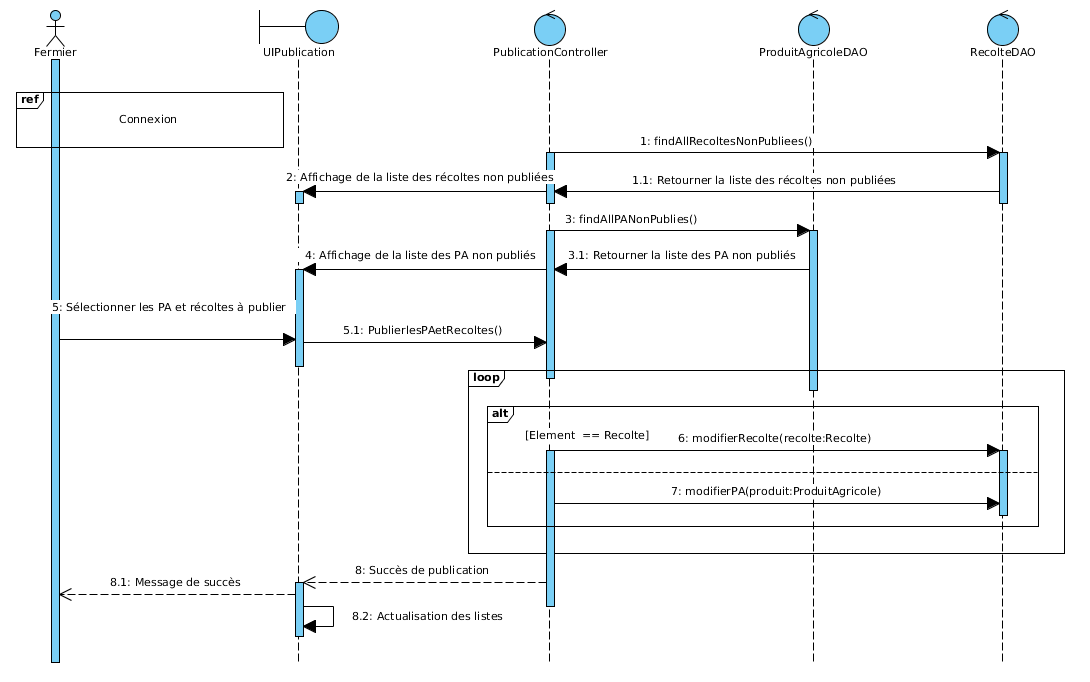
*Figure 9 : diagramme de séquence de création d’un PA*

