

CNAM Paris

NFE114 - Systèmes d'information web

APPLICATION DE PARTAGE DE CONTENU ET MESSAGERIE.

Nouailhaguet Luc

Table des matières

Présentation du projet :	4
Découpage de l'analyse :	5
Préparation :	5
Méthode UWE :	6
Empilement parallèle :	7
Planification analytique :	9
Cas d'utilisations :	9
Package UC internaute	10
Cas d'utilisation S'enregistrer sur le site	10
Package UC stockage :	12
Cas d'utilisation Ajouter des fichiers	12
Package carnet d'adresse	15
Cas d'utilisation : Supprimer un contact	15
Package Authentification :	17
Cas d'utilisation S'authentifier	17
Cas d'utilisation Modifier son profil	19
Cas d'utilisation : Effectuer un signalement	21
Cas d'utilisation : Se déconnecter	21
Package conversation :	22
Cas d'utilisation Construire une conversation	22
Cas d'utilisation : Répondre à une invitation	23
Cas d'utilisation : partager du contenu	25
Cas d'utilisation : Envoyer un message	27
Sous package communication	29
Cas d'utilisation : Passer un appel	29
Sous-package Gestion	31
Cas d'utilisation Archiver une conversation	32
Cas d'utilisation Inviter un utilisateur	33
Maquette des interfaces :	35
Interface d'enregistrement :	35
Interface de connexion :	35
Interface Profil :	36

Interface stockage	37
Interface Carnet d'adresse	38
Interface des conversations	40
Interface de partage de contenu :	40
Interface Edition/création de conversation :	41
Interface d'invitation	42
Interface de communication :	43
Application de la méthode UWE :	44
Diagramme de contenu :	44
Utilisateur :	44
Propriétaire :	45
Diagrammes de navigations	46
Utilisateur :	46
Propriétaire :	48
Participant :	49
Diagrammes de présentation :	51
Utilisateur :	51
Participant :	55
Propriétaire :	61
Implémentation :	62
Présentation :	62
Infrastructure :	62
Le framework :	63
Structure du système de fichier :	66
Enregistrement :	66
Authentification :	69
Profil :	70
Conversation :	74
Références :	77
Bibliographie :	77
Sites internet :	77
Site du projet :	77

Présentation du projet :

On se propose de réaliser un site internet de partage de contenu entre utilisateurs.

Chaque utilisateur, une fois connecté, peut partager du contenu avec tout autre utilisateur du site.

On distingue deux types de contenus :

- binaire : images, sons, vidéo, fichiers
- textuel : message

Les contenus peuvent être déposés sur le site, ils peuvent aussi provenir d'une source extérieure par le biais de l'utilisation d'url à l'instar des liens de lecteur vidéo embarqués YouTube.

Pour les contenus de type binaire, on limite en volume d'espace disque les éléments déposés.

Pour les contenus de type message, on limite en nombre de caractères.

Les utilisateurs du système s'enregistrent sur le site via une interface web.

Lors de l'enregistrement d'un utilisateur sur le site, ce dernier reçoit un mail de confirmation dans lequel figure un lien de validation d'inscription. Une fois cliqué, ce lien lui permettra d'accéder à son espace personnel sur le site.

Depuis cet espace personnel, l'utilisateur a la possibilité de :

- Stocker du contenu de type binaire via une interface de gestion
- Accéder aux conversations auxquelles il participe
- Accéder à un carnet de ses contacts
- Editer ses informations de profil

Les conversations sont opaques pour les non-invités, et complètement transparentes pour les interlocuteurs.

Pour communiquer, un utilisateur du système initie une conversation en la construisant et y invite d'autres utilisateurs.

Ces derniers seront notifiés de cette invitation par les voies de notifications paramétrées dans le profil et auront le loisir d'y répondre dans un canal de conversation spécifique : leur canal d'invitation.

Le créateur d'une conversation est propriétaire de celle-ci. Pour le moment, il ne peut pas transférer la propriété à d'autres participants de la conversation. Seul le propriétaire de la conversation peut changer le nom de la conversation.

Tout participant peut ajouter d'autres utilisateurs à la conversation.

L'ajout d'utilisateur à une conversation peut se faire en utilisant un identifiant unique spécifique au site ou en utilisant la liste de ses contacts.

La seule manière d'ajouter des utilisateurs à une liste de contact est qu'il participe à une conversation.

Tout utilisateur invité se voit ajouté à sa liste de contact, l'utilisateur qui l'a invité.

Chaque conversation permet aux utilisateurs

- De partager du contenu
- D'initialiser un appel vidéo

Il n'y a pas de système de modération sur le site mais on prévoit une adresse électronique pour permettre aux utilisateurs de signaler des comportements pénallement condamnables.

Tout utilisateur peut supprimer tout autre utilisateur de sa liste de contact. Toute suppression est réciproque.

Les conversations peuvent être archivées par leur propriétaire. Au bout d'une durée non déterminée pour le moment, les archives sont supprimées. Les conversations archivées restent accessibles en consultation pour l'ensemble des interlocuteurs.

La notion de statut n'existe pas pour l'utilisateur.

Il n'existe pas de moteur de recherche des utilisateurs du site.

Découpage de l'analyse :

Préparation :

On se propose de découper l'analyse projet par itérations successives en se servant du produit de chaque itération comme donnée d'entrée de l'itération suivante.

Sachant qu'il est tout à fait possible de revenir en arrière sur certaines itérations afin de les enrichir de ce qu'une itération en cours permet de soulever comme éléments analytiques non induis par l'itération précédente.

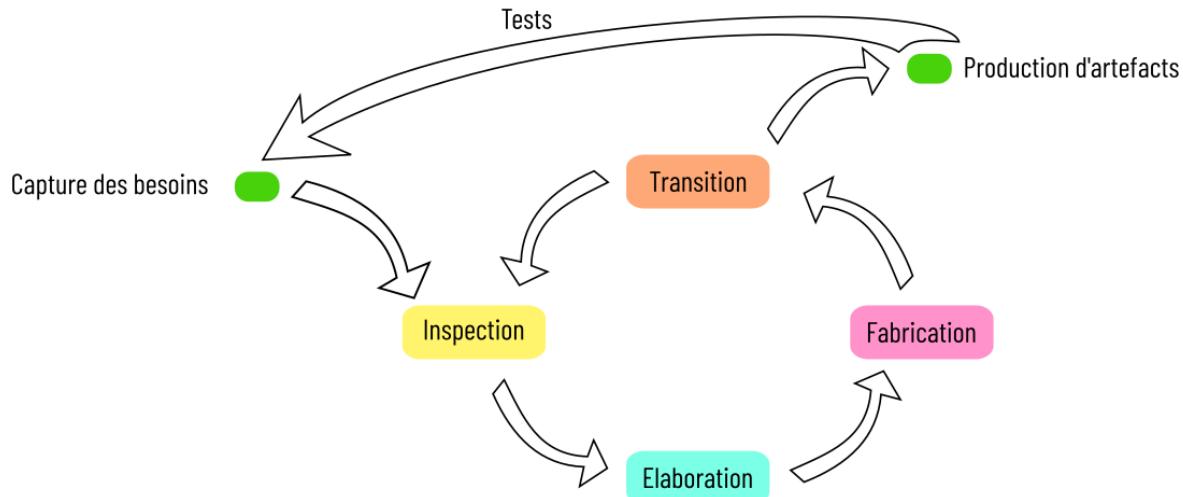


Figure 1 - Itérations appliquées à l'analyse

La phase de test consiste dans la confrontation de l'actuel aux besoins escomptés.

Méthode UWE :

La méthode UWE est un profil UML particulier taillé pour la modélisation d'application web. Les diagrammes se décomposent essentiellement en :

- Diagramme de contenu statique
- Diagramme de navigation
- Diagramme de présentation

Un ensemble de transformations permet de passer de type de diagramme en type de diagramme suivant la séquence illustrée.



Figure 2 - Séquence des phases de la méthode UWE

La caractéristique de l'application de cette méthode est que les acteurs du système procèdent de contenus qui sont propres à l'exercice de leur métier, induisant par le fait des diagrammes de navigation et de présentation spécifiques.

Il est donc essentiel de bien définir les acteurs et leurs cas d'utilisation.

Dans la mesure où l'on souhaite aller assez vite, nous allons utiliser une petite mise en parallèle de la méthode classique pour la rendre plus puissante.

Empilement parallèle :

Dans un premier temps on découpe l'application non pas comme une séquence mais comme un ensemble de trois partitions :

- Contenu
- Navigation
- Présentation

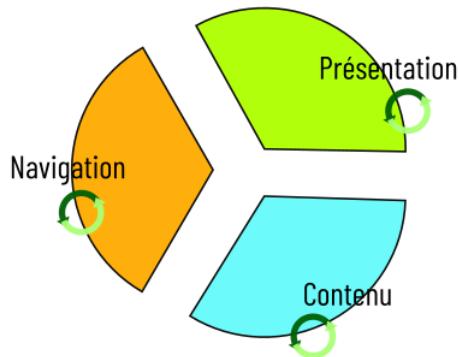


Figure 3 - Partitions des espaces du profil UWE

Au lieu de commencer par le diagramme de contenu, on va effectuer le travail sur les trois parties en même temps.

Le bootstrap du contexte d'analyse sera effectué grâce aux méthodes classiques UML de modélisation de l'expression des besoins.

C'est ce contexte analytique qui sera le contexte des itérations, devenant ainsi le référentiel au sein duquel on se servira des transformations du profil UWE pour faire converger la modélisation vers le produit fini.

L'idée est de permettre l'itération en silo.

Par exemple pour la partie présentation, de manière non exhaustive, il devient possible d'itérer en silo :

- La conception des écrans d'interface utilisateur
- Les transformations UWE des diagrammes de navigation.

Ainsi en va-t-il aussi des diagrammes de séquence système pour la pile de navigation.



Figure 4 - Pile des partitions

L'intégrité structurelle du profil UWE induira par transformation inverse d'une pile vers une autre l'enrichissement du modèle complet.



Figure 5 - Report du résultat des itérations sur les piles adjacentes

Planification analytique :

Nous allons déterminer les cas d'utilisation et produire un découpage en package afin de découpler au maximum la base de code.

Pour chaque cas d'utilisation nous détaillerons le diagramme de séquence système et la description textuelle associée ainsi que les scénarios alternatifs.

Puis après une première analyse du découpage statique, nous produirons un ensemble d'écran représentant les interfaces utilisateurs.

Nous aurons ainsi défini la base fonctionnelle minimale de production sur laquelle itérer UWE.

Cas d'utilisations :

D'après les besoins projet on distingue 4 acteurs :

- Internaute
- Utilisateur du système
- Participant d'une conversation
- Propriétaire d'une conversation

Ainsi que 5 axes statiques :

- Le stockage

- L'authentification
- Le carnet d'adresse
- Les conversations
- Le hors système

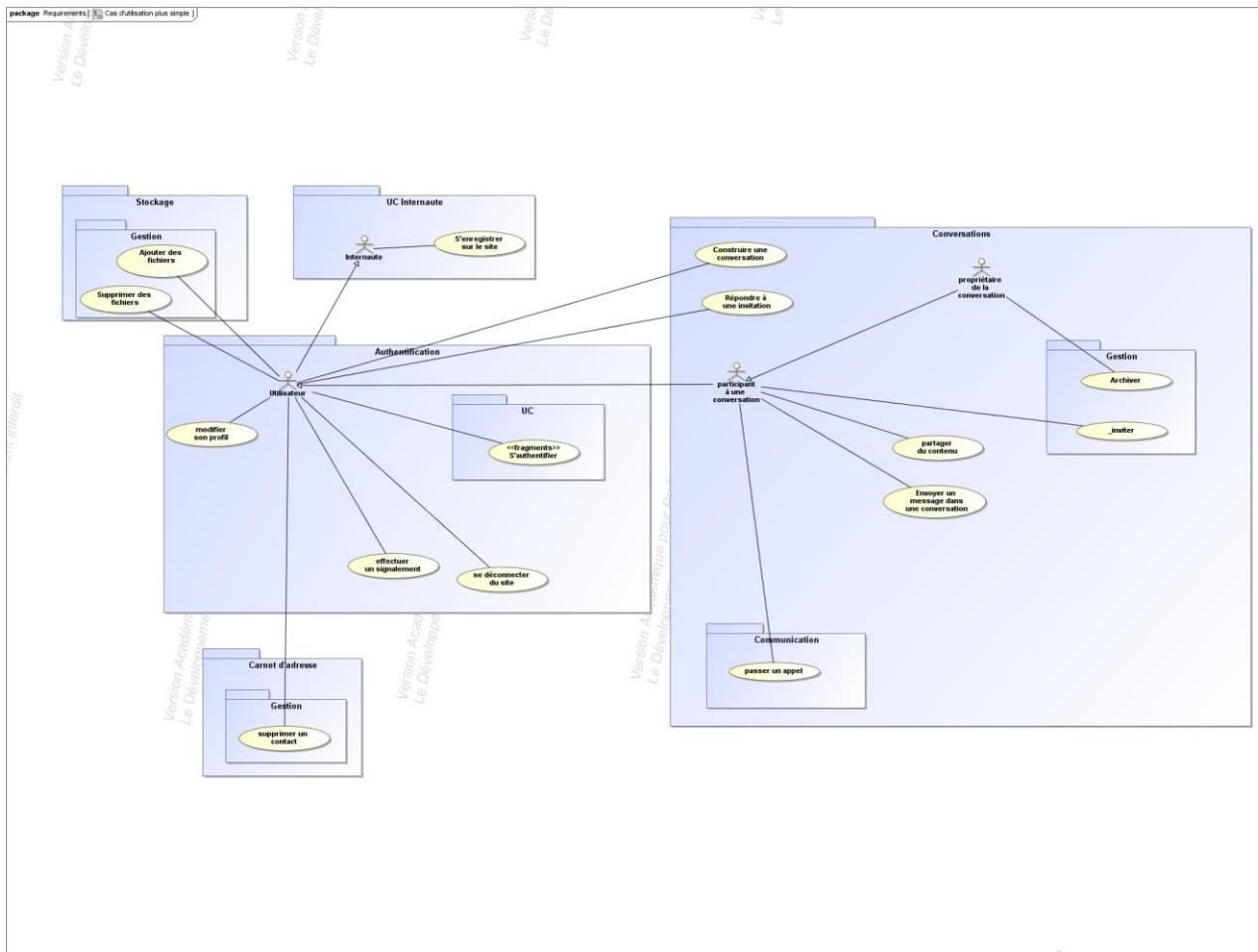


Figure 6 - Cas d'utilisation

Package UC internaute

Cas d'utilisation S'enregistrer sur le site

Description textuelle :

Acteur principal :

Un internaute.

Objectif :

Pouvoir s'authentifier dans le système

Scénario nominal :

1. L'internaute demande à s'enregistrer en tant qu'utilisateur
2. Le système lui propose une interface d'enregistrement
3. L'internaute renseigne les champs courriel et mot de passe de l'interface
4. Le système valide les données renseignées.
5. Le système vérifie l'identité de l'utilisateur en envoyant un lien de validation d'inscription au courriel renseigné précédemment.
6. L'internaute active son compte utilisateur.
7. Le système vérifie que l'activation est valide.
8. Le système envoie une interface de profil d'utilisateur.

Alternatives :

Le courriel existe déjà :

4a. Le système identifie que le courriel est déjà attribué.

1. Le système signale que le courriel est déjà attribué.

LE SCENARIO REPREND AU POINT 3 DU SCENARIO NOMINAL.

Le courriel est mal formatté :

4b. Le système identifie que le courriel est mal formatté.

1. Le système signale que le courriel est déjà attribué.

LE SCENARIO REPREND AU POINT 3 DU SCENARIO NOMINAL.

Erreurs :

Le lien d'activation n'est plus valide : (la date de validité est expirée)

7a. Le système identifie que la date de validité du token d'activation est expirée.

1. Le système signale à l'utilisateur que le token d'activation est expirée.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

Diagramme de séquence système :

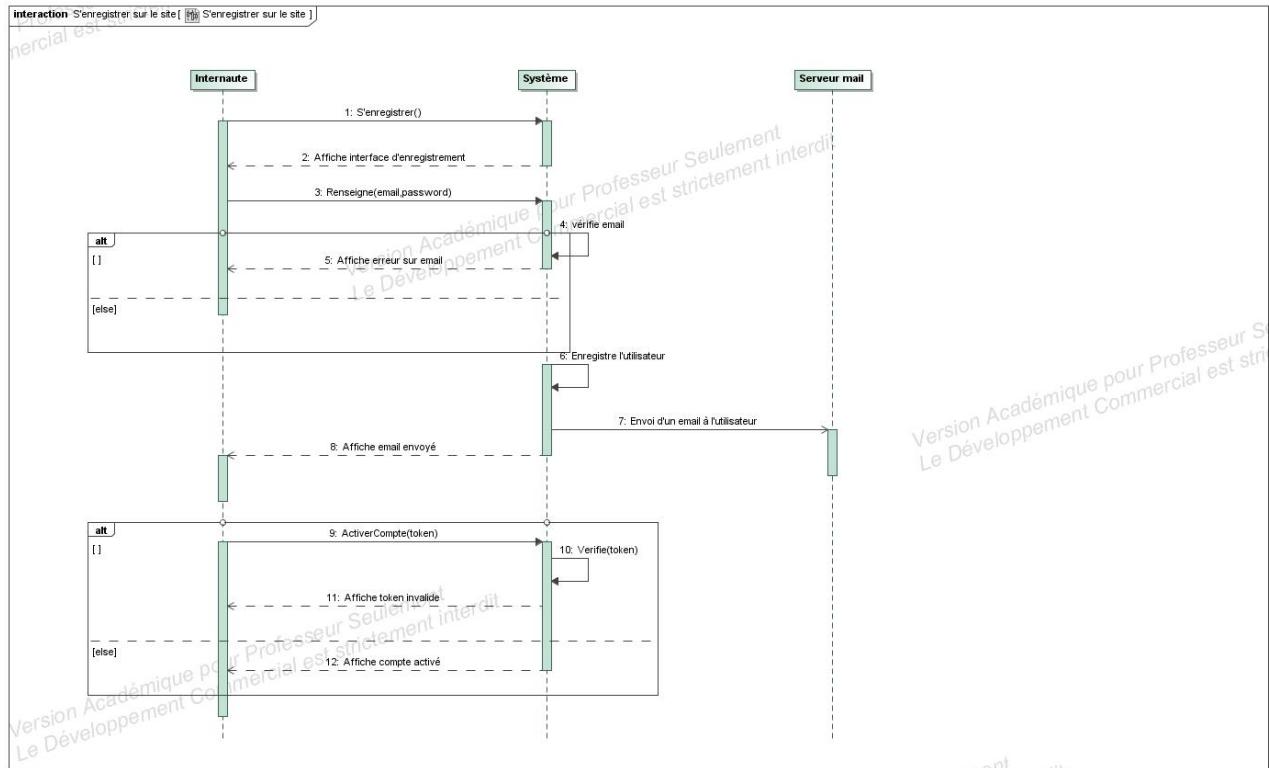


Figure 7 - Diagramme de séquence système du cas d'utilisation authentification

Package UC stockage :

Sous-package gestion

Cas d'utilisation Ajouter des fichiers

Description textuelle :

Cas d'utilisation

Ajouter des fichiers

Acteur principal :

Un utilisateur.

Objectif :

Stocker des fichiers dans son espace de stockage.

Précondition :

Être authentifié.

Postcondition :

Les données sont stockées dans le système.

Scénario nominal :

1. L'utilisateur demande l'interface de transfert de fichier.
2. Le système lui renvoie l'interface de transfert.
3. L'utilisateur choisit le fichier à téléverser.
4. Le système vérifie les conditions de stockage.
5. Le système range le fichier dans la bonne catégorie au bon endroit.
6. Le système notifie l'utilisateur que son fichier a été enregistré.

Alternatives :

4a. Le système s'aperçoit que le volume du fichier dépasse la capacité de stockage.

1. Le système affiche un message à l'utilisateur le notifiant du problème.

LE CAS D'UTILISATION REPRENDS AU CAS 1.