**4.2.3.4. Diagramme de séquence de modification d'un PA**

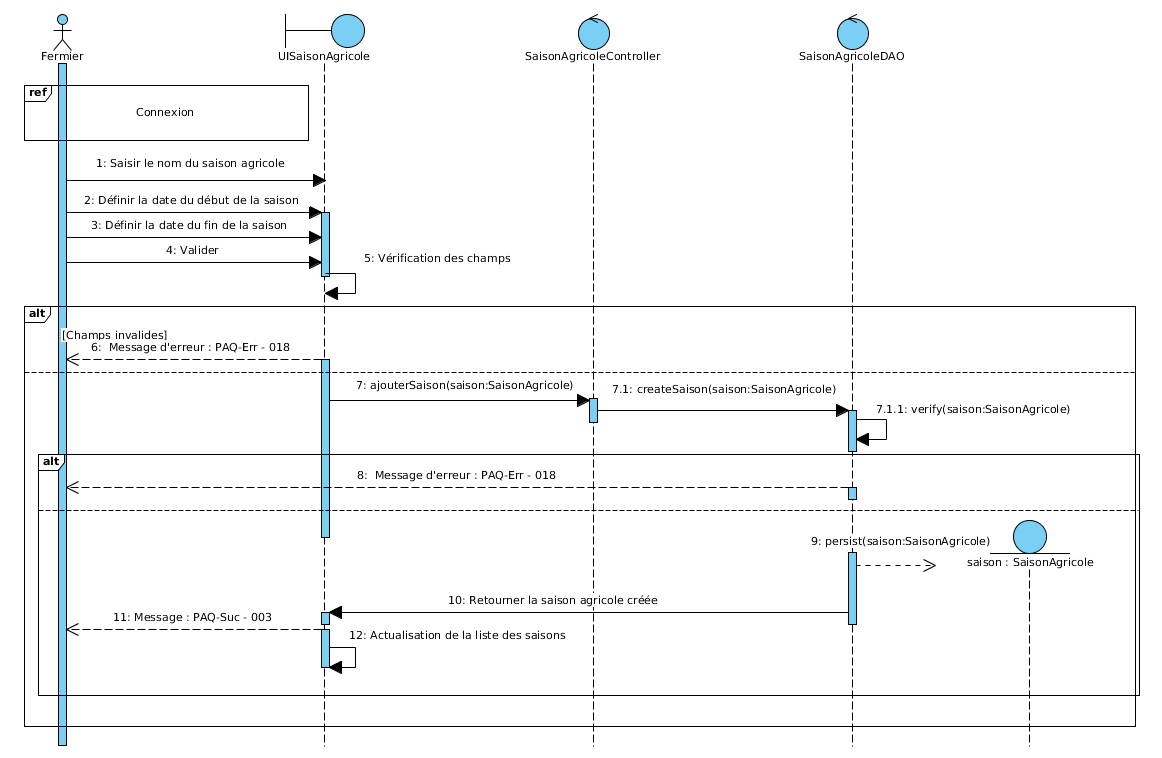


*Figure 10 : diagramme de séquence de modification d’un PA*

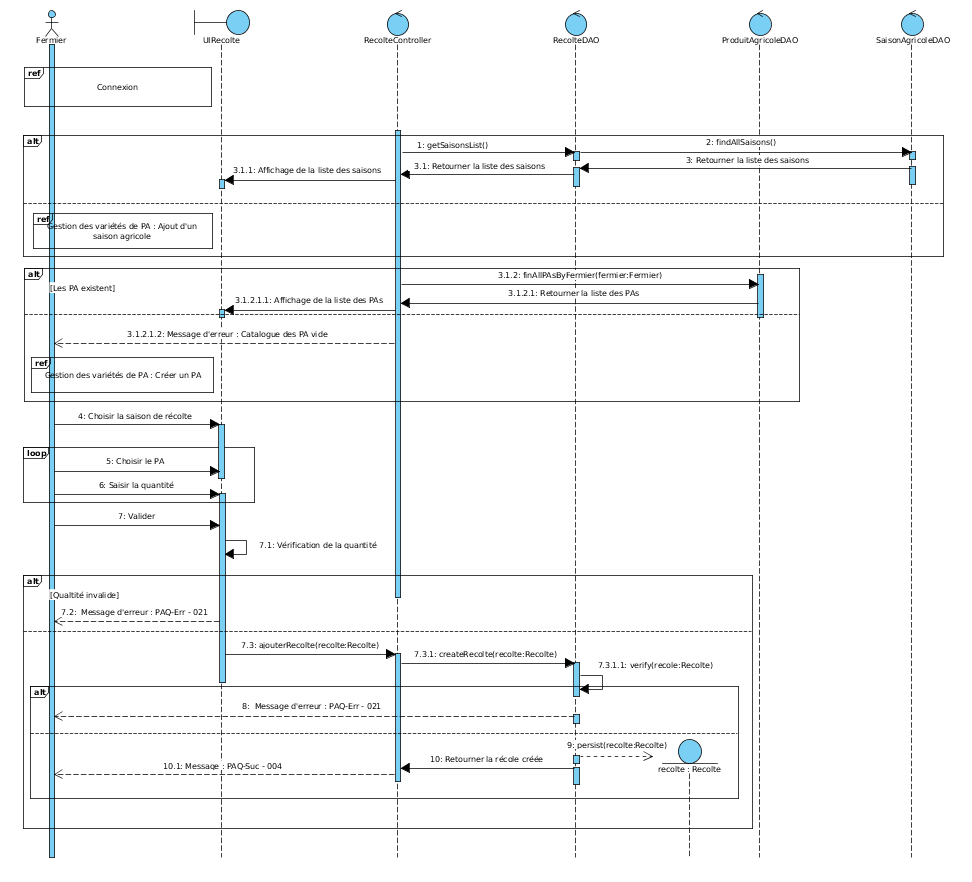
**4.2.3.5. Diagramme de séquence de publication d'un PA**