Diagramme de séquence système :

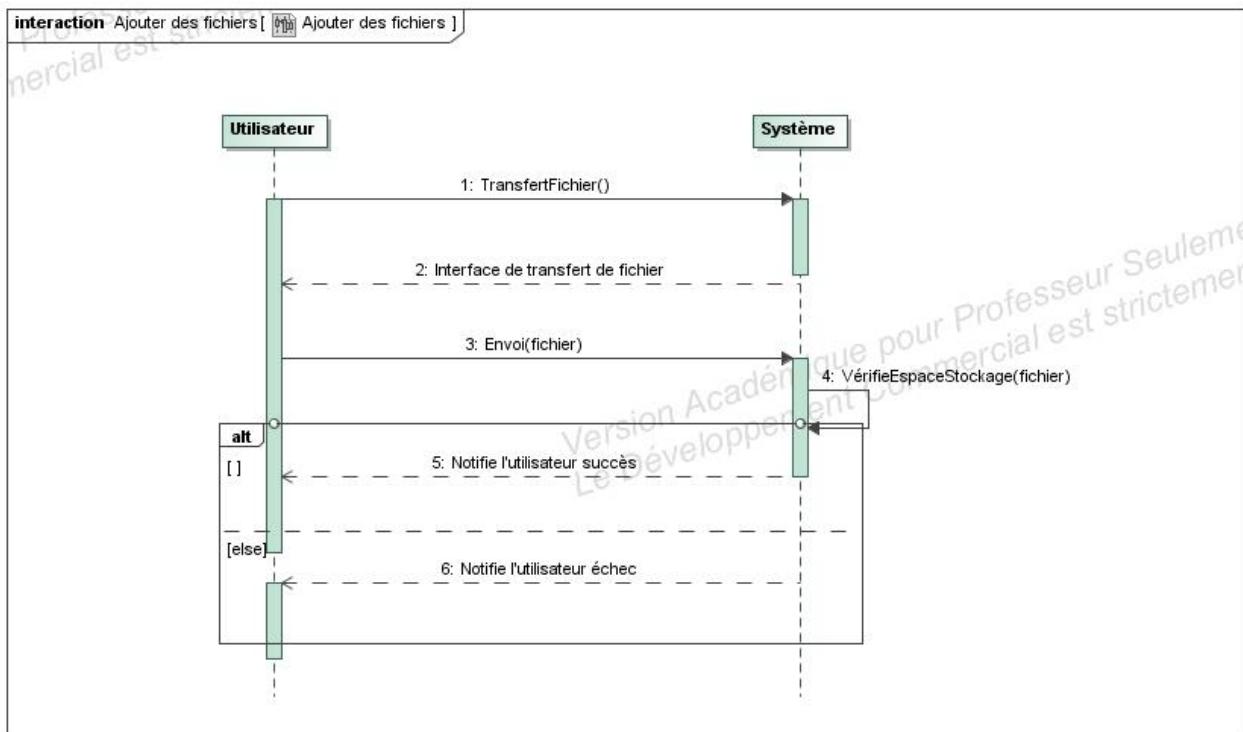


Figure 8 - Diagramme de séquence du cas d'utilisation d'ajout de fichier

Cas d'utilisation : Supprimer des fichiers

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Supprimer des fichiers de l'espace de stockage.

Précondition :

Être authentifié.

Postcondition :

Les fichiers sont supprimés de l'espace de stockage.

Scénario nominal :

1. L'utilisateur demande la liste des fichiers présent dans son espace de stockage.
2. Le système lui renvoie la liste des fichiers.
3. L'utilisateur choisit un fichier à supprimer dans la liste proposée.

4. Le système supprime le fichier de l'espace de stockage.
5. Le système notifie l'utilisateur de la suppression.

Erreurs :

4a. Le système ne peut pas supprimer le fichier.

1. Le système indique à l'utilisateur que la suppression n'a pu être effectuée.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

Diagramme de séquence système :

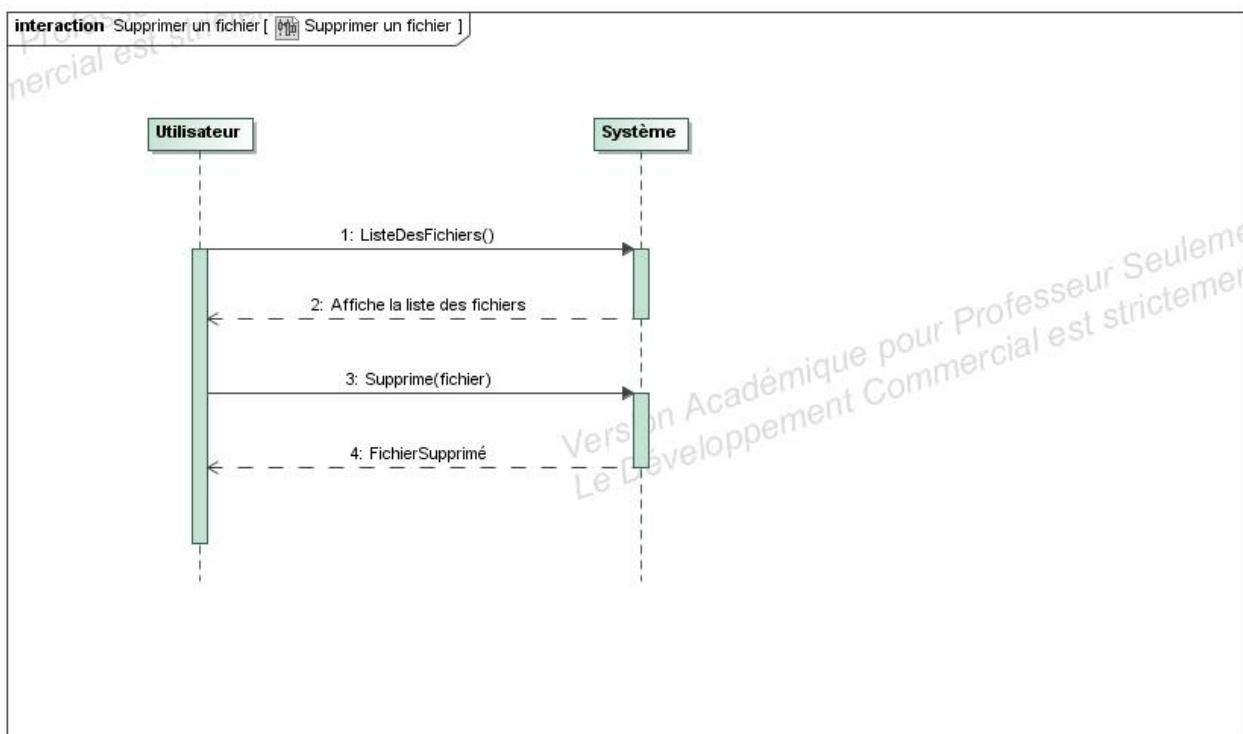


Figure 9 - diagramme de séquence système du cas d'utilisation suppression de fichiers

Package carnet d'adresse

Sous-package gestion

Cas d'utilisation : Supprimer un contact

Description textuelle :

Acteur principal :

Un utilisateur.

Objectif :

Supprimer un contact de la liste de ses contacts.

Précondition :

Être authentifié.

Postcondition :

Le contact n'est plus présent dans la liste des contacts de l'utilisateur.

L'utilisateur n'est plus présent dans la liste des contacts du contact supprimé.

Scénario nominal :

1. L'utilisateur demande la liste des contacts.
2. Le système lui remet la liste de ses contacts.
3. L'utilisateur demande la suppression d'un utilisateur de sa liste des contacts.
4. Le système supprime le contact de la liste.
5. Le système supprime l'utilisateur de la liste des contacts du contact.
6. Le système renvoie la nouvelle liste de contact

Erreurs :

E1 :

4a. Le système ne parvient pas à effectuer la suppression de l'utilisateur :

1. Le système prévient l'utilisateur qu'une erreur s'est produite.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

E2 :

5a. Le système ne parvient pas à supprimer l'utilisateur de la liste des contacts du contact à supprimer.

1. Le système ajoute le contact à la liste des contacts de l'utilisateur. [Opération inverse de l'opération 4]
2. Le système prévient l'utilisateur qu'une erreur s'est produite.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

Diagramme de séquence système :

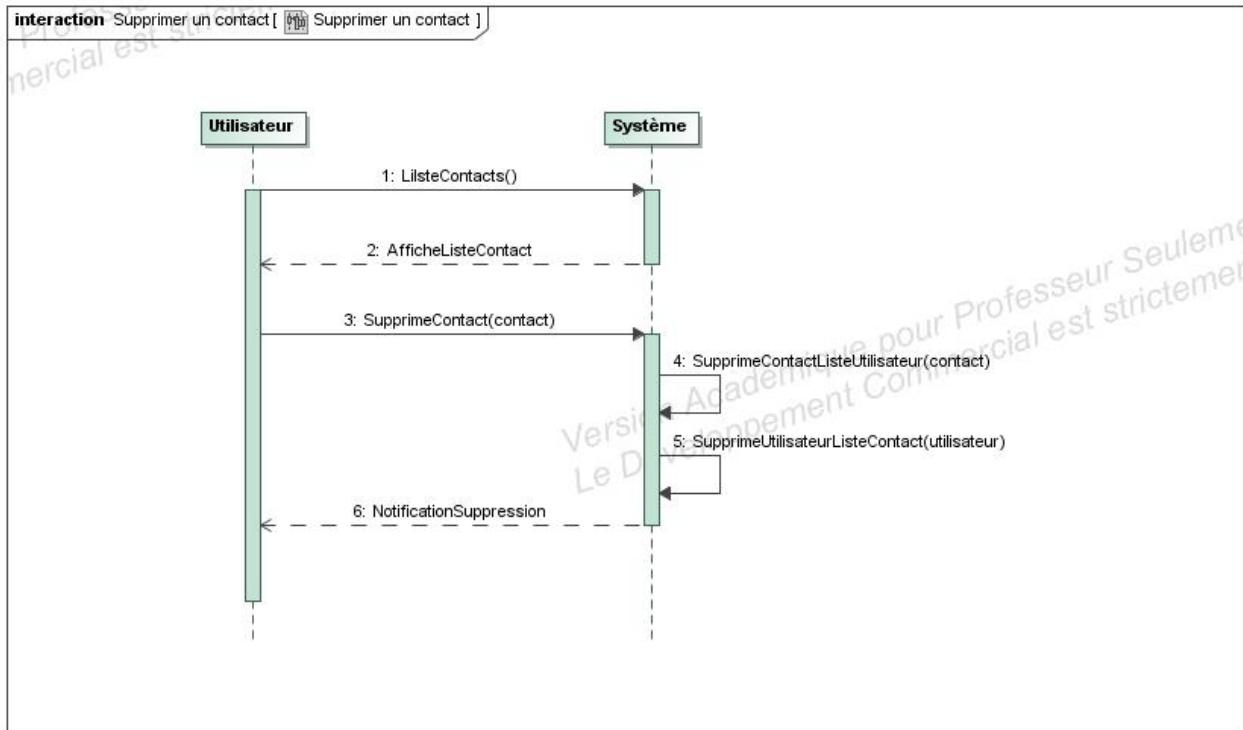


Figure 10 - diagramme de séquence système du cas d'utilisation suppression d'un utilisateur

Package Authentification :

Cas d'utilisation fragmentaire, sous-package UC.

Cas d'utilisation S'authentifier

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Pouvoir interagir avec le système

Scénario nominal :

1. Un utilisateur demande à se connecter.
2. Le système présente une interface de connexion dans laquelle l'utilisateur peut remplir les champs courriel et mot de passe.

3. L'utilisateur renseigne ses identifiants.
4. Le système valide les identifiants.
5. Le système présente à l'utilisateur sa page de profil.

Alternatives :

L'utilisateur commet une erreur au moment de renseigner ses identifiants :

3a. L'utilisateur renseigne des identifiants erronés

1. Le système identifie une erreur dans les identifiants.
2. Le système signale l'erreur à l'utilisateur.

LE SCENARIO REPREND AU POINT 3.

Diagramme de séquence système :

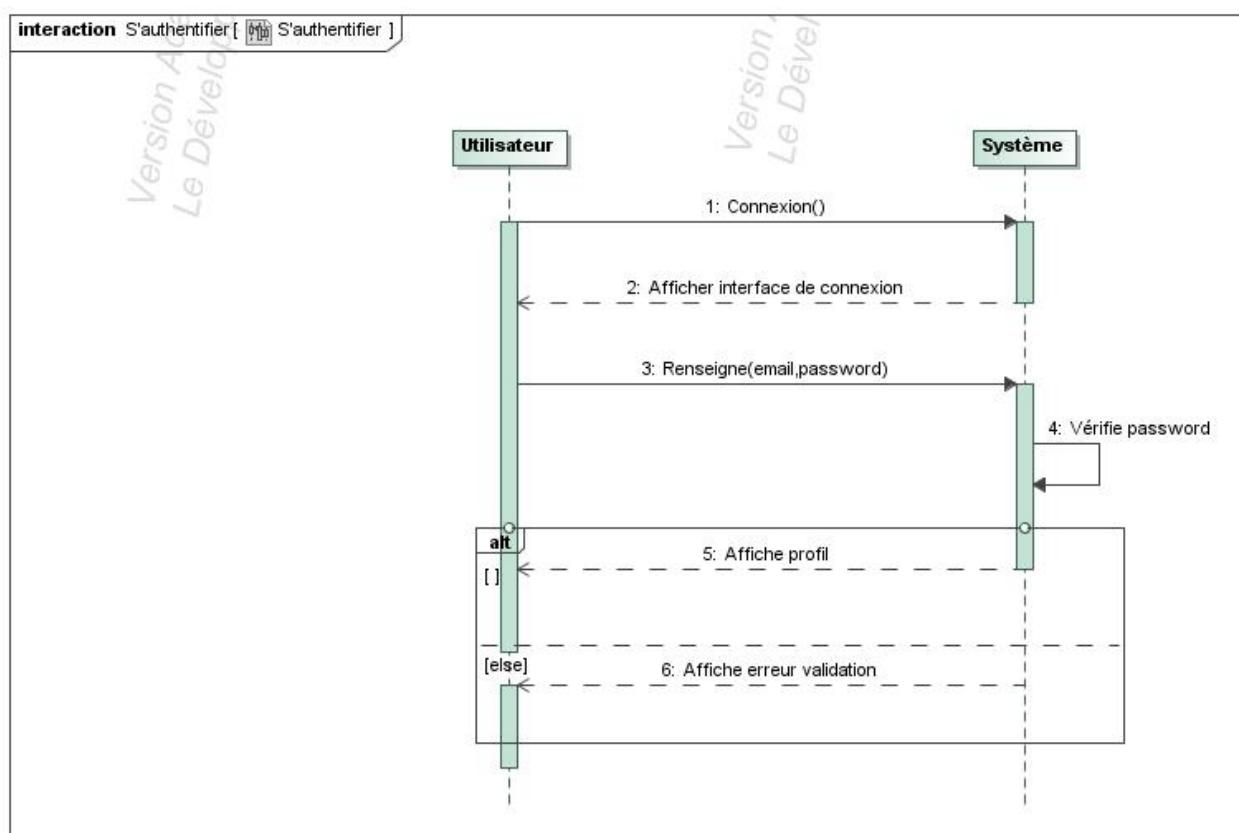


Figure 11 - diagramme de séquence système pour le cas d'utilisation s'authentifier

Cas d'utilisation Modifier son profil.

Description textuelle :

Acteur principal :

Un utilisateur.

Objectif :

Permettre à un utilisateur de gérer son profil.

Précondition :

Être authentifié.

Postcondition :

Le nouveau profil est enregistré.

Scénario nominal :

1. L'utilisateur demande à voir son profil.
2. Le système lui propose une interface de gestion de profil.
3. L'utilisateur modifie les informations.
4. Le système valide les informations.
5. Le système affiche les informations modifiées à l'utilisateur.
6. L'utilisateur demande l'enregistrement des informations modifiées.
7. Le système enregistre le nouveau profil.
8. Le système notifie l'utilisateur de l'insertion.

Alternatives :

L'utilisateur ne respecte pas le format d'entrée des données profil :

3a. L'utilisateur commet des erreurs lors de la modification.

1. Le système vérifie les informations et détecte les erreurs.
2. Le système affiche les informations modifiées et les erreurs.

LE SCENARIO REPREND AU POINT 3.

Erreurs :

7a. Le système ne parvient pas faire la mise à jour du profil.

1. Le système informe que la mise à jour n'a pas pu être effectuée.

LE CAS D'UTILISATION SE TERMINE PAR UN ECHEC.

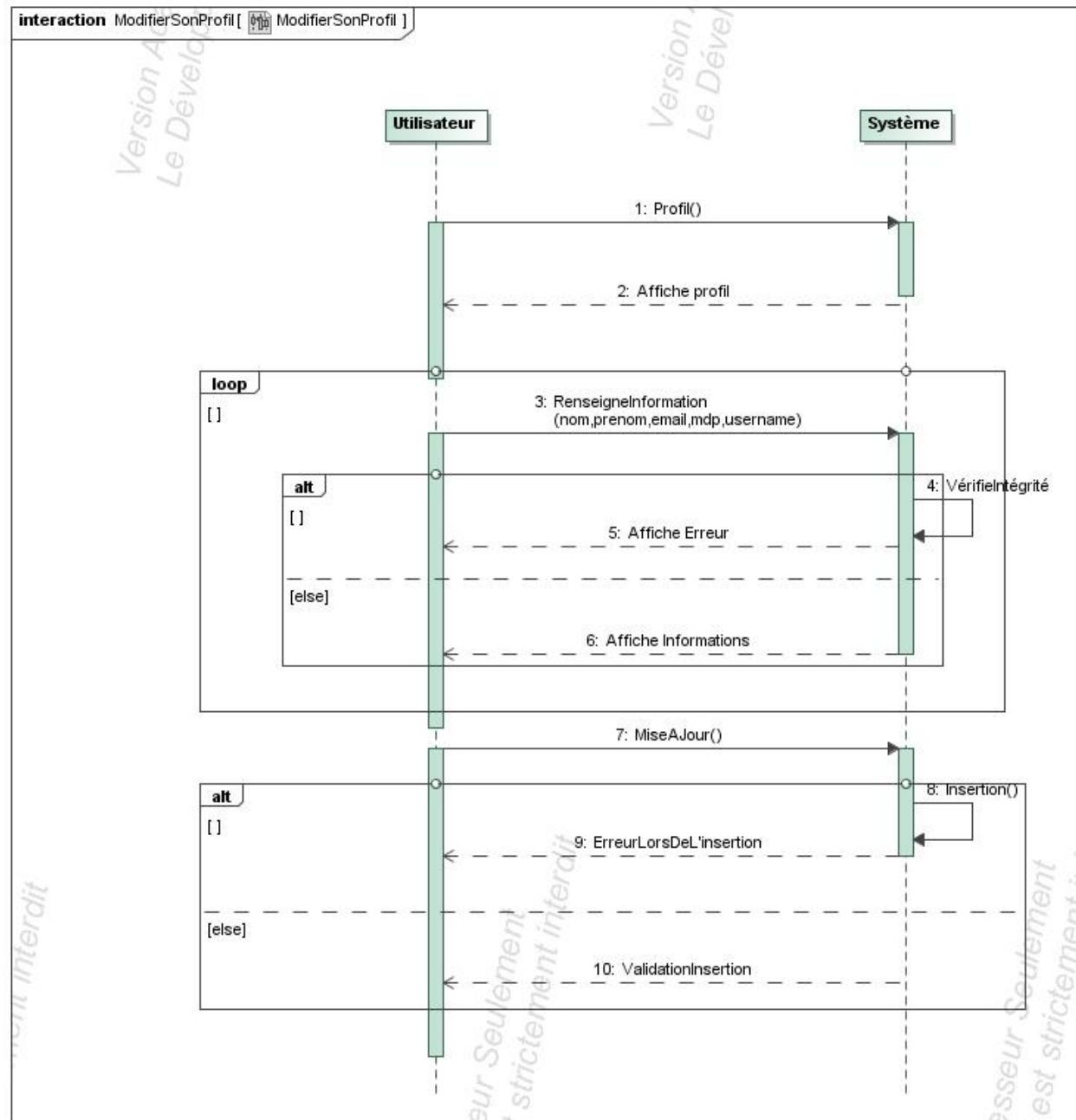


Figure 12 - diagramme de séquence système du cas d'utilisation modification du profil

Cas d'utilisation : Effectuer un signalement.

Description textuelle :

Acteur principal :

Un utilisateur.

Objectif :

Permettre à un utilisateur d'effectuer un signalement.

Précondition :

Être authentifié.

Postcondition :

Le signalement est effectué.

Scénario nominal :

1. L'utilisateur demande à effectuer un signalement.
2. Le système délie la notification à un service d'envoi de courriel externe.