*Figure 11 : diagramme de séquence de publication de PA*

**4.2.3.6. Diagramme de séquence de création d'une saison agricole**

*Figure 12 : diagramme de séquence de création d’une saison agricole*

**4.2.3.7. Diagramme de séquence de création d'une récolte de PA**

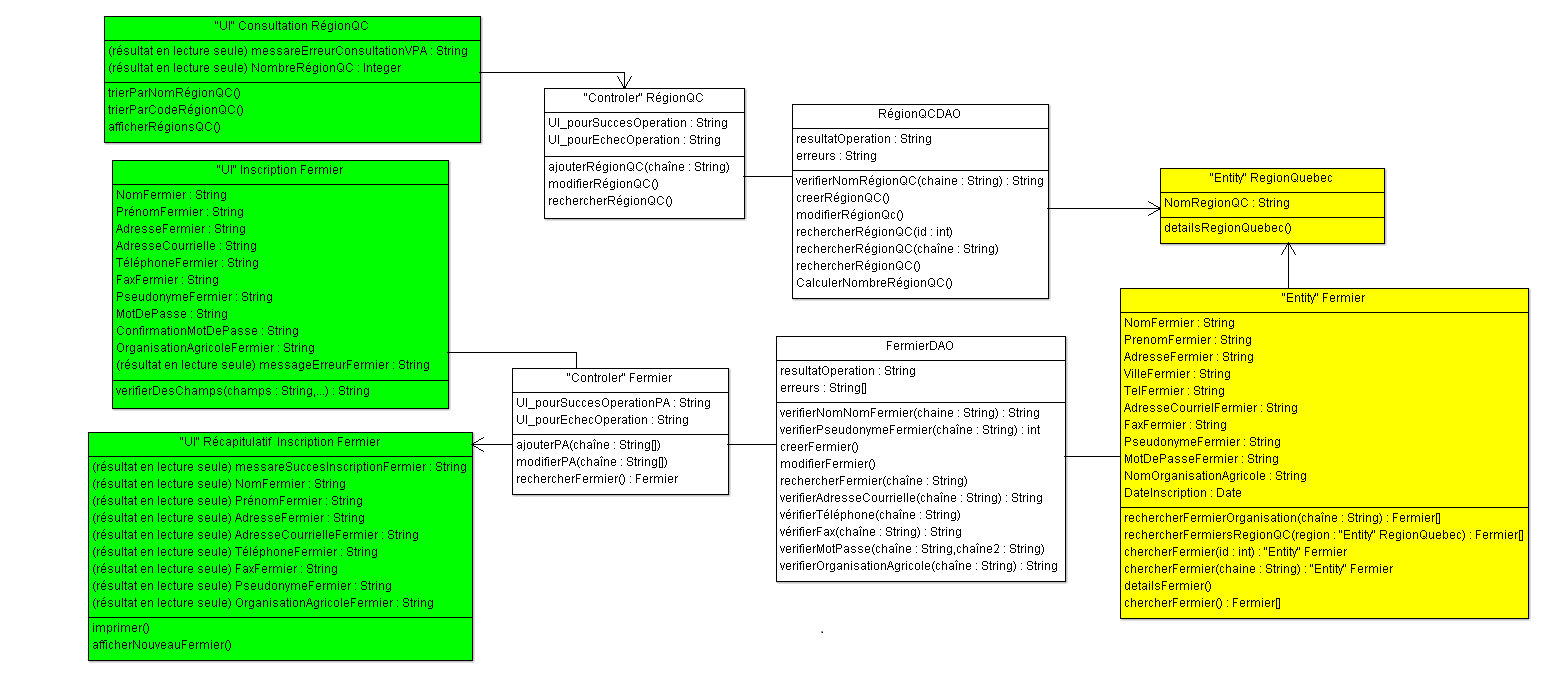
*Figure 13 : diagramme de séquence de création d’une récolte PA*

### 4.2.4.     Raisonnement

Dans les diagrammes structurels et de séquences, nous avons mis en œuvre clairement et respecté les spécifications du modèle MVC. Ainsi, nous avons séparé les classes de vue de celles de contrôleur et des entités. Les spécification des méthodes ou fonctions varient selon la couche.