Le diagramme de séquence système est trivial.

Cas d'utilisation : Se déconnecter

Description textuelle :

Acteur principal :

Un utilisateur.

Objectif :

Permettre à un utilisateur de se déconnecter du site.

Précondition :

Être authentifié.

Postcondition :

L'utilisateur est déconnecté.

Scénario nominal :

1. L'utilisateur demande à être déconnecté du système.
2. Le système valide la déconnexion.

Le diagramme de séquence système est trivial.

Package conversation :

Cas d'utilisation Construire une conversation

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Obtenir un canal de conversation dans lequel converser

Précondition :

Être authentifié.

Postcondition :

Une nouvelle conversation est disponible à l'utilisateur.

Scénario nominal :

1. L'utilisateur demande à construire une conversation.
2. Le système lui envoie une interface de construction.
3. L'utilisateur renseigne le nom de la conversation.
4. Le système instancie la conversation.
5. Le système prévient l'utilisateur que la conversation est créée.

Erreurs :

4a. Une erreur se produit lors de l'instanciation.

1. Le système prévient l'utilisateur qu'une erreur s'est produite.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

Diagramme de séquence système

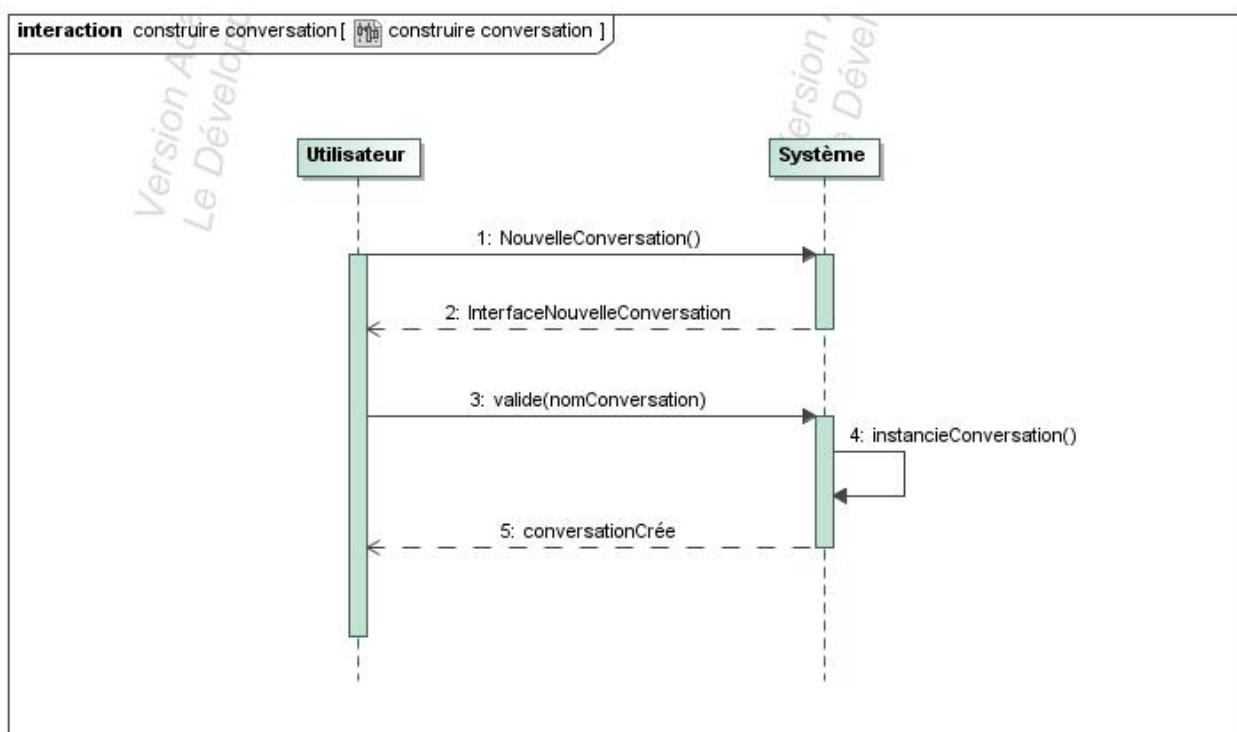


Figure 13 - diagramme de séquence système pour le cas d'utilisation de déconnexion

Cas d'utilisation : Répondre à une invitation

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Décliner ou accepter sa participation à une conversation.

Précondition :

Être authentifié.

Postcondition :

L'état de l'invitation est défini.

Scénario nominal :

1. L'utilisateur demande la liste de ses invitations.
2. Le système renvoie la liste.
3. L'utilisateur sélectionne une invitation dans la liste.
4. Le système affiche l'interface de l'invitation.
5. L'utilisateur accepte l'invitation
6. Le système vérifie si les deux contacts font mutuellement partie de leur liste de contacts et constate que c'est le cas.
7. Le système ajoute la conversation associée à l'invitation dans la liste des conversations accessibles à l'utilisateur
8. Le système ajoute l'utilisateur à la liste des participants de la conversation.
9. Le système notifie l'émetteur de l'invitation de la décision de l'utilisateur.

Alternatives :

5a. L'utilisateur décline l'invitation

LE CAS D'UTILISATION PASSE DIRECTEMENT AU POINT 9 DU SCENARIO NOMINAL.

6a. Le système vérifie si les deux contacts font mutuellement partie de leur liste de contacts et constate que ce **n'est pas** le cas.

1. Le système ajoute l'utilisateur à la liste des contacts de l'émetteur de l'invitation.
2. Le système ajoute l'émetteur de l'invitation à la liste des contacts de l'utilisateur.

LE CAS D'UTILISATION SE POURSUIT EN PASSANT AU POINT 7 DU SCENARIO NOMINAL.

Diagramme de séquence système :

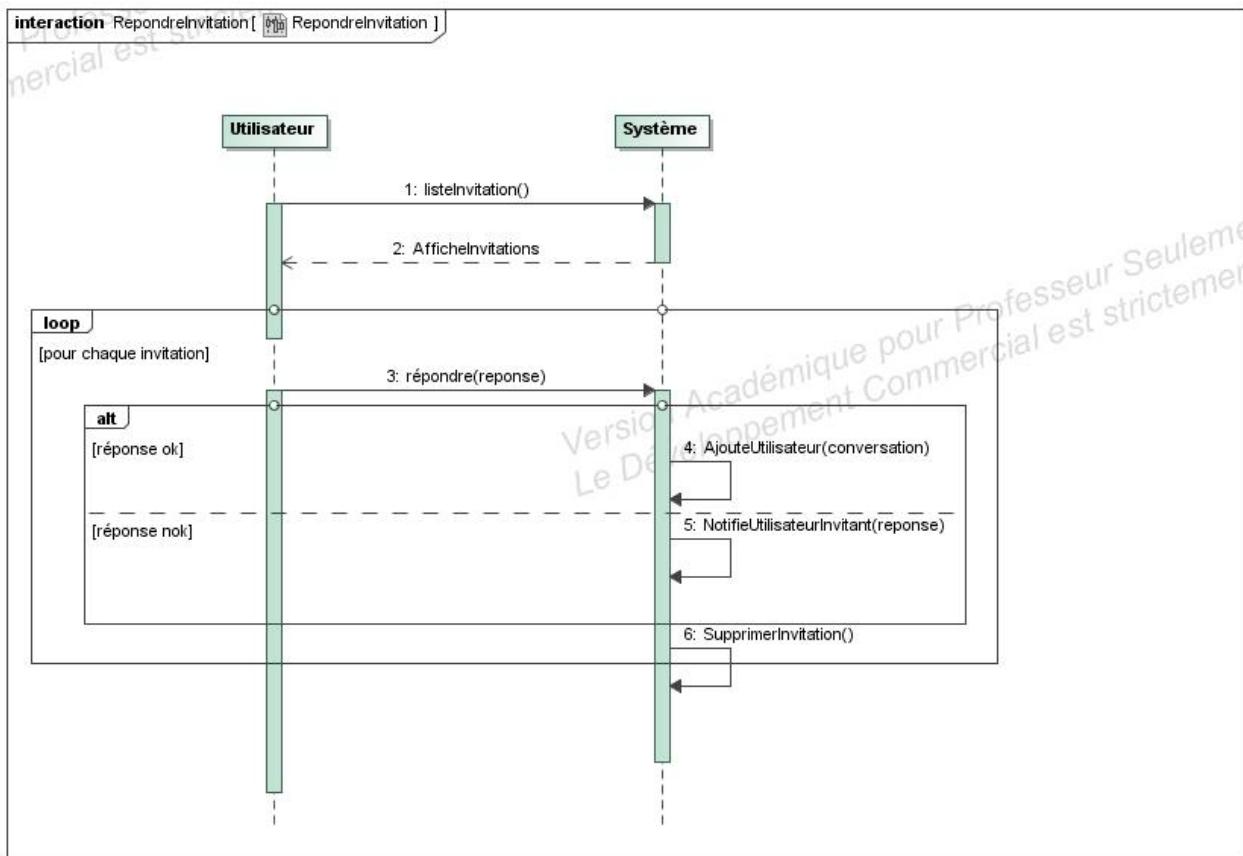


Figure 14 - diagramme de séquence système de réponse à une invitation

Cas d'utilisation : partager du contenu

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Faire en sorte de rendre un contenu accessible à des participants d'une conversation.

Précondition :

Être authentifié.

Postcondition :

Le contenu est accessible aux participants d'une conversation.

Scénario nominal :

1. L'utilisateur demande la liste des conversations.
2. Le système renvoie la liste des conversations.
3. L'utilisateur sélectionne une conversation.
4. Le système affiche l'interface de la conversation.
5. L'utilisateur demande à partager du contenu.
6. Le système cherche le contenu stocké par l'utilisateur.
7. Le système affiche la liste des fichiers stockés par l'utilisateur.
8. L'utilisateur choisit un fichier.
9. Le système publie dans la conversation un lien vers le fichier de l'utilisateur.

Alternatives :

5a. L'utilisateur copie le fichier à partager dans la conversation

1. Le système détermine si le fichier peut être stocké dans l'espace de stockage.
2. Le système copie le fichier dans l'espace de stockage.

LE SCENARIO REPREND AU POINT 9 DU SCENARIO NOMINAL.

Erreur :

5a1a Le système ne peut pas copier le fichier dans l'espace de stockage de l'utilisateur.

1. Le système avertit l'utilisateur que le fichier ne peut pas être copié dans l'espace de stockage.

LE SCENARIO SE TERMINE EN ECHEC.

Diagramme de séquence système :

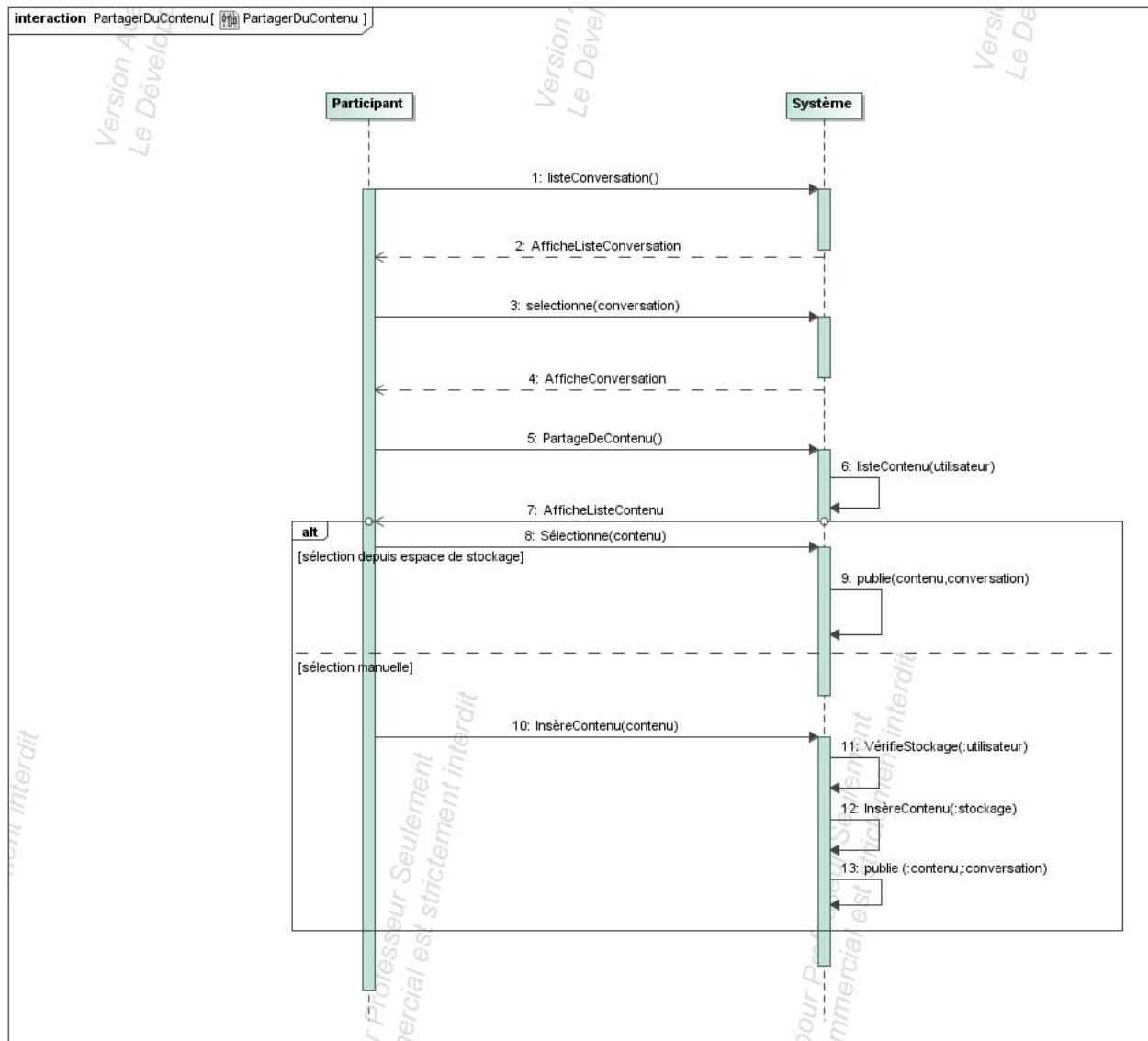


Figure 15 - diagramme de séquence système du cas d'utilisation de partage de contenu

Cas d'utilisation : Envoyer un message

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Envoyer un message dans une conversation.

Précondition :

Être authentifié.

Postcondition :

Le message est publié dans la conversation.

Scénario nominal

1. L'utilisateur la liste des conversations
 2. Le système renvoie la liste des conversations
 3. L'utilisateur sélectionne une conversation
 4. Le système renvoie la conversation
 5. L'utilisateur envoie un message dans la conversation
 6. Le système enregistre le message dans la conversation.
 7. Le système affiche la conversation à jour.

Diagramme de séquence système

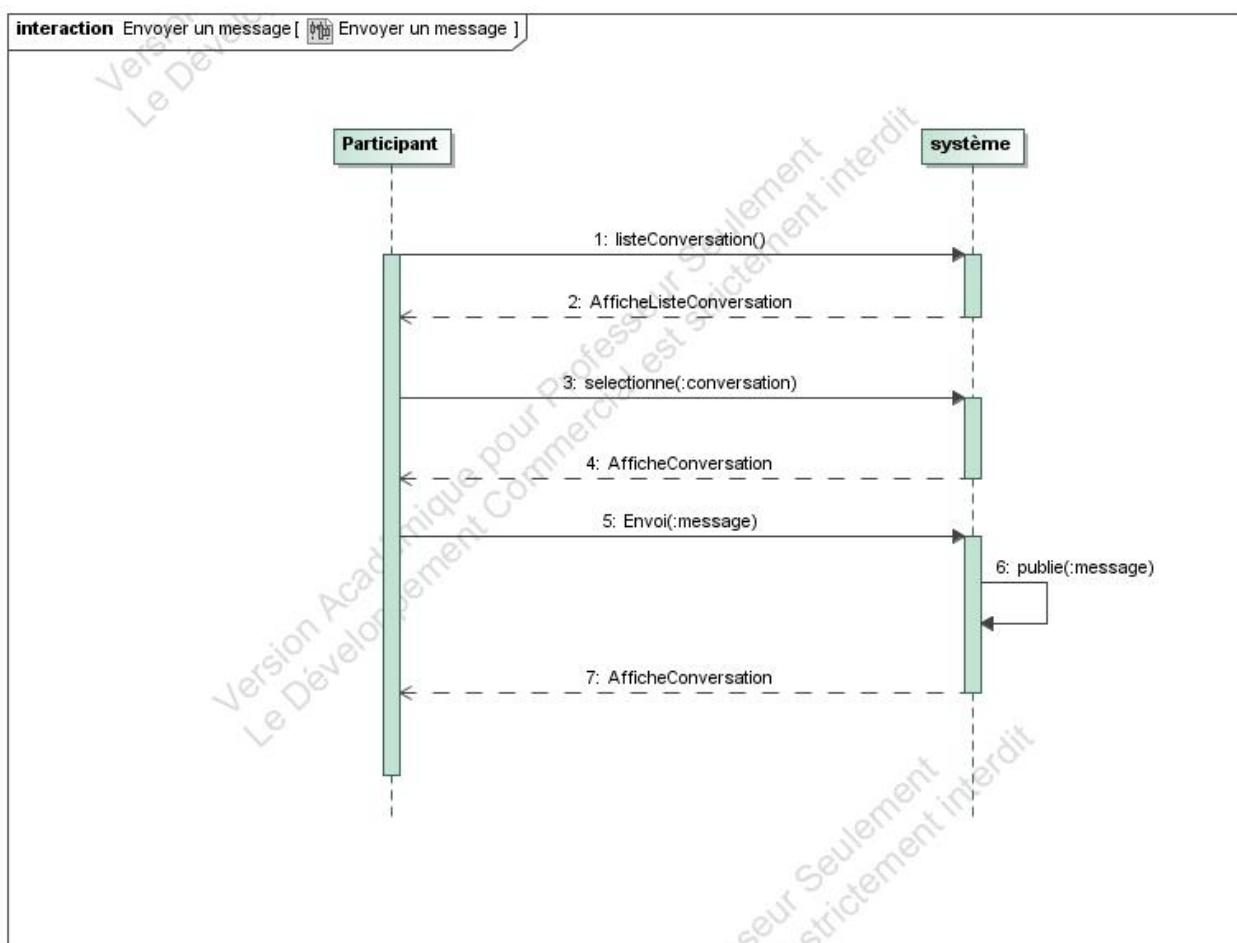


Figure 16 - diagramme de séquence système du cas d'utilisation d'envoi de message

Sous package communication

Cas d'utilisation : Passer un appel

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Etablir une communication vidéo ou audio avec un autre utilisateur.

Précondition :

Être authentifié.

Scénario nominal

1. L'utilisateur demande une liste de ses contacts
2. Le système lui renvoie une interface de sélection
3. L'utilisateur choisit un contact
4. Le système lui renvoie une interface d'interaction
5. L'utilisateur choisit appel
6. Le système envoie une notification d'appel à l'utilisateur appelé
7. Le système envoie une interface de communication à l'appelant
8. L'utilisateur appelé répond favorablement à l'appel
9. Le système met en relation les flux audios des deux utilisateurs
10. Les utilisateurs dialoguent.
11. Un des deux utilisateurs décide de terminer l'appel
12. Le système coupe la liaison

Alternatives

1a. L'utilisateur demande une liste des participants d'une conversation.

1. Le système lui renvoie la liste des participants
2. L'utilisateur sélectionne un participant

3. Le système renvoie une interface d'interaction
4. L'utilisateur choisi appel.

LE CAS D'UTILISATION CONTINU AU POINT 6 DU SCENARIO NOMINAL.

8a. L'utilisateur appelé ne répond pas favorablement à l'appel.

LE CAS D'UTILISATION SAUTE A L'ETAPE 11 DU SCENARIO NOMINAL.

10a. Un des deux utilisateurs ou les deux utilisateurs décident de diffuser la vidéo associé à leur caméra.

1. Le système met en relation les flux vidéo des utilisateurs

LE CAS D'UTILISATION REPREND A L'ETAPE 10 DU SCENARIO NOMINAL.

Extension à l'alternative 10a1 :

1. Un des utilisateurs ou les deux utilisateurs décident de stopper la diffusion vidéo.
2. Le système arrête de mettre en relation les flux vidéo des utilisateurs.