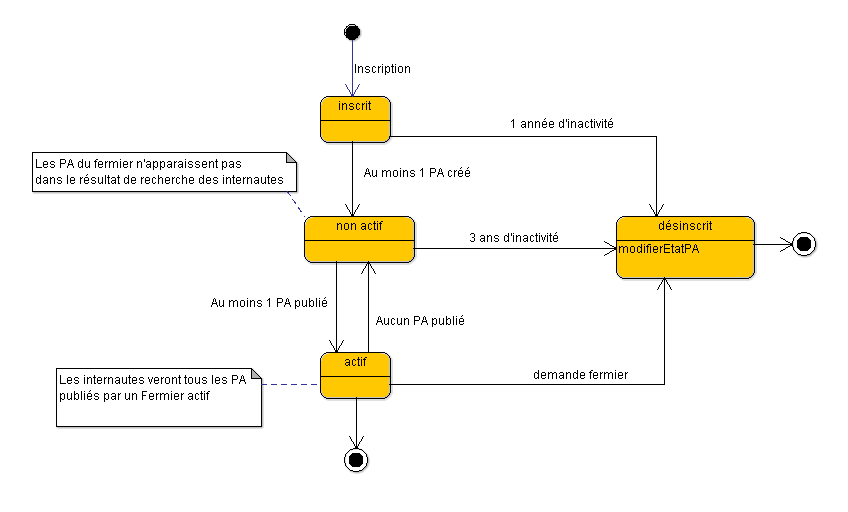
## 4.3. Conception détaillé de l’élément 2 : Inscription Fermier

### 4.3.1.     Vue structurelle

Diagramme de classes, de composants, de structures composites, de déploiement, d’objets, de packages. *Figure 14 : diagramme de classes de la gestion d’inscription Fermier*

### 4.3.2.     Vue comportementale

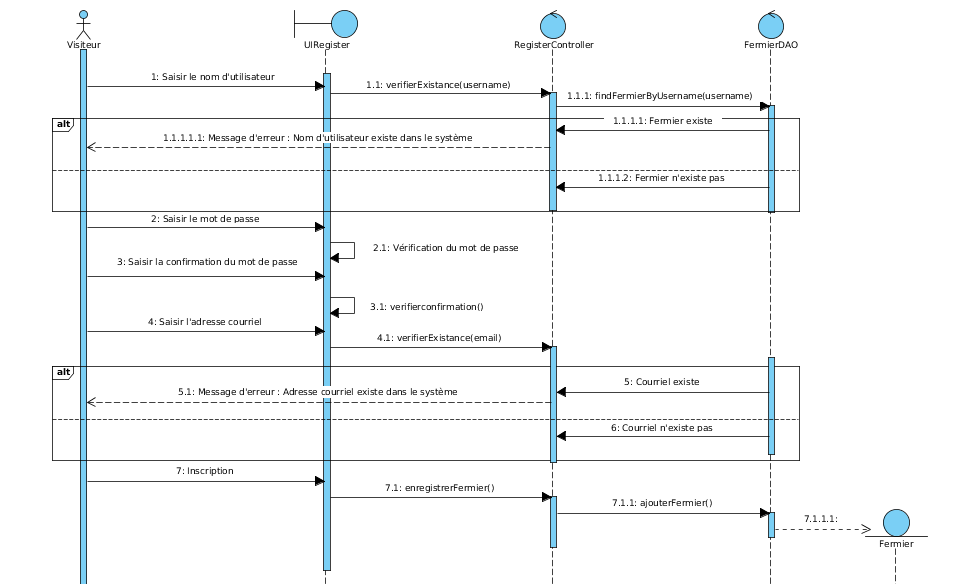
Tout comme le PA, le Fermier est un principal objet de l'application ProdAgriviQ. Le diagramme d'état de la figure ci-après présente le cycle de vie d'une instance d'un Fermier :