LE CAS D'UTILISATION REPREND A L'ETAPE 10 DU SCENARIO NOMINAL.

10b. Un des deux utilisateurs ou les deux utilisateurs décident d'arrêter la diffuser de leur flux audio

1. Le système arrête de mettre en relation la diffusion audio des utilisateurs.

LE CAS D'UTILISATION REPREND A L'ETAPE 10 DU SCENARIO NOMINAL.

Erreurs

2a. Le système lui renvoie une interface de sélection sur une liste vide.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

Diagramme de séquence :

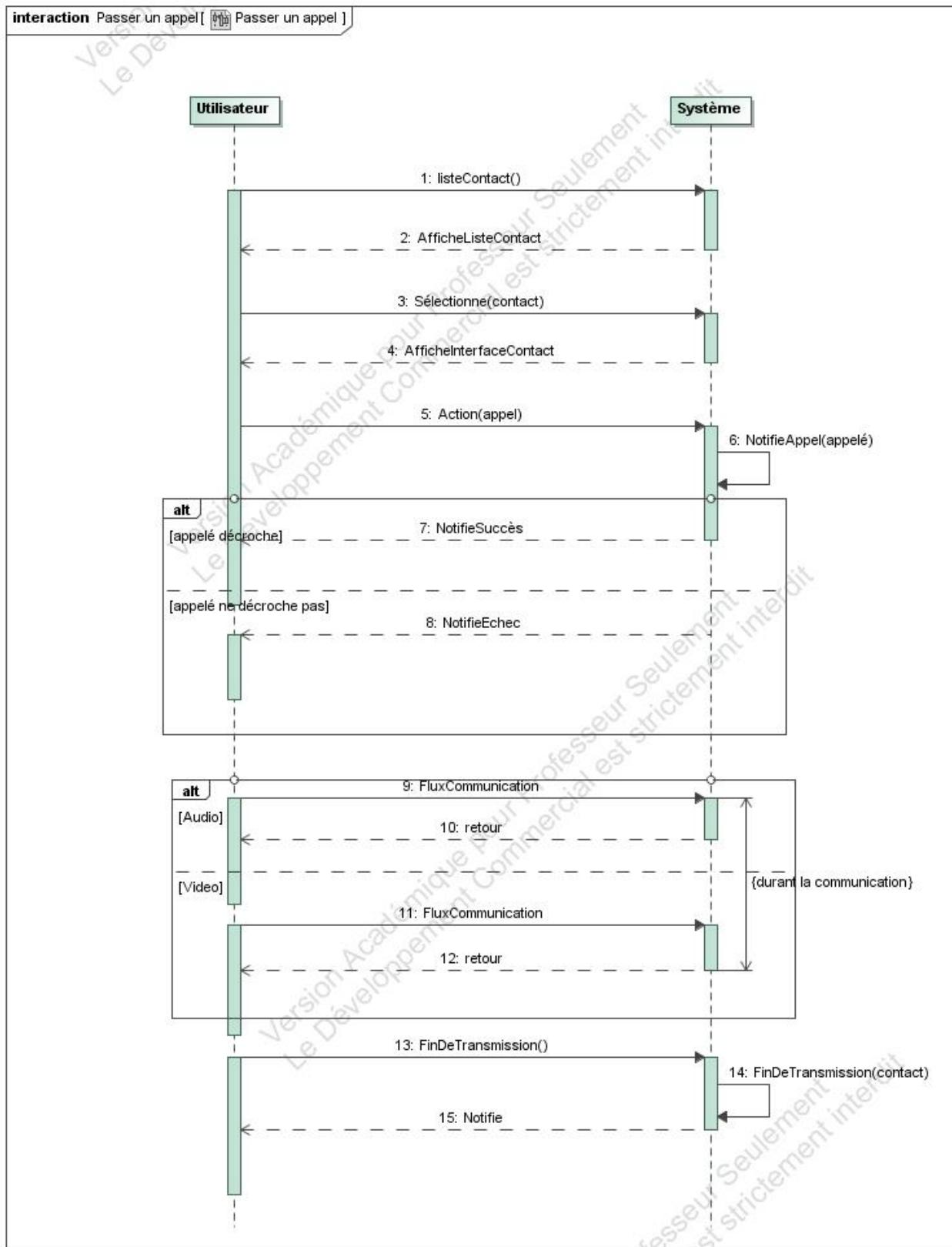


Figure 17 - diagramme de séquence système du cas d'utilisation passer un appel

Sous-package Gestion

Cas d'utilisation Archiver une conversation

Description textuelle

Acteur principal :

Un utilisateur.

Objectif :

Archiver la conversation afin que personne ne puisse plus poster de message.

Précondition :

Être authentifié.

Postcondition :

La conversation est archivée.

Scénario nominal

1. L'utilisateur demande la liste des conversations.
2. Le système envoie la liste des conversations.
3. L'utilisateur choisit la conversation à archiver et demande au système d'archiver.
4. Le système passe l'état de la conversation à archivée.
5. Le système notifie l'utilisateur que la conversation a été archivée.

Diagramme de séquence

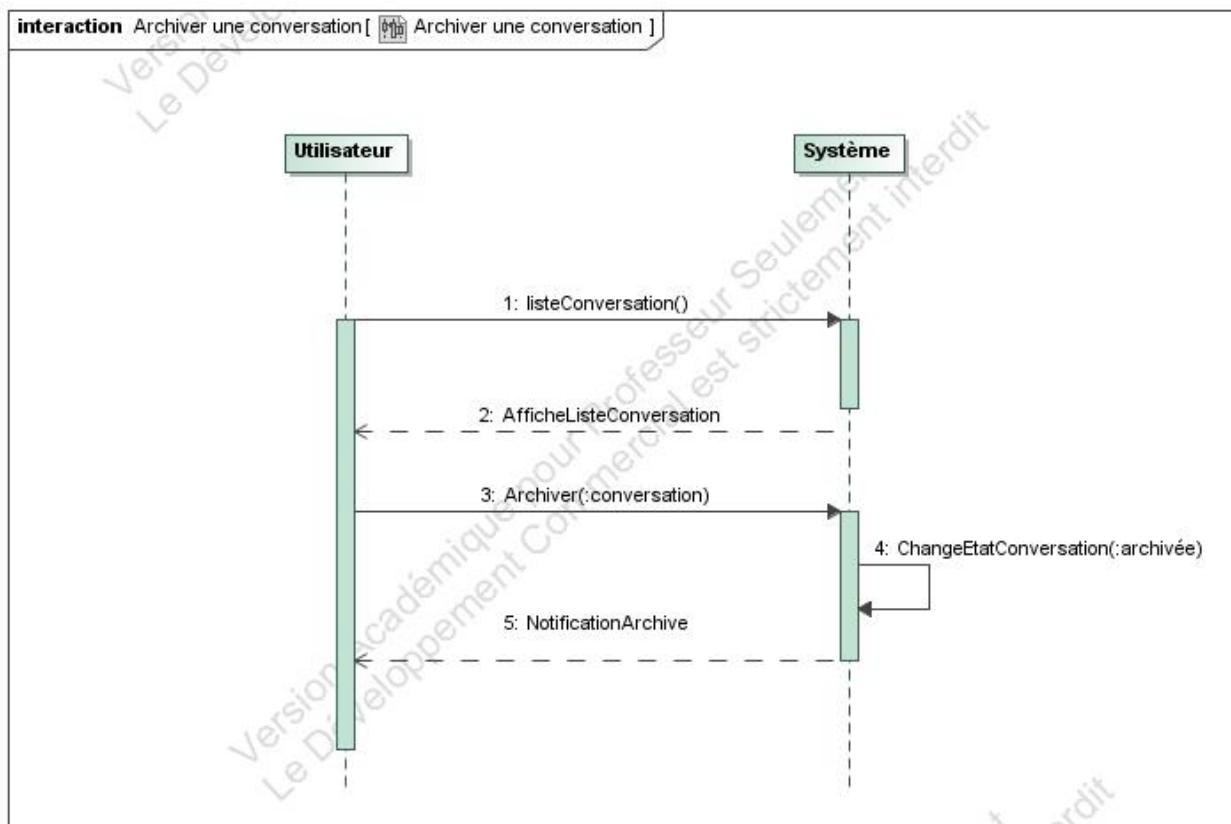


Figure 18 - diagramme de séquence du cas d'utilisation d'archivage d'une conversation

Cas d'utilisation Inviter un utilisateur

Description textuelle

Acteur principal :

Un participant à une conversation.

Objectif :

Inviter un utilisateur à rejoindre une conversation.

Précondition :

Participer à une conversation.

Postcondition :

L'utilisateur reçoit une invitation à se joindre à la conversation.

Scénario nominal

1. Le participant demande la liste des conversations.

2. Le système lui renvoie la liste des conversations auxquelles ils participent.
 3. Le participant choisit une conversation.
 4. Le système affiche la conversation.
 5. L'utilisateur demande à inviter quelqu'un
 6. Le système demande de lui indiquer l'identifiant
 7. L'utilisateur renseigne l'identifiant
 8. Le système envoie une invitation à l'utilisateur identifié par l'identifiant.
 9. Le système indique à l'utilisateur que l'invitation a été envoyée.

Erreurs

8a. Le système ne parvient pas à identifier l'utilisateur à inviter.

1. Le système avertit l'utilisateur qu'il n'y a pas d'utilisateur possédant l'identifiant renseigné.

LE CAS D'UTILISATION SE TERMINE EN ECHEC.

Diagramme de séquence système

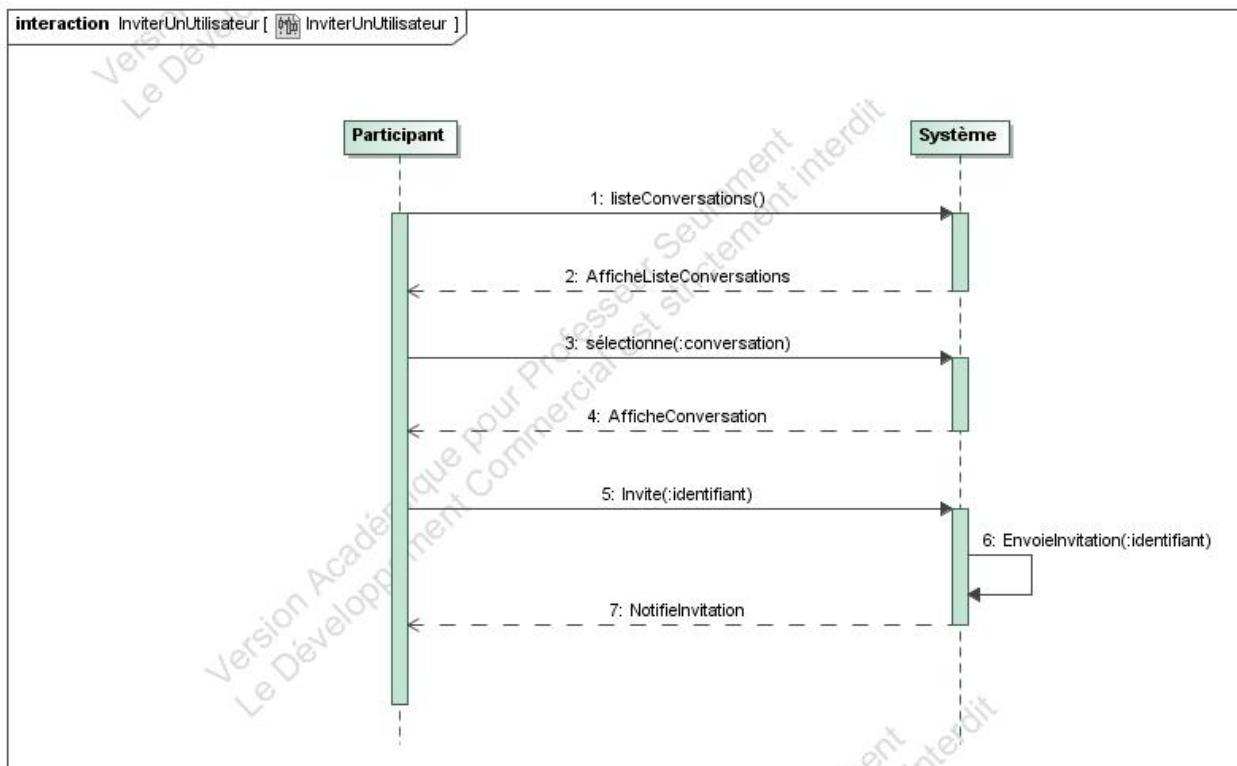
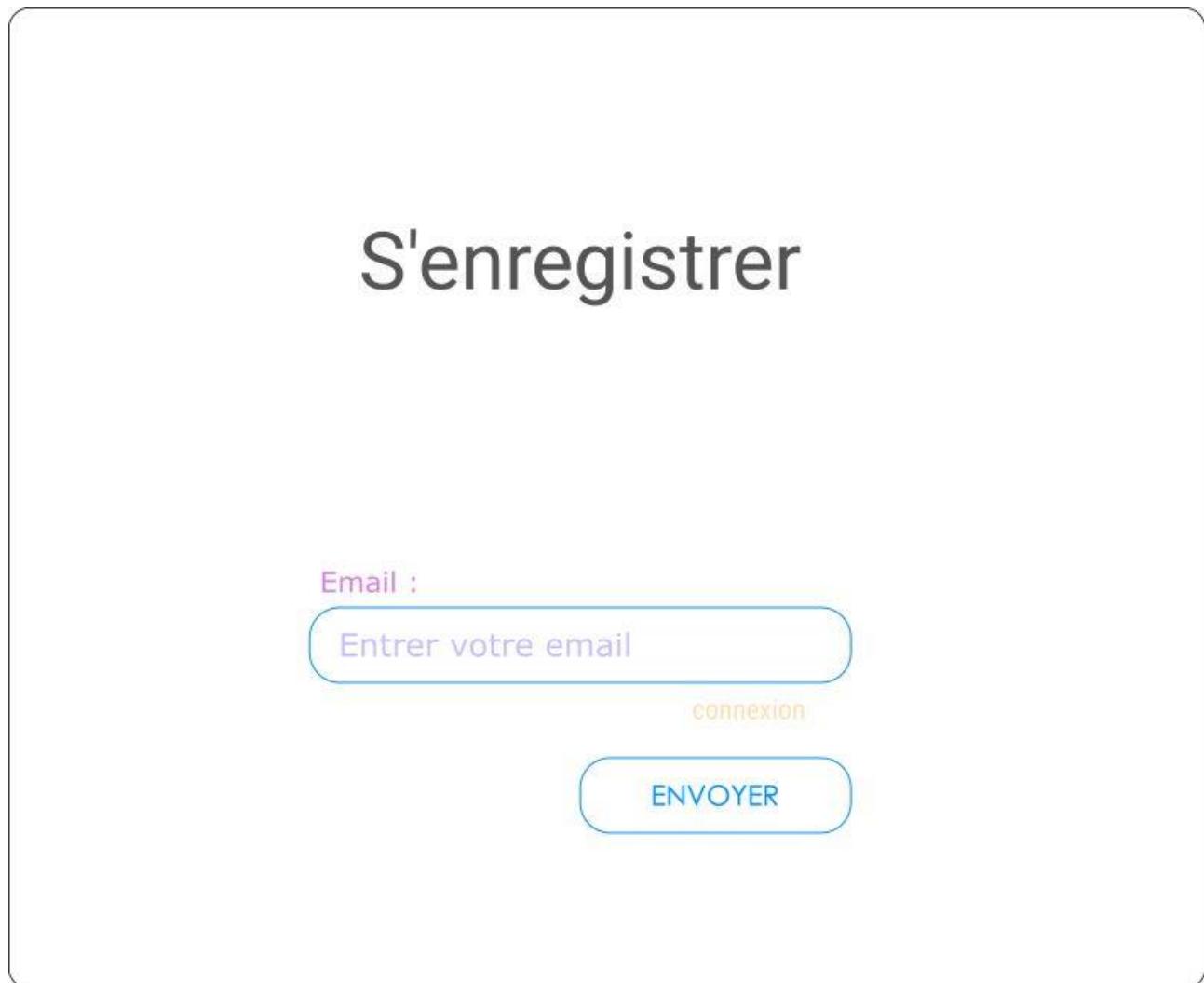


Figure 19 - diagramme de séquence du cas d'utilisation inviter un utilisateur

Maquette des interfaces :

Interface d'enregistrement :



Maquette de l'interface d'enregistrement. L'écran affiche le titre "S'enregistrer" en gros caractères. En dessous, il y a un champ de saisie pour l'email, étiqueté "Email :" et contenant la placeholder "Entrer votre email". En dessous du champ, il y a un bouton "ENVOYER" et un lien "connexion".

Interface de connexion :

Connexion

Identifiant :

Entrer votre identifiant

Mot de passe :

Entrer votre mot de passe

[mot de passe oublié](#)

VALIDER

Interface Profil :

Profil



Nom :
Nouailhaguet

Prénom :
Luc

Email :
luc.nouailhaguet@gmail.com

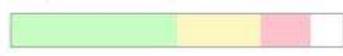
Nom d'utilisateur :
Zorglub

Mot de passe ::
***** 

METTRE A JOUR

Identifiant
FRGT-TTGD-3421-VVSR

Espace disque : **85%**



Stockage



Contacts



Conversations

Notifications :



Notification Bureau

 **ACTIVE**

Notification Email

 **INACTIVE**

Interface stockage


Nom d'utilisateur

Stockage 85%





Images

Textes

Compressés

Musiques

Vidéos

déposez vos fichiers ou cliquez

trier par
DATE
ordre

Nom affiché

Nom sur le disque

Nom affiché

Nom sur le disque

Nom affiché
Nom sur le disque
Nom affiché
Nom sur le disque

VALIDER
ANNULER

Nom affiché

Nom sur le disque

Interface Carnet d'adresse

[Date]

38



Nom d'utilisateur



Nom du contact



Nom d'utilisateur

date d'ajout



Nom d'utilisateur

date d'ajout



Nom d'utilisateur

date d'ajout



Interface des conversations

The interface is a digital communication platform. At the top left is a user profile icon with a beard and a blue 'Nom d'utilisateur' field. To the right is a 'INVITATIONS' section with a '3' badge, an 'ARCHIVES' section, and a prominent 'NOUVELLE CONVERSATION' button. The main area displays a conversation with 'Utilisateur1' and a message from 'Vous'. Below this, there are three more conversation entries with message counts (5, 9, and 1). A central input field says 'Tapez votre message ici.' with a send icon. On the right, there are sharing icons.

Nom d'utilisateur

INVITATIONS 3

Utilisateur1 Bonjour, Comment allez-vous ? il y a 1 minute

ARCHIVES

VOUS Bonjour, très bien merci ! maintenant

NOUVEL UTILISATEUR

Titre de la conversation 5

Titre de la conversation 9

Titre de la conversation

Tapez votre message ici.