*Figure 15 : diagramme d’état d’un objet Fermier*

### 4.3.3.     Autres vues pertinentes

**4.3.3.1. Diagramme de séquence d'inscription d'un Fermier**

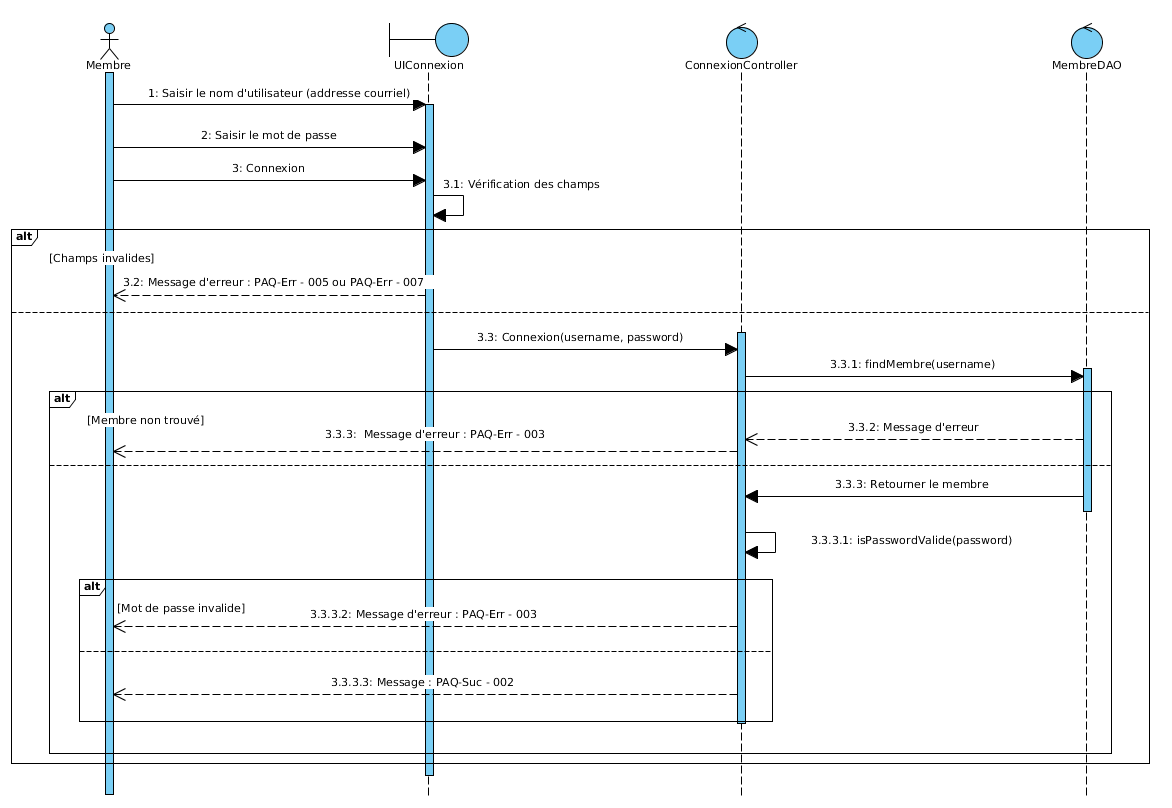


*Figure 16 : diagramme de séquence de l’inscription d’un Fermier*

**4.3.3.2. Diagramme de séquence de connexion d'un Fermier à l'application web ProdAgriviQ**

Pour se connecter à ProdAgriviQ, un membre notamment un Fermier fournit son nom d'utilisateur (pseudonyme) ou bien son adresse courriel et son mot de passe. En effet, ces informations sont celles qu'il a données lors de son inscription. L'application va procéder d'abord à la vérification syntaxique de ces informations au niveau du client (navigateur) avant de l'envoyer à son serveur. Éventuellement, un message d'erreur sera retourné en cas d'erreur syntaxique.

Au niveau du serveur, une robuste vérification est effectuée avant de donnée l'accès au Fermier.