Interface de partage de contenu :



Nom d'utilisateur

INVITATIONS 3

ARCHIVES

NOUVELLE CONVERSATION

Titre de la conversation

Nom d'utilisateur

Nom d'utilisateur

NOUVEL UTILISATEUR

Titre de la conversation 5

Titre de la conversation 9+

Titre de la conversation

Entrez le nom du fichier

ANNULER

Image icon

Document icon

File icon

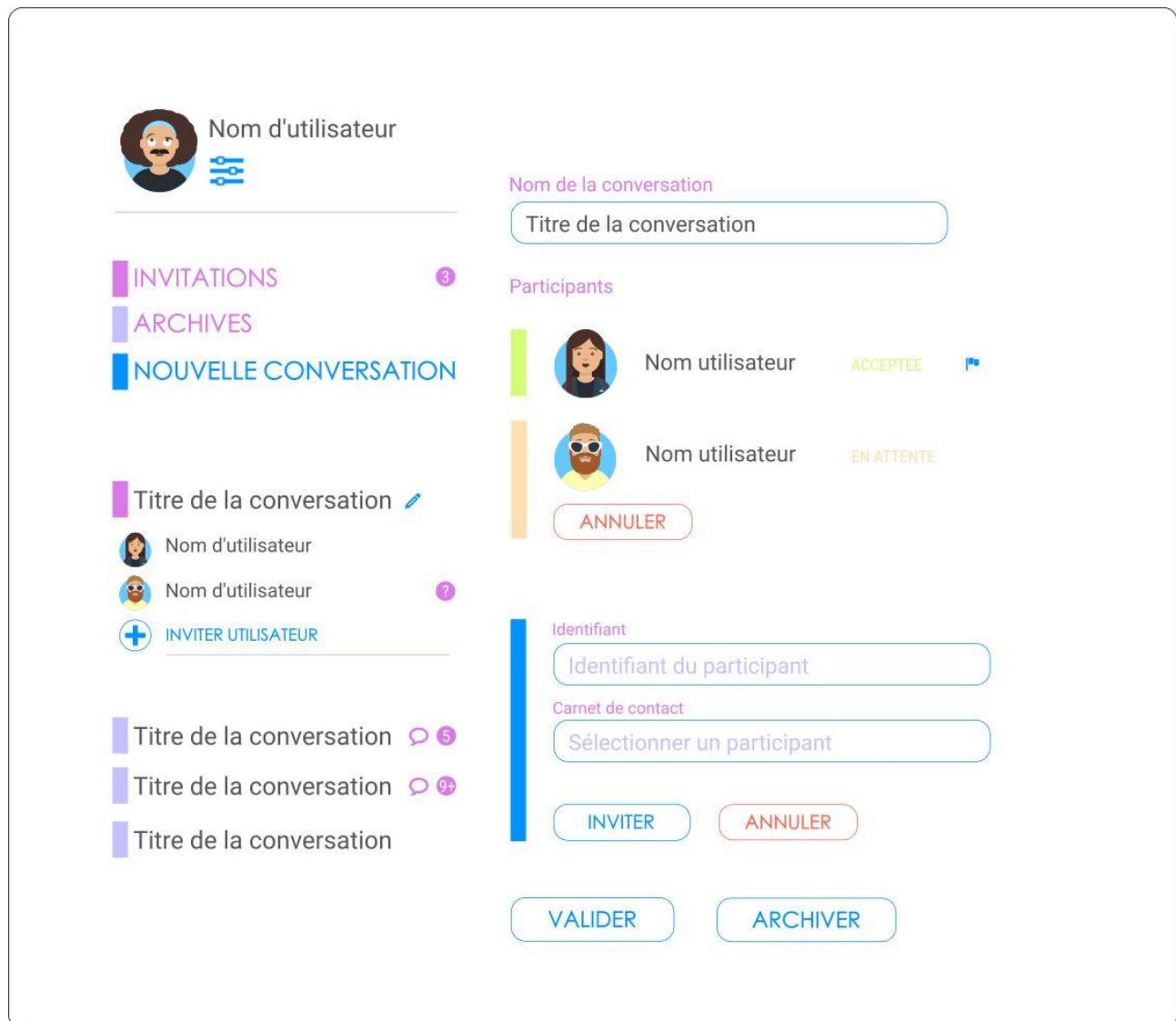
Music icon

Video icon

Nom du fichier

Nom sur le disque

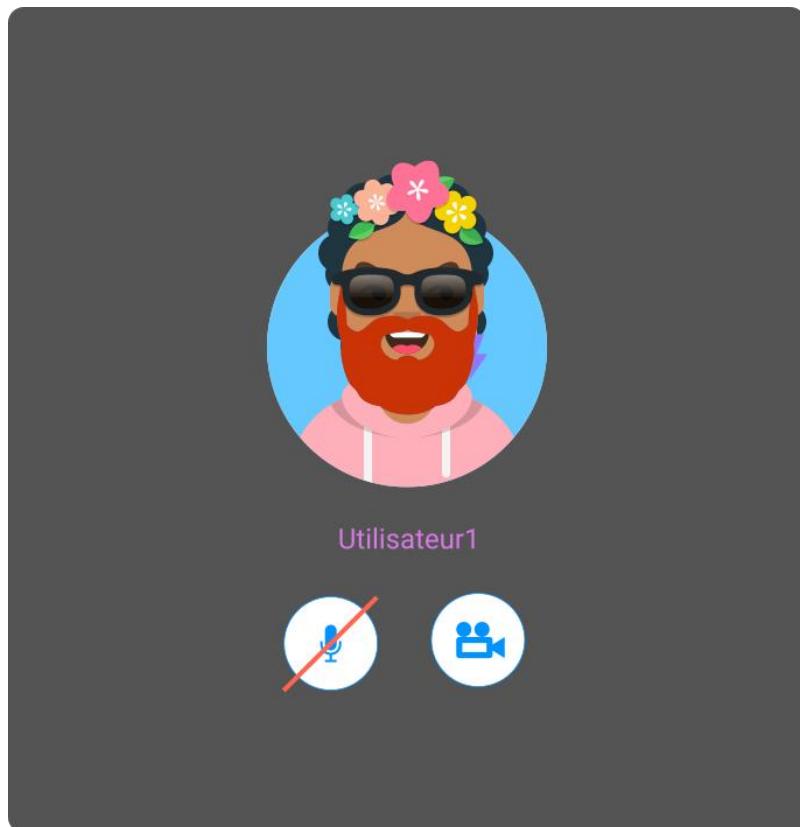
Interface Edition/création de conversation :



Interface d'invitation

The interface is a communication application. At the top left is a user profile with a blue background and a black icon. To its right is the text "Nom d'utilisateur" and a gear icon. Below this are three navigation buttons: "INVITATIONS" (purple), "ARCHIVES" (purple), and "NOUVELLE CONVERSATION" (blue). A small purple circle with the number "3" is positioned next to the "INVITATIONS" button. To the right of the navigation buttons is another user profile with a blue background and a white icon. Next to it is the text "Nom d'utilisateur" and "Vous a invité dans Titre de la conversation". Below this is a text box containing the message "Je ne peux pas, j'ai piscine." with two buttons below it: "PARTICIPER" (blue) and "DECLINER" (red). On the left side of the main content area, there is a list of conversation titles: "Titre de la conversation", "Titre de la conversation 5", "Titre de la conversation 99", and "Titre de la conversation". To the right of this list is another user profile with a blue background and a white icon. Next to it is the text "Nom d'utilisateur" and "Vous a invité dans Titre de la conversation". Below this is a text box containing the message "Commentaires" with two buttons below it: "PARTICIPER" (blue) and "DECLINER" (red).

Interface de communication :



Application de la méthode UWE :

Diagramme de contenu :

Pour éviter d'encombrer le diagramme de contenu, les opérations triviales de lecture/écriture des attributs de classe ne sont pas présentées.

Utilisateur :

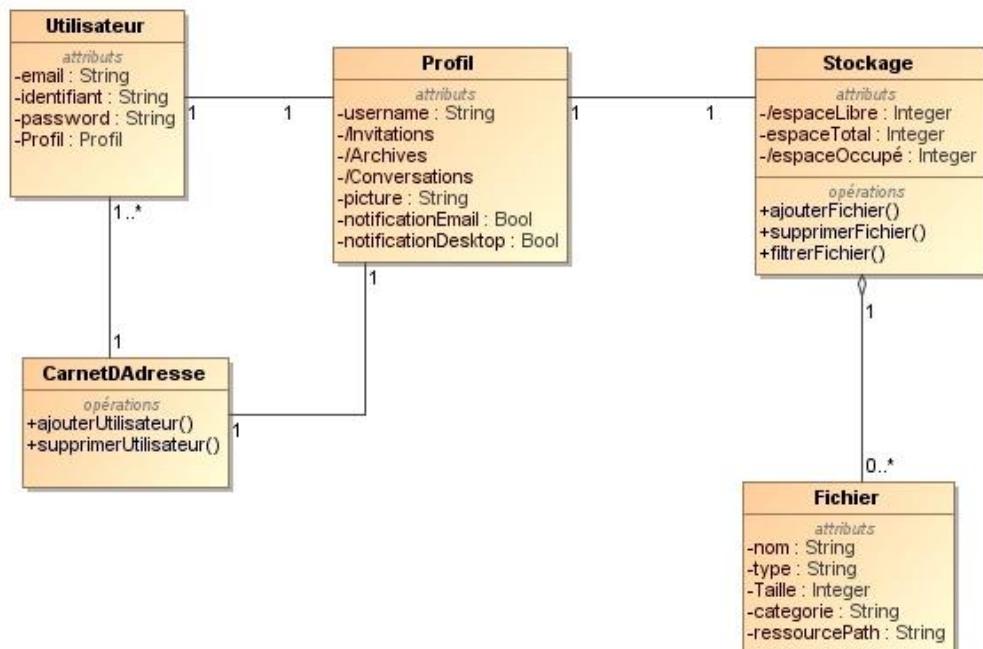


Figure 20 - diagramme de contenu pour l'acteur utilisateur du système

Propriétaire :

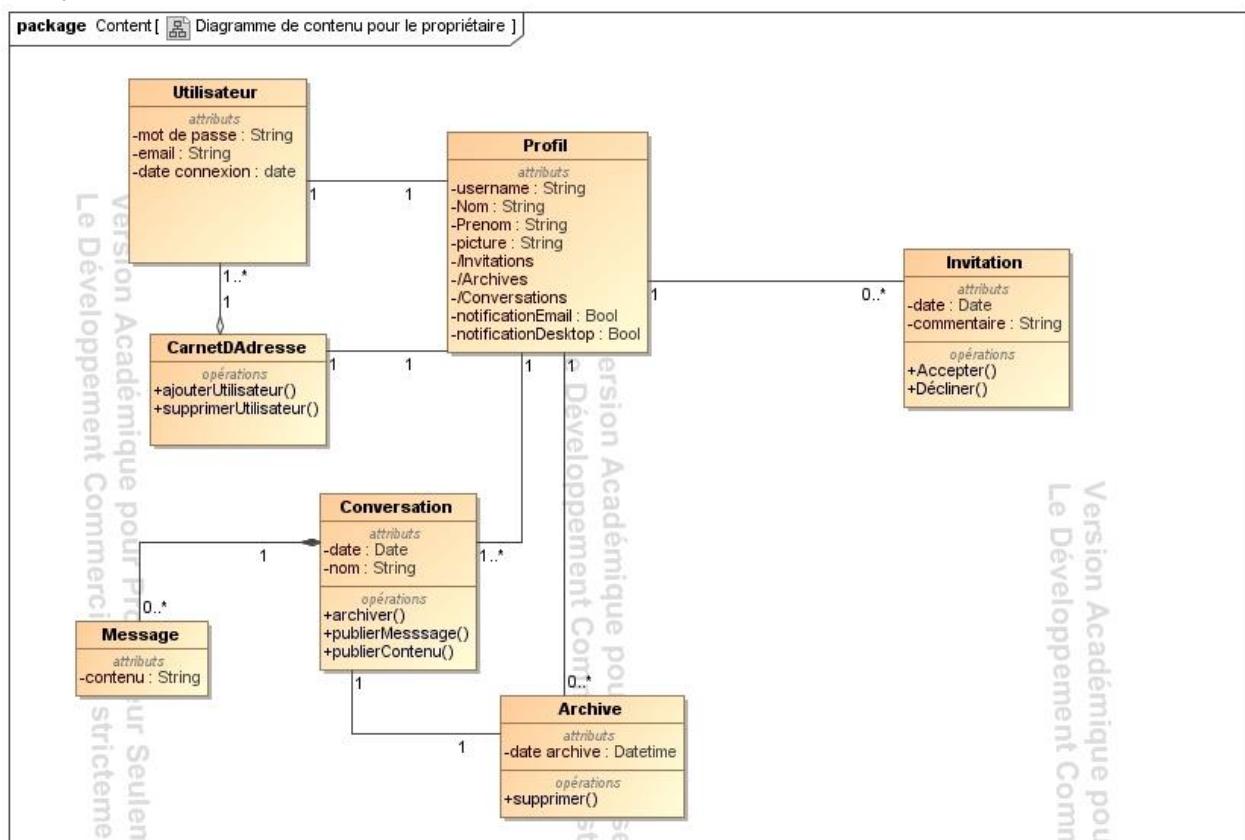


Figure 21 - diagramme de contenu pour l'acteur propriétaire

Participant :

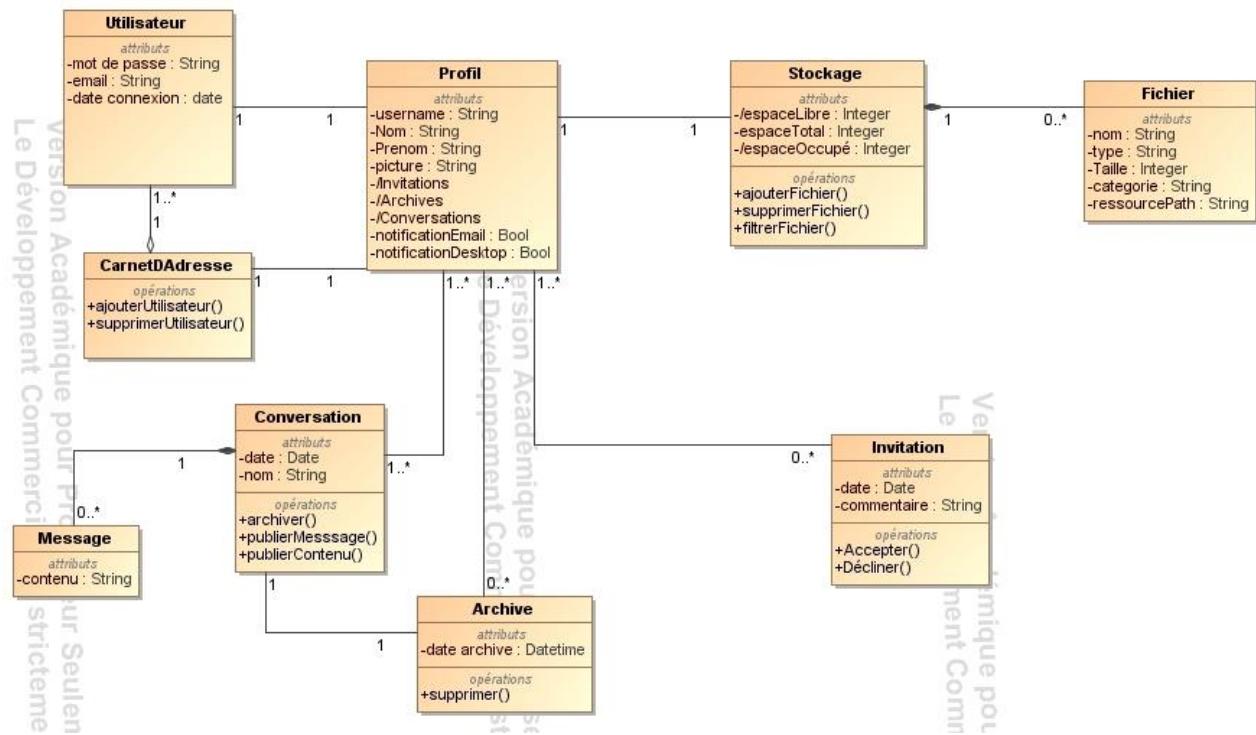


Figure 22 - diagramme de contenu pour l'acteur participant

La multiplicité de la relation Invitation-Profile, coté Profil, ne reflète pas exactement le modèle. Je n'ai pas trouvé comment spécifier une multiplicité fixé à un nombre.

La multiplicité réelle est de 2.

Diagrammes de navigations

En vertu des transformations disponibles dans la méthode UWE et les multiplicités relationnelles présentes sur les diagrammes de contenu pour les différents acteurs, on aboutit aux diagrammes de navigation suivants

Utilisateur :

Diagramme de navigation :

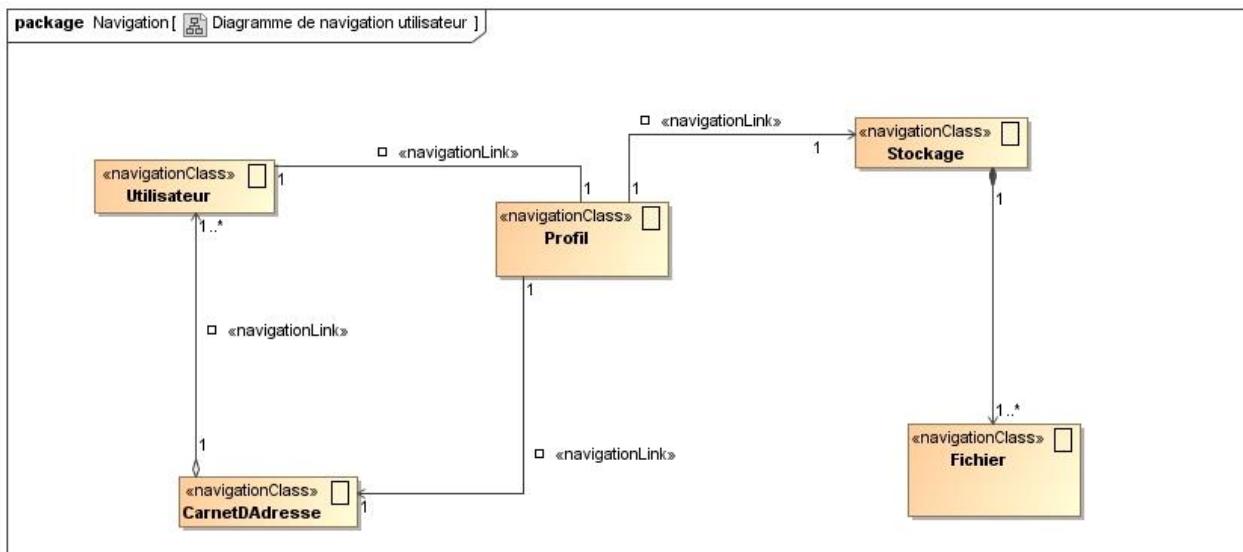


Figure 23 - diagramme de navigation de l'acteur utilisateur

Diagramme d'accès :

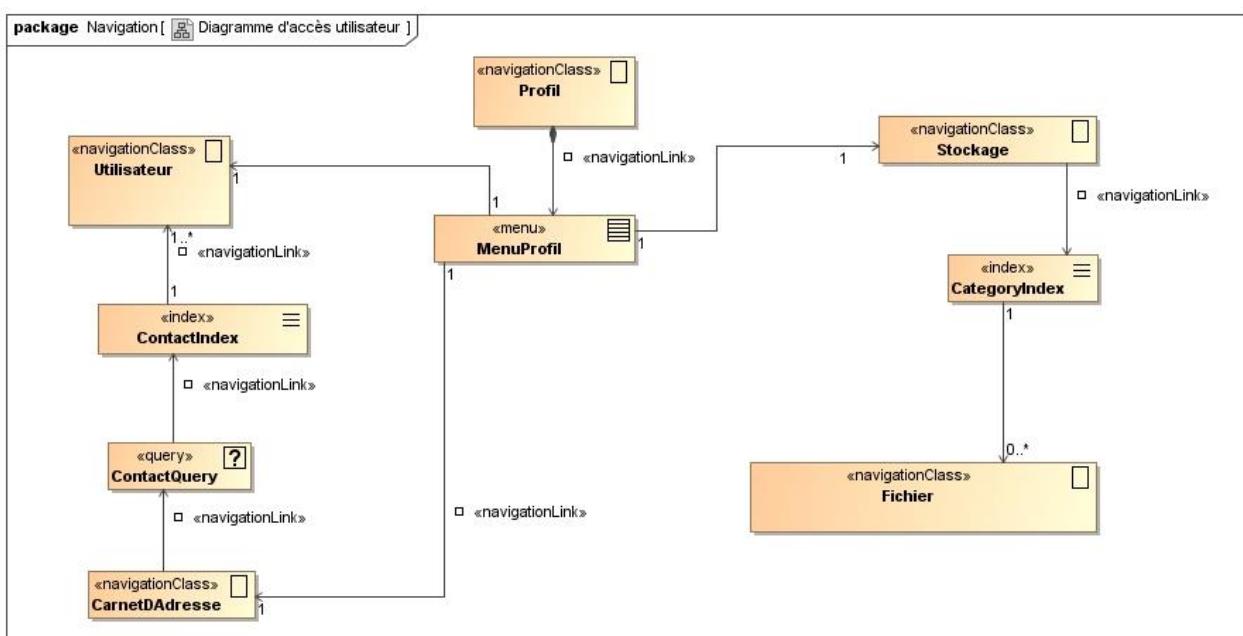


Figure 24 - diagramme d'accès de l'acteur utilisateur

Le profil permet l'accès au :

- Stockage
- Carnet d'adresse
- Utilisateur

Grâce au menu MENUPROFIL.

Ce motif d'accès sera réemployé dans les diagrammes d'accès des autres acteurs.

Propriétaire :

Diagramme de navigation :

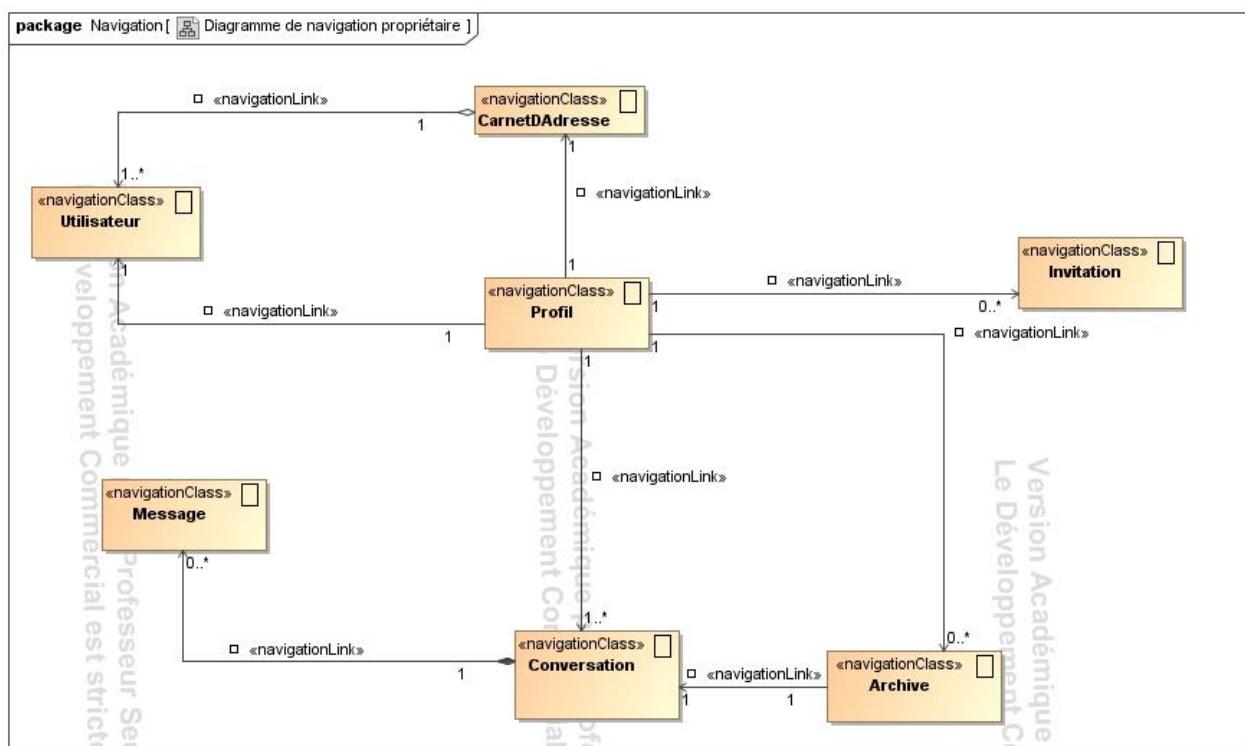


Figure 25 - diagramme de navigation de l'acteur propriétaire

Le propriétaire d'une conversation peut envoyer des invitations à participer aux conversations qu'il crée.

Le propriétaire d'une conversation peut archiver celle-ci.

Bien qu'il ne soit pas indispensable de faire figurer les messages puisqu'ils concernent l'acteur participant, il m'a semblé que l'ajouter était opportun en vertu du fait d'un propriétaire peut accéder à ses propres messages.

Diagramme d'accès :

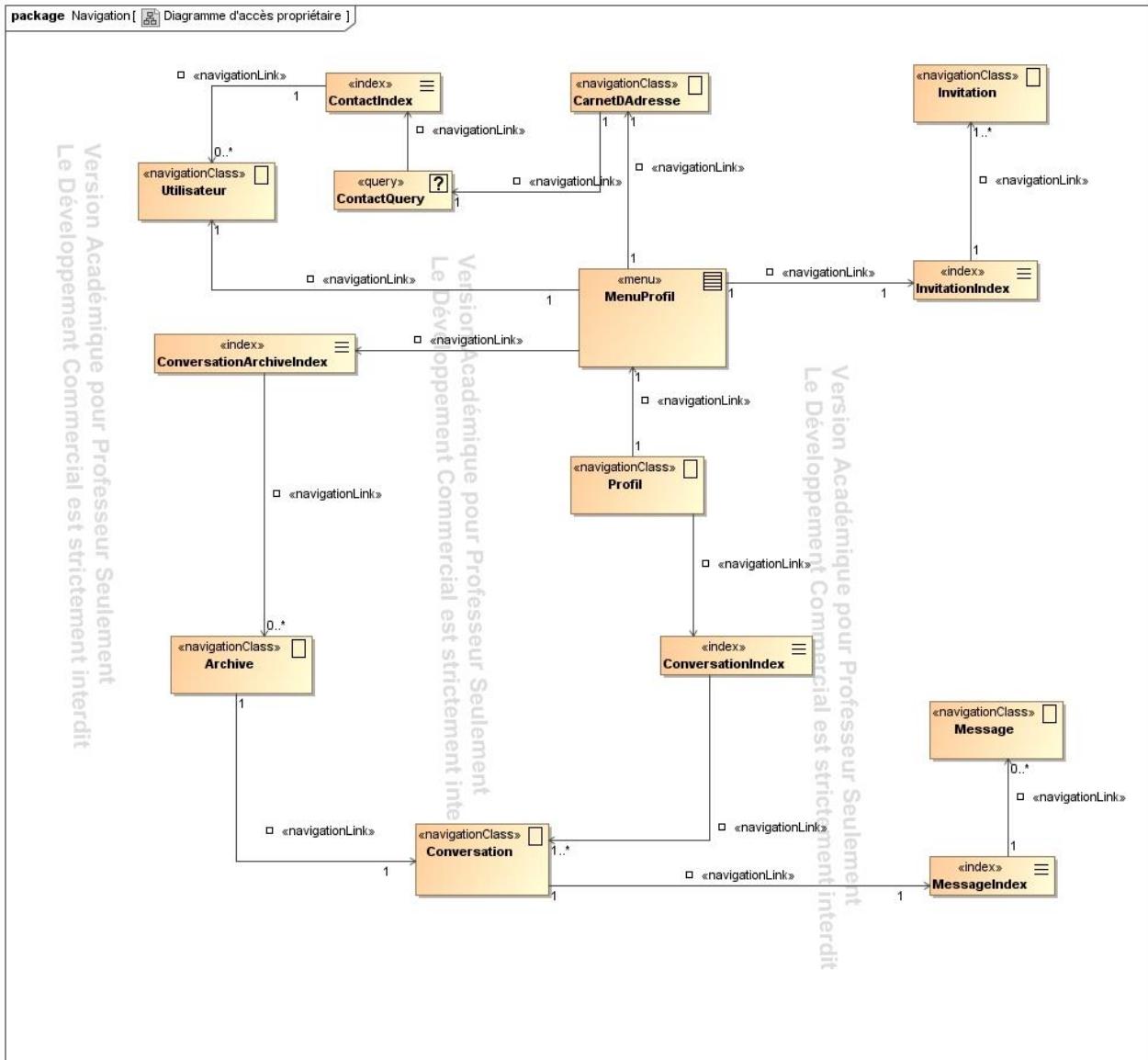


Figure 26 - diagramme d'accès de l'acteur propriétaire

Les conversations peuvent être archivées, leur propriétaire y accède grâce à un index.

Chaque conversation permet l'accès aux messages via un index.

Les invitations envoyées sont également indexées.

Participant :

Diagramme de navigation :

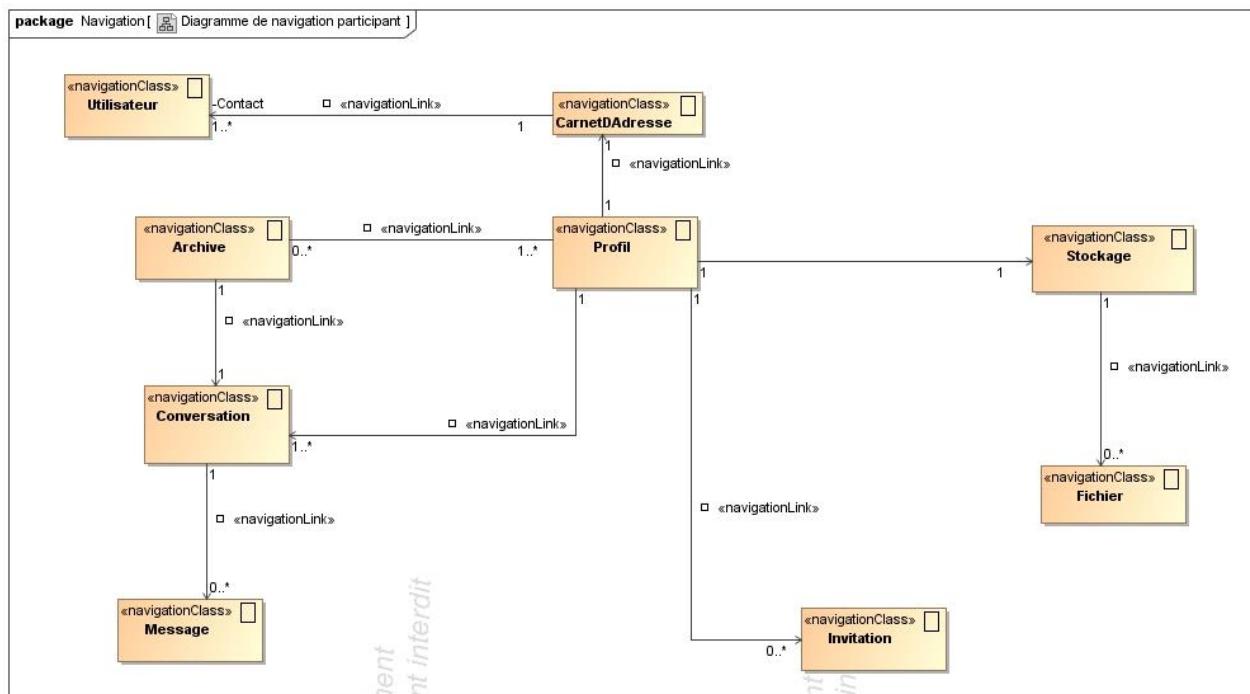


Figure 27 - diagramme de navigation de l'acteur participant

Diagramme d'accès :

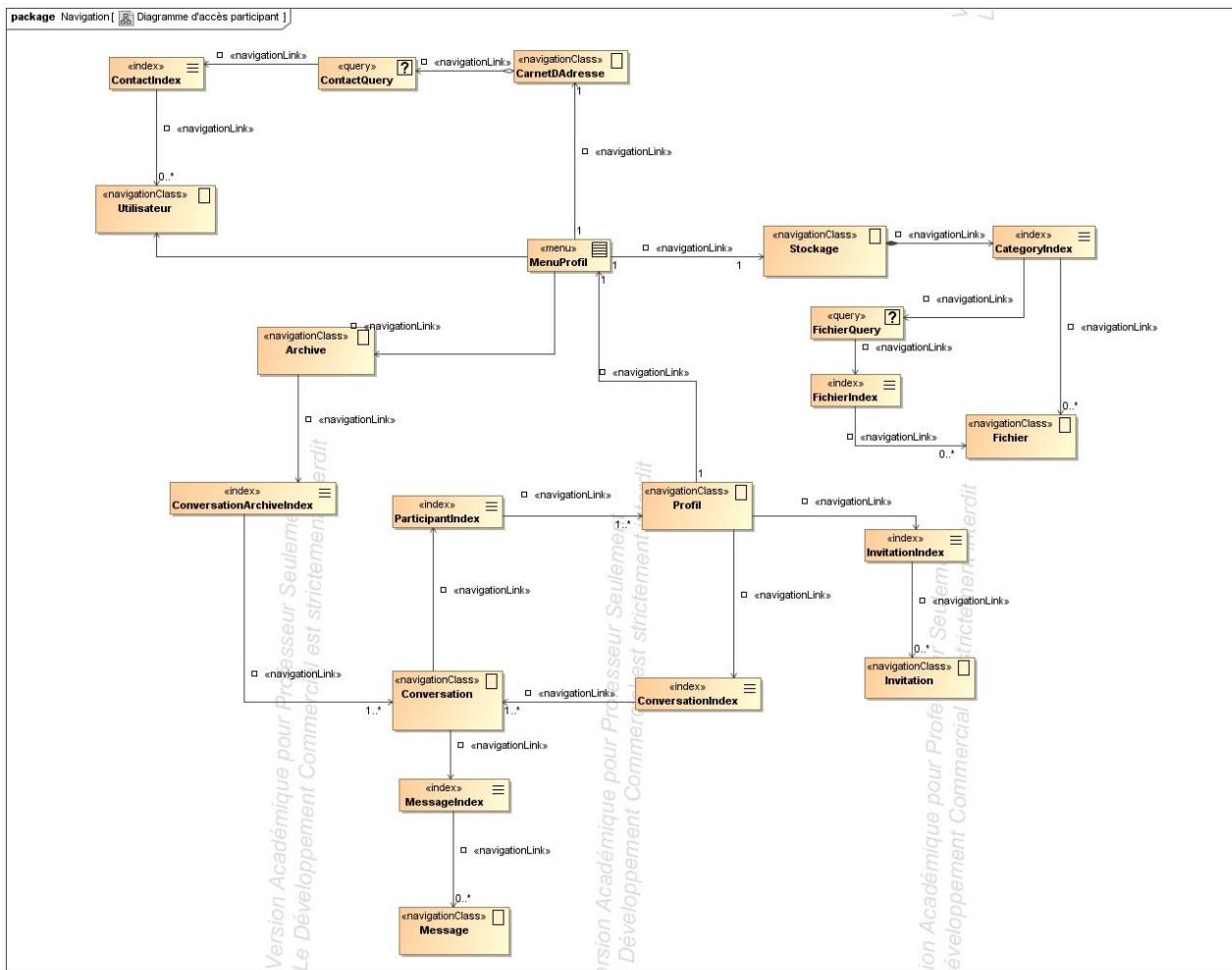


Figure 28 - diagramme d'accès de l'acteur participant

Diagrammes de présentation :

Les diagrammes de présentations sont particulièrement fournis aux vues des écrans que l'on souhaite réaliser en vertu des diagrammes de navigation précédemment spécifiés.

Ils seront présentés sous la forme d'un ensemble de pages de présentation (au sens du profil UWE) dans lesquelles les unités d'informations navigables requises pour la réalisation de la maquette associée (au sens des exigences métier) seront disposées.

Utilisateur :

Un utilisateur du système a accès à une page qui recense les informations de son profil. Ce contenu lui permet de naviguer vers les le stockage, et son carnet d'adresse par le biais d'ancres.

Page profil :

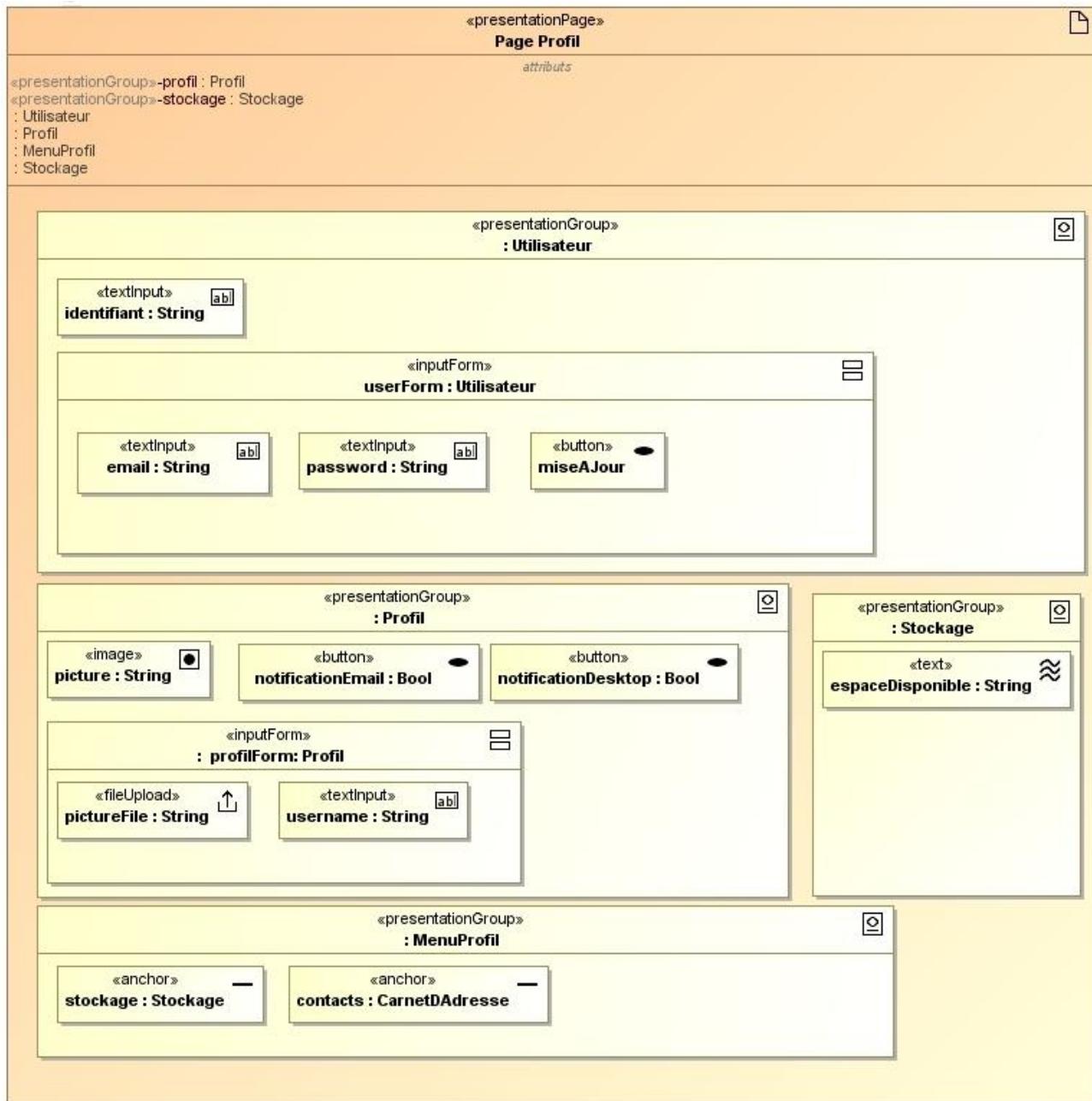


Figure 29 - diagramme de présentation de la page de profil de l'acteur utilisateur

Stockage

La page de présentation expose notamment une ancre vers la page de présentation des fichiers.