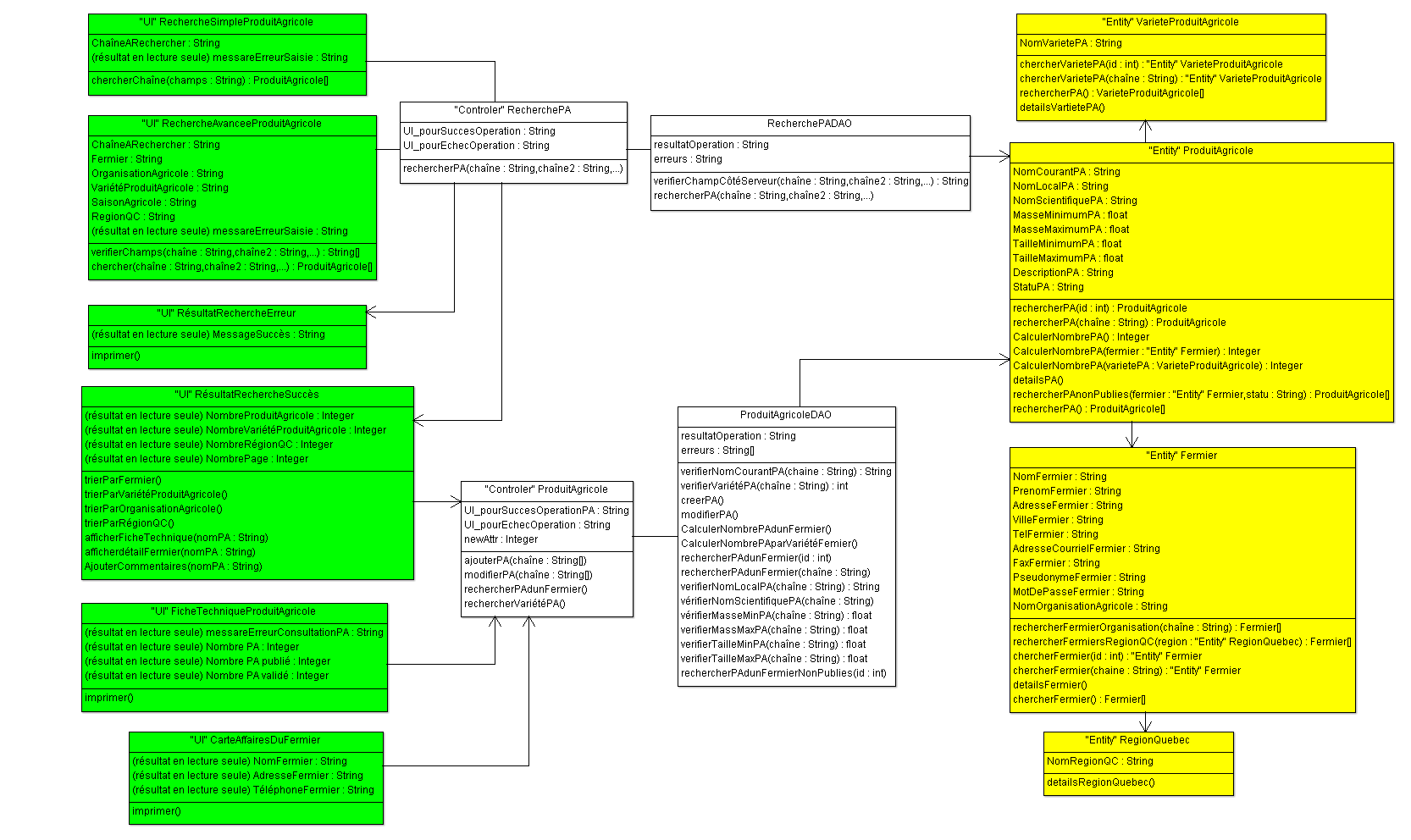


*Figure 17 : diagramme de séquence de connexion d’un Fermier*

## 4.4. Conception détaillé de l’élément 3 : Rechercher produit agricole

Les produits agricoles et récoltes agricoles d'une saison créés et publiés par les fermiers figureront dans les résultats de recherche des internautes.

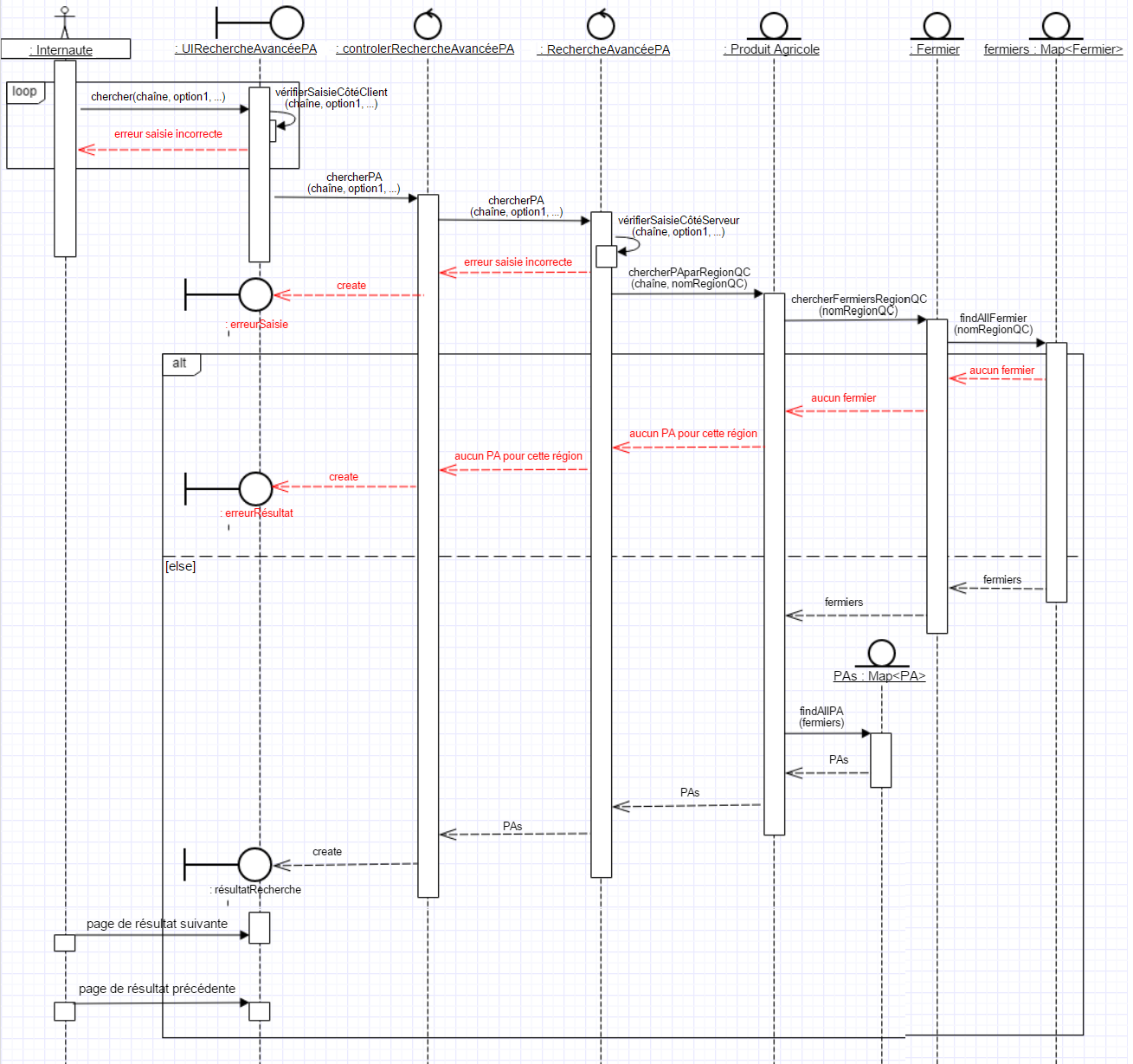
### 4.4.1.     Vue structurelle



*Figure 18 : diagramme de classes du module de recherche de PA*

### 4.4.2. Diagramme de séquence de recherche multicritère de PA d'une région de Québec

Les objets de l'application ProdAgriviQ qui sont sollicités lors de la recherche simple ou multicritère de PA sont nombreux et varient en fonction des critères choisis par le fermier. Le diagramme ci-dessous est celui de la recherche de PA d'une région du Québec par un internaute. Ce diagramme de séquence met en exergue les interactions entre les instances de vue, de contrôleur et de modèle.  Au sujet des instances du modèle, l'on peut remarquer les interactions entre les entités Fermier et Produit Agricole qui visent à fournir une réponse à la demande de l'internaute. Au cas où, aucun Fermier n'est retrouvé pour la région de Québec spécifiée par l'internaute, un message d'erreur lui est retourné sinon le résultat de recherche comportant les PA et informations des fermiers correspondant est affiché :



*Figure 19 : diagramme de séquence de la recherche de PA d’une région du Québec*