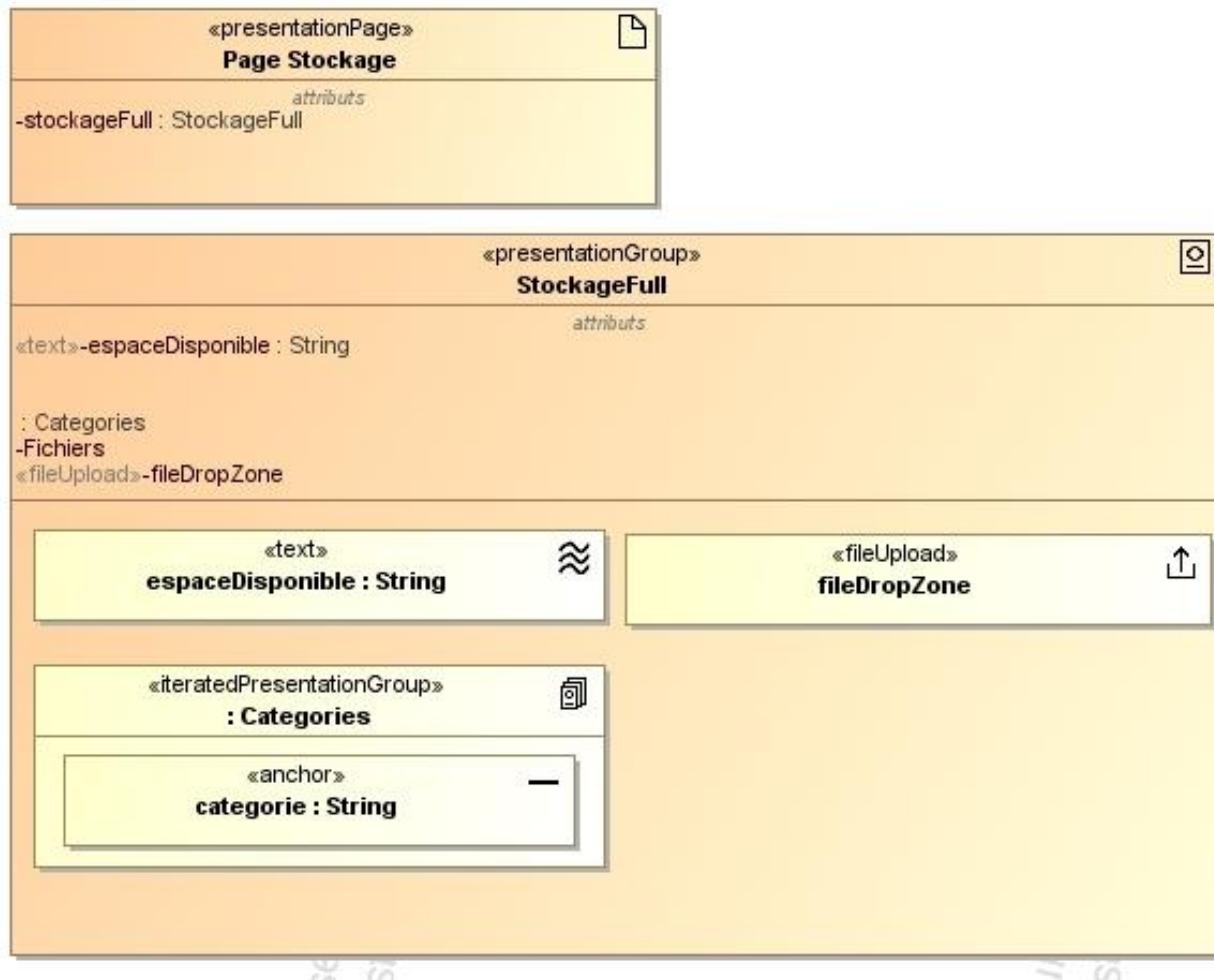


Figure 30 - diagramme de présentation de la page stockage

La collection des fichiers présentée sous forme de groupe permet de naviguer par une ancre vers l'unité d'information présentant le fichier.

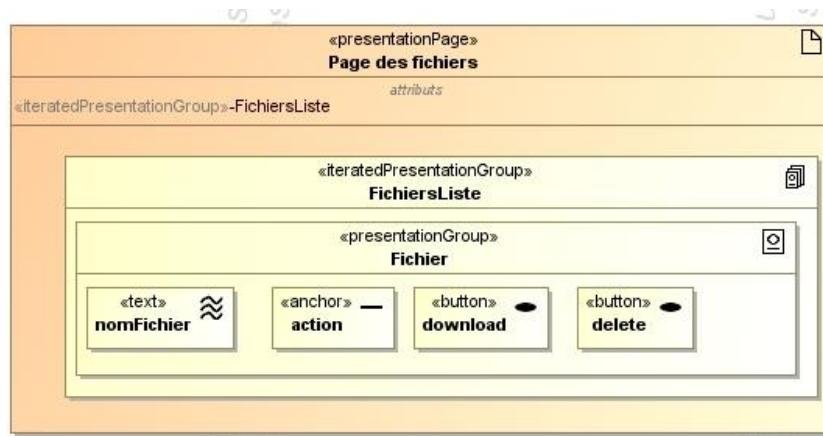


Figure 31 - diagramme de présentation de l'unité d'information fichier

Au sein duquel une ancre mène à une unité d'interface utilisateur permettant la modification du nom du fichier.

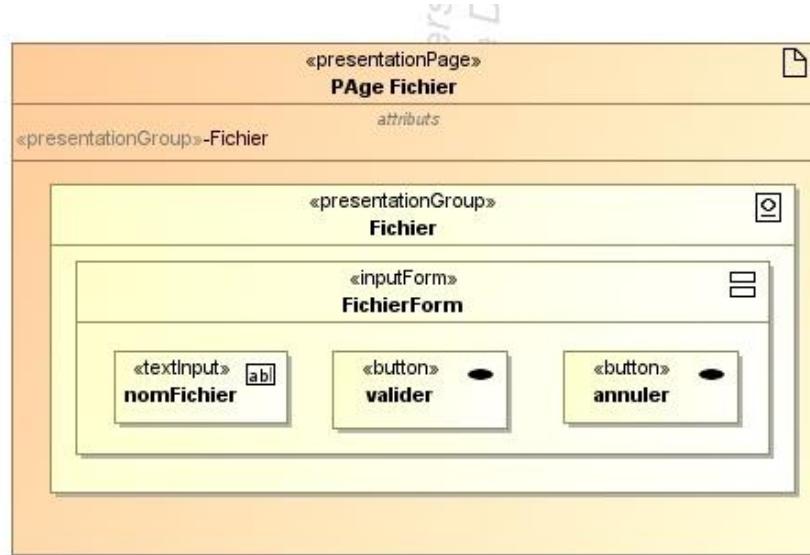


Figure 32 - diagramme de présentation de l'unité d'information interface d'interaction

Carnet d'adresse

L'ancre CARNETDADRESSE de la page de présentation profil permet l'accès à la page de présentation SELECTIONCONTACT.

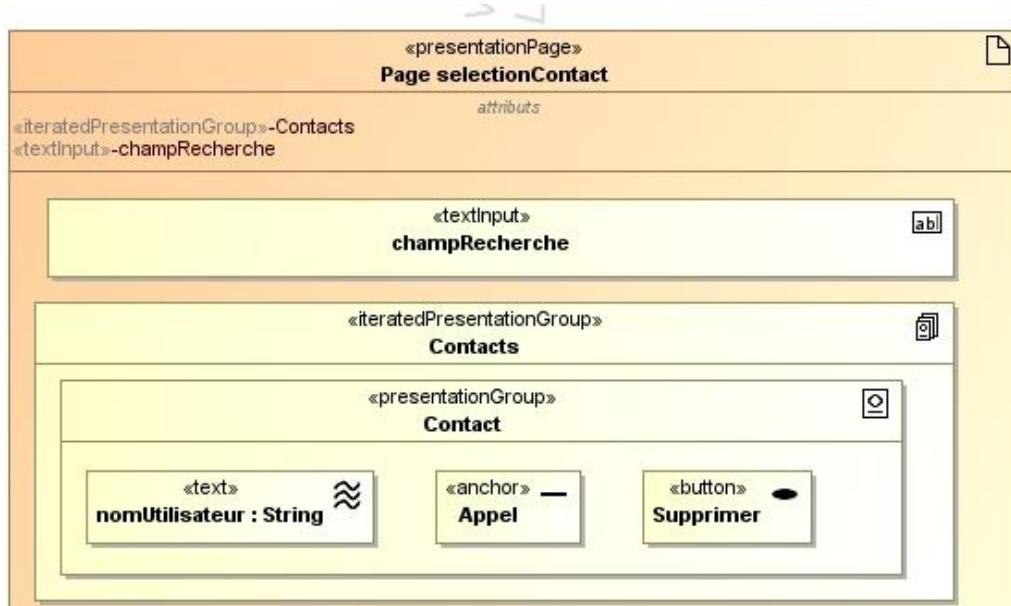


Figure 33 - page de présentation du carnet d'adresse

L'ancre APPEL mène à la page de présentation PAGEDECOMMUNICATION.

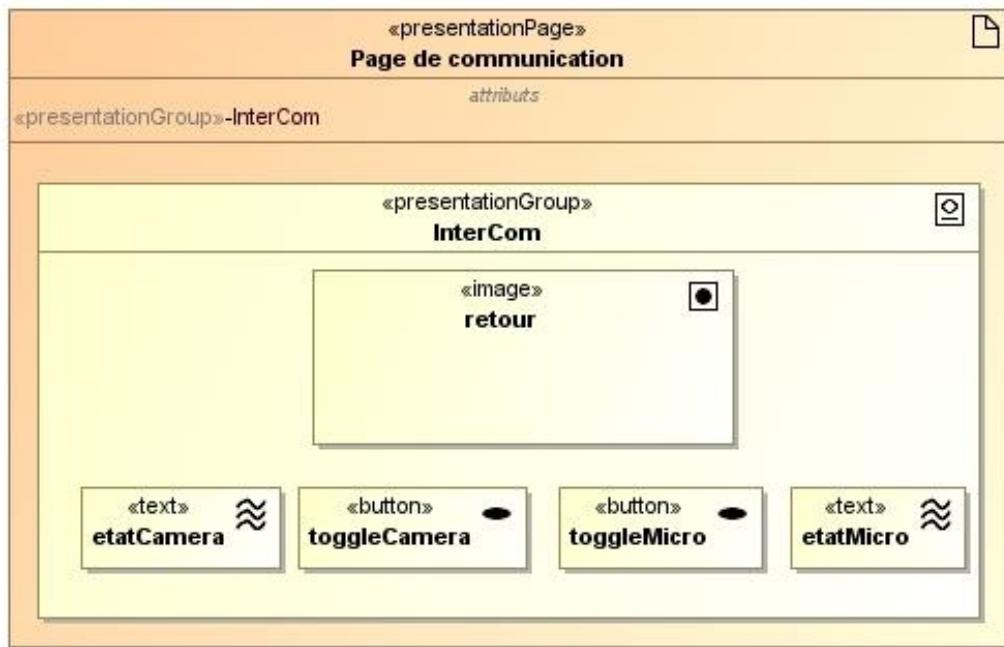


Figure 34 - page de présentation de la communication temps réel

Participant :

La page de présentation pour l'acteur participant diffère de celui de l'utilisateur dans la représentation du MENUPROFIL.

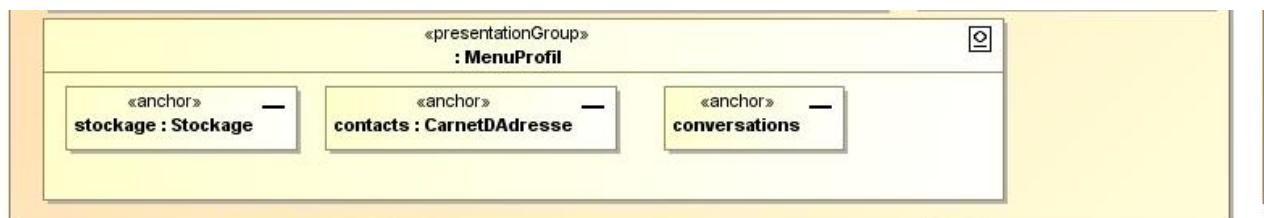


Figure 35 - groupe de présentation du menu profil pour l'acteur participant

L'ancre conversations permet d'accéder à la page de présentation CONVERSATIONS, dans laquelle sont présentées les conversations auxquelles le participant participe.

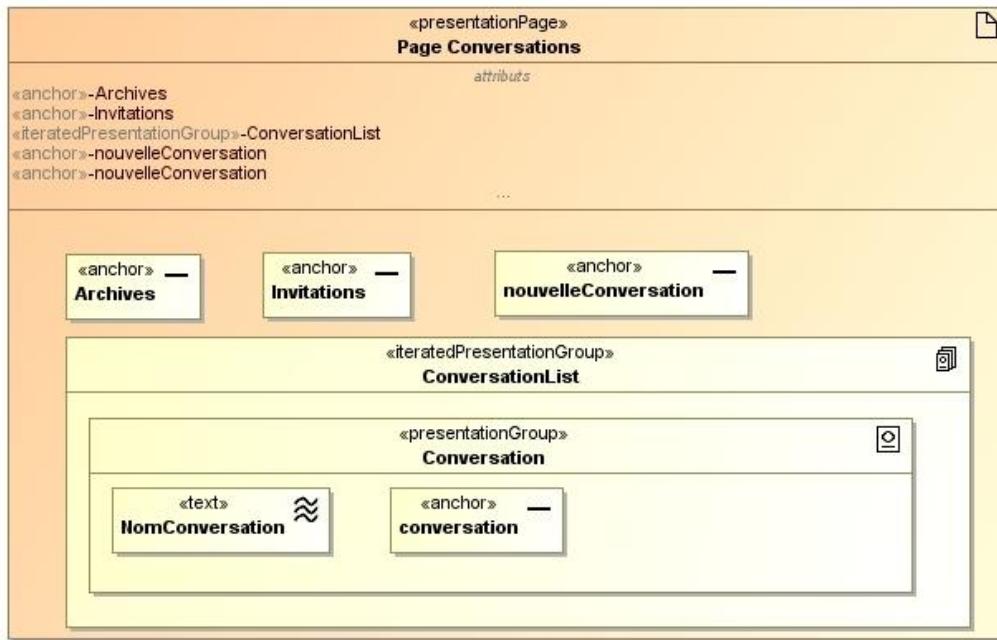


Figure 36 - page de présentation des conversations pour un participant

L'ancre NOUVELLECONVERSATION transforme l'acteur **participant** en l'acteur **propriétaire**.

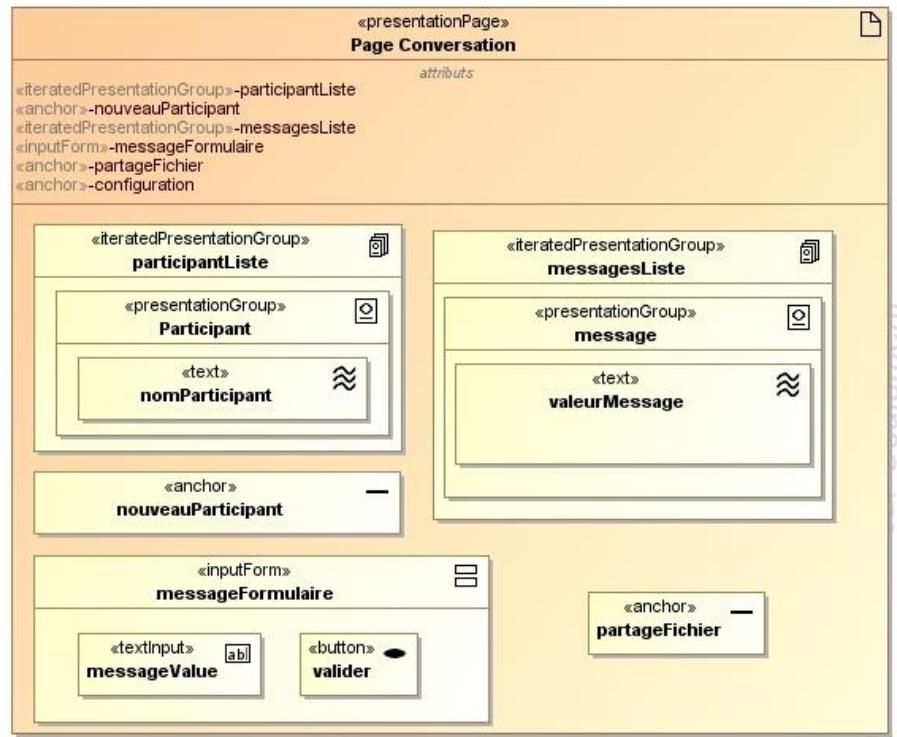


Figure 37 - page de présentation des conversations

Chaque conversation est accessible par l'ancre CONVERSATION qui mène à la page de présentation pour une conversation. L'ancre INVITATIONS permet d'accéder à la page de présentation des INVITATIONS.

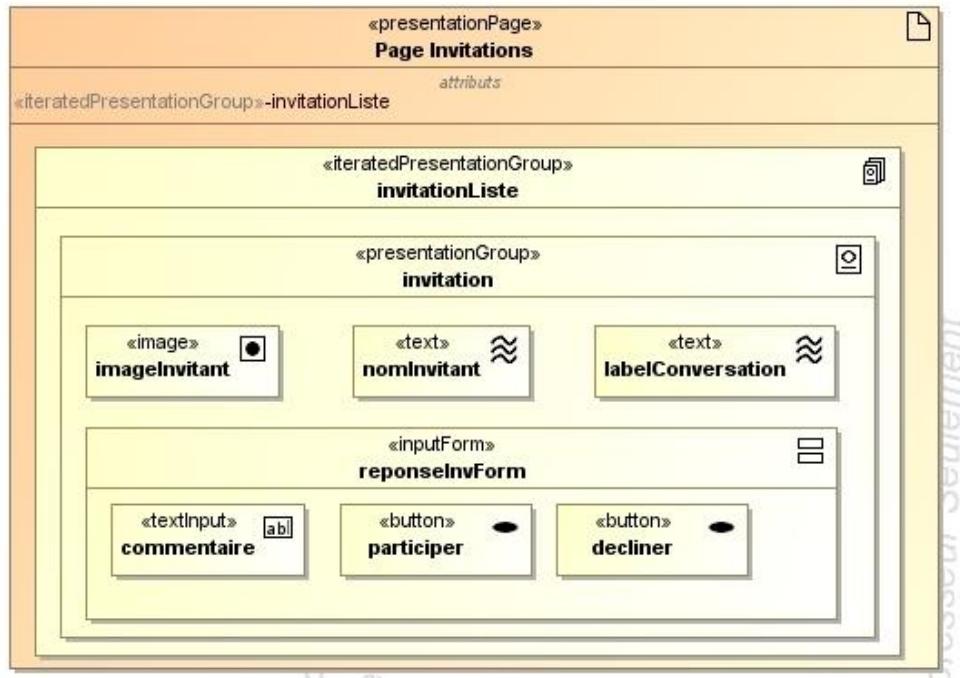


Figure 38 - page de présentation des invitations

Pour chaque conversation auquel un participant participe, il peut inviter un autre utilisateur en utilisant l'ancre NOUVEAUPARTICIPANT menant à la page de présentation NOUVEAUPARTICIPANT.

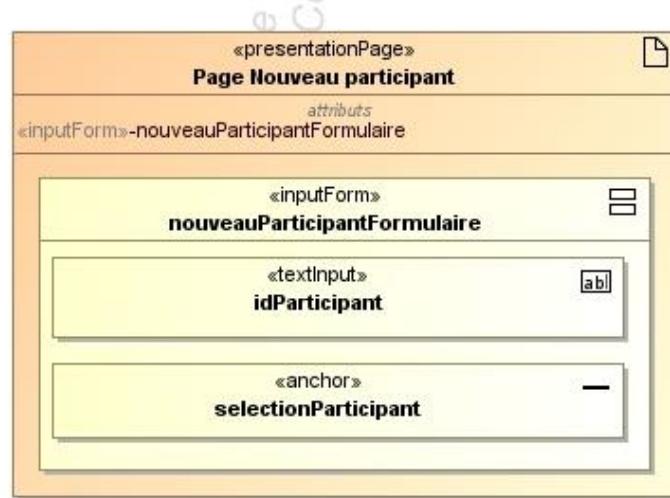


Figure 39 - page de présentation nouveau participant

Cette unité de présentation autorise la sélection d'un nouveau participant grâce à un code d'identification IDPARTICIPANT ou en se servant du carnet de contact. Le carnet de contact étant accessible par l'ancre SELECTIONPARTICIPANT.

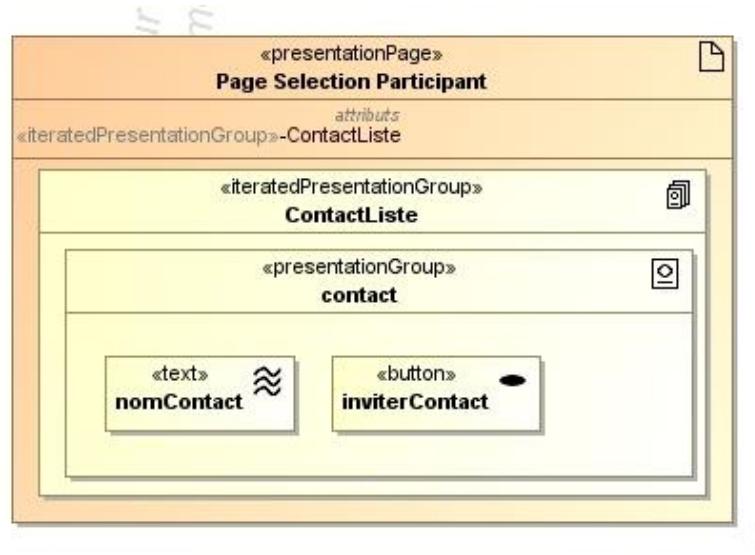


Figure 40 - page de présentation du carnet d'adresse lors d'une invitation à participer à une conversation

Le bouton INVITERCONTACT permet d'envoyer une invitation à un contact.

Le partage de contenu dans une conversation est disponible via la page de présentation PARTAGEFICHIER.



Figure 41 - page de présentation d'un partage de contenu

Disposant d'une ancre FICHIERSCATEGORIE renvoyant vers la page de présentation des résultats indexant les catégories de fichier.

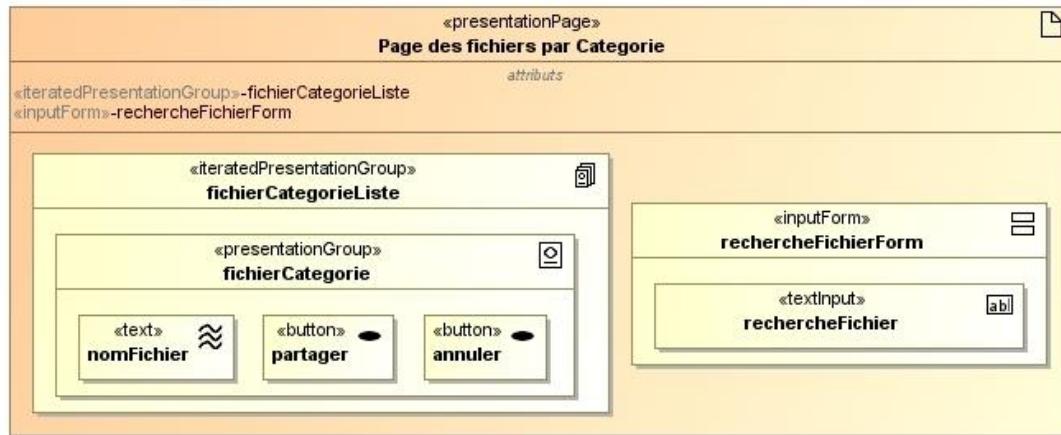


Figure 42 - page de présentation de la sélection de fichier par catégorie lors du partage de contenu

Le formulaire RECHERCHEFICHIERFORM agit comme un filtre sur les résultats par catégorie.

Les boutons PARTAGER/ANNULER permettent respectivement de partager le contenu ou annule l'action de partage en cours.

Propriétaire :

L'ancre NOUVELLECONVERSATION de la **figure 36** permet à tout participant de devenir propriétaire.

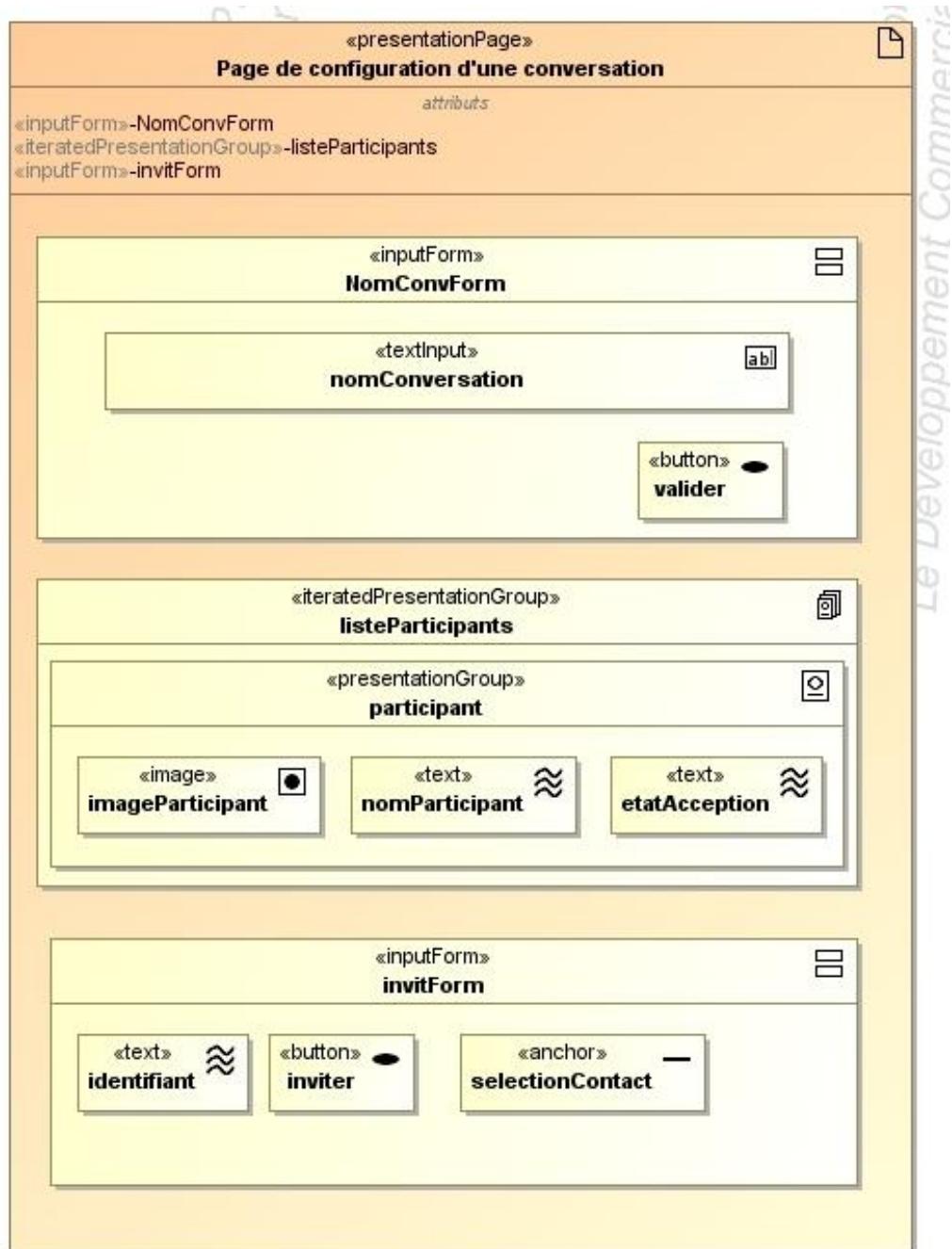


Figure 43 - page de présentation d'une nouvelle conversation

L'ancre SELECTIONCONTACT renvoie à la page de présentation SELECTIONPARTICIPANT de la figure 38.

Implémentation :

Présentation :

Pour l'implémentation, j'ai choisi le framework Symfony codé en PHP.

Le DAO sera géré par l'ORM Doctrine à travers une extension du framework qui l'intègre parfaitement.

Le framework permet l'inversion de responsabilité par injection de dépendance via un container de service.

C'est via ce container de service que seront manipulés :

- Le gestionnaire d'entités (responsable de la persistance)
- Le gestionnaire des uploads de fichiers

Afin d'être le plus agile possible, l'infrastructure aux fondations du code framework sera containerisé.

Pour mettre en place la solution, nous allons utiliser un pattern MVC. La partie communication asynchrone sera gérée via le protocole mercure. La partie communication temps réel en utilisant la spécification WebRTC.

Note

Je n'ai pas eu le temps de faire l'implémentation complète, j'aurais notamment voulu avancer sur l'implémentation de la messagerie asynchrone en mercure et la mise en place de l'infrastructure Webrtc ainsi que l'interfaçage.

J'aurais également souhaité effectuer une implémentation Domain Driven Design avec une base architecturale orientée événement en me servant du composant Messenger de Symfony mais cela dépasse de loin le cadre du cours, y compris au niveau du temps requis pour tout mettre en place de manière propre.

Infrastructure :

L'infrastructure est composée de trois éléments :

- Un serveur web
- Le langage d'interprétation PHP interfacé par FCGI
- Un serveur de base de données

Le serveur web choisi est le serveur Nginx. L'image du service docker est l'image officiel du serveur. Le fichier de configuration du serveur se situe dans le répertoire nfe114Docker.

Le service qui se charge de l'interprétation PHP est un service dont l'image a été construite pour satisfaire les dépendances aux extensions suivantes :

- Intl
- Opcache

- Pdo_mysql
- Apcu
- Xdebug
- Zip

La base de l'image est l'image officielle php :7.3-fpm.

Le moteur de base de données sera hébergé par le service officiel mariadb.

https://hub.docker.com/_/nginx
https://hub.docker.com/_/php
https://hub.docker.com/_/mariadb

Le tout communiquant sur un réseau virtuel dont l'hôte est le logiciel docker. <https://www.docker.com/>

La mise en place de l'infrastructure sur la plateforme de développement s'effectue grâce au logiciel docker-compose par le biais d'un fichier de configuration à syntaxe déclarative. <https://docs.docker.com/compose/>

```
/usr/local/bin/docker-compose -f /Users/luc/Workspace/symfony/nfe114-project/nfe114Docker/docker-compose.yml up -d
Docker Compose is now in the Docker CLI, try `docker compose up` 

Starting nfe114docker_database_service_1 ... done
Starting nfe114docker_php_fpm_1      ... done
Starting nfe114docker_serveur_web_1 ... done
```

Illustration 1 - Service d'infrastructure en route

Le framework :

L'installation du framework **Symfony** s'effectue grâce à l'application **composer**.

C'est une application qui permet d'intégrer des paquets PHP servant de dépendance à un code base. Le repository principal de composer est **packagist**.

Composer permet en outre la résolution des dépendances au moyen de la construction d'un graphe de dépendances. Il permet aussi de centraliser l'inclusion de tous les paquets en un point à travers un mécanisme d'auto-chargement à la volée conforme à la spécification PSR-4.

Le fichier de configuration est un fichier déclaratif que l'on peut soit modifier avec un éditeur de texte soit à travers les commandes que composer propose.

<https://getcomposer.org/>
<https://packagist.org/>
<https://www.php-fig.org/psr/psr-4/>

Parmi les manières de configurer le framework, j'ai retenu la manière des annotations et la manière yaml :

- Le mapping des routes avec les requêtes http et le mapping des objets avec les entités persistées sont configurées avec des annotations,
- Le reste de la configuration est défini dans le répertoire config, sous forme de fichiers déclaratifs yaml



Illustration 2 - fichiers de configuration

Les assets seront gérés de manière centralisée par l'empaqueteur **Webpack** à travers l'emballeur Symfony **Encore**. Les images, les fichiers de script javascript et les fichiers de style seront intégrés sous un format minifié, polyfillé au sein d'un seul fichier servi en fonction de la page pour éviter le gaspillage de bande-passante.

<https://fr.wikipedia.org/wiki/Polyfill>
<https://webpack.js.org/>
<https://babeljs.io/>
<https://www.npmjs.com/package/@symfony/webpack-encore>
<https://github.com/symfony/webpack-encore>
<https://www.npmjs.com/package/webpack>
<https://nodejs.org/fr/>

Webpack est un module **nodejs**, il vous faudra donc installer **nodejs**, ainsi qu'un gestionnaire de paquets node : npm ou **Yarn** par exemple. Pour ma part, j'utilise **Yarn**.

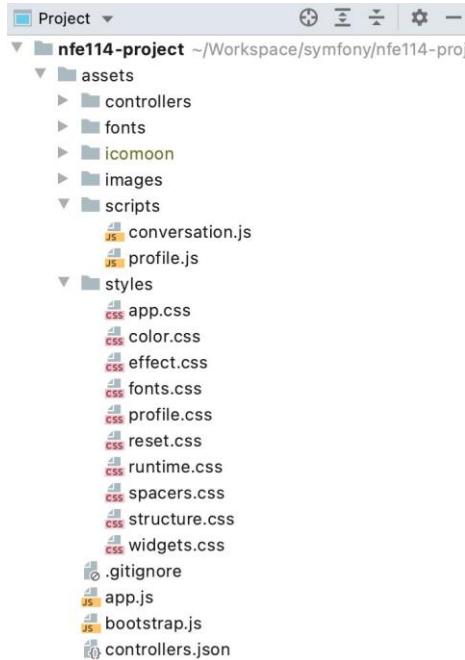


Illustration 3 - Fichiers d'assets

La configuration de **Webpack** s'effectue dans le fichier `webpack.config.js` à la racine du projet. Comme on laisse **Encore** s'occuper de la configuration, c'est finalement lui qui va gérer l'ajout des plugins et l'ajout de nouvelles entrées.

Il est impératif qu'après toute modification du fichier de configuration `webpack.config.js`, les assets soient recomplilés.

```
luc@MacBook-Pro-de-Nouailhaguet nfe114-project % yarn encore dev
yarn run v1.22.10
$ /Users/luc/Workspace/symfony/nfe114-project/node_modules/.bin/encore dev
Running webpack ...
[DONE] Compiled successfully in 802ms 5:51:13 PM

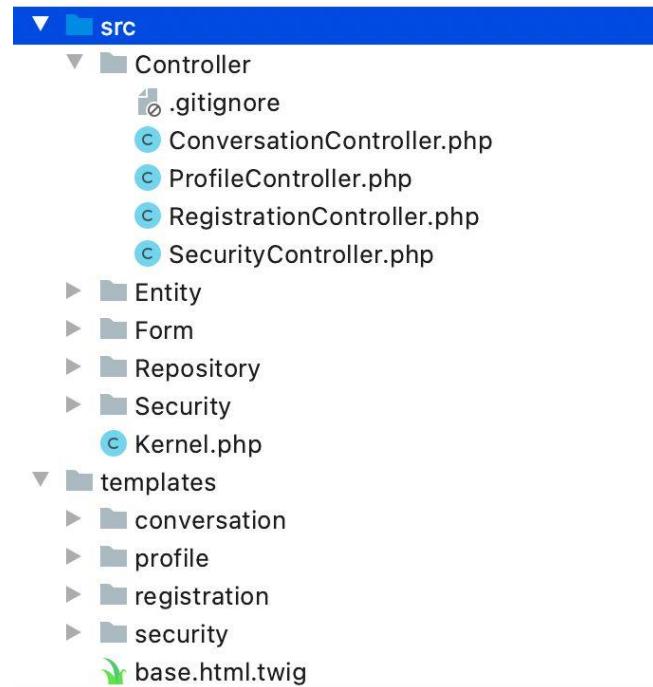
22 files written to public/build
Entrypoint app [big] 568 KiB (193 KiB) = runtime.js 14.4 KiB vendors-node_modules_core-js_internals_add-to-uncopables_js-node_modules_core-js_internals_a-f7cf62.js 131 KiB
  vendors-node_modules_symfony_stimulus-bridge_dist_index_js-node_modules_core-js_modules_es_ob-89d390.js 379 KiB app.css 19.4 KiB app.js 23.5 KiB 8 auxiliary assets
Entrypoint profile [big] 1.68 MiB = 6 assets
Entrypoint conversation [big] 997 KiB = runtime.js 14.4 KiB vendors-node_modules_jquery_dist_jquery_js.js 774 KiB vendors-node_modules_core-js_internals_add-to-uncopables_js-node_modules_core-js_internals_a-f7cf62.js 131 KiB vendors-node_modules_core-js_modules_es_array_filter_js-node_modules_core-js_modules_es_array-4ebf9a.js 66.9 KiB conversation.js 10.7 KiB
Entrypoint _tmp_copy 14.4 KiB (19.9 KiB) = runtime.js 1 auxiliary asset
webpack compiled successfully
✖ Done in 11.61s.
```

Illustration 4 - Compilation des assets

Pour toutes modifications internes aux fichiers déjà présents dans le répertoire assets, mettre **Encore** en mode watch est suffisant. [yarn encore dev –watch]

Structure du système de fichier :

La structure du système de fichier suit la structure de fichier classique Model View Controller :



Enregistrement :

Cas d'utilisation *S'enregistrer* du package UC-internaute.

POUR S'ENREGISTRER IL EST NECESSAIRE DE CONFIGURER LE DSN DU MAILSENDER DANS LE FICHIER D'ENVIRONNEMENT.

```
### !!!!!! CHANGE YOUR KEY !!!!!!!!
###> symfony/framework-bundle ###
APP_ENV=dev
APP_SECRET=98f604f00877b47ab018993f6d438022
###< symfony/framework-bundle ###

###> symfony/mailer ###
MAILER_DSN=PutYourSenderCredentialHere
###< symfony/mailer ###
```

Illustration 6 - Partie du fichier d'environnement .env où définir les informations d'identification du sender mail

On construit une entité **User**.

Cette entité est conceptuellement équivalente à la modélisation logique précédemment définie à une propriété près qui va nous permettre de cloisonner l'accès à l'application via le composant Symfony de sécurité.

Cette propriété est la propriété **rôle**.

Pour qu'un utilisateur accède à l'application, il est nécessaire que cet utilisateur soit pleinement authentifié.

▼  user	
 id	int(11) (auto increment)
 email	varchar(180)
 roles	longtext
 uuid	varchar(36)
 password	varchar(255)
 create_datetime	datetime
 update_datetime	datetime
 is_verified	tinyint(1)
 PRIMARY	(id)
 UNIQ_8D93D649D17F50A6	(uuid)
 UNIQ_8D93D649E7927C74	(email)
 UNIQ_8D93D649D17F50A6	(uuid) UNIQUE
 UNIQ_8D93D649E7927C74	(email) UNIQUE

Illustration 7 - Entité Conceptuelle User

On utilisera l'uuid comme identifiant unique pour les invitations.

La page d'enregistrement est une page accessible à tout internaute, dans laquelle il renseigne son adresse électronique, identifiant unique sur le site, ainsi que son mot de passe.

Le contrôleur instancie un formulaire qui va être injecté dans la page servie par le serveur web en utilisant le moteur de gabarit de page html : **Twig**.

```

$user = User::create();
$form = $this->createForm( type: RegistrationFormType::class, $user);
$form->handleRequest($request);

if ($form->isSubmitted() && $form->isValid()) {
    // encode the plain password
    $user->setPassword(
        $passwordEncoder->encodePassword(
            $user,
            $form->get('plainPassword')->getData()
        )
    );
}

$entityManager = $this->getDoctrine()->getManager();
$entityManager->persist($user);
$entityManager->flush();

// generate a signed url and email it to the user
$this->emailVerifier->sendEmailConfirmation( verifyEmailRouteName: 'app_verify_email', $user,
    (new TemplatedEmail())
        ->from(new Address( address: 'nfe114.mailsender@gmail.com', name: 'nfe114 mail bot'))
        ->to($user->getEmail())
        ->subject( subject: 'Please Confirm your Email')
        ->htmlTemplate( template: 'registration/confirmation_email.html.twig')
);
// do anything else you need here, like send an email

return $this->redirectToRoute( route: 'app_profile');
}

return $this->render( view: 'registration/register.html.twig', [
    'registrationForm' => $form->createView(),
]);
}

```

Illustration 8 - Contrôleur d'enregistrement

La page d'enregistrement servie par un formulaire d'envoyer un courriel contenant un lien de validation. Une fois le lien cliqué, l'utilisateur pourra se connecter au site via la page d'authentification.

localhost

Register

Email

Password

Agree terms

Register

Illustration 9 - page d'enregistrement

Je n'ai ni stylisé la page d'enregistrement de compte, ni stylisé le mail reçu.

Hi! Please confirm your email!

Please confirm your email address by clicking the following link:

[Confirm my Email](#). This link will expire in 1 hour.

Cheers!



Illustration 10 - Courriel d'enregistrement

Authentification :

Une fois enregistré, un utilisateur doit s'authentifié pour accéder au site. Il est donc nécessaire de définir un contrôleur d'authentification.

Ce contrôleur d'authentification sera couplé au composant sécurité du framework Symfony afin de respecter le besoin métier d'isolation de l'application du reste de l'internet.

```
# Easy way to control access for large sections of your site
# Note: Only the *first* access control that matches will be used
access_control:
    - { path: ^/login|register, roles: PUBLIC_ACCESS }
    - { path: ^/, roles: [ IS_AUTHENTICATED_FULLY ] }
```

Illustration 11 - partie de la configuration du composant sécurité

Tout utilisateur essayant d'accéder à une page autre que login ou register doit être pleinement authentifié.

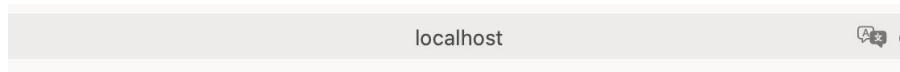


Illustration 12 - page d'authentification

Une fois authentifié, un utilisateur accède à sa page de profil, il devient donc naturel de conceptualiser le profil utilisateur.

Profil :

Le profil utilisateur contient toutes les infos de l'utilisateur comme son nom, son prénom ou sa photo de profile.

Je prends le parti de conceptualiser l'image de profil en tant que fichier comme un identifiant. En l'occurrence le nom du fichier rendu unique.

On construit l'entité profile qu'on lie à un utilisateur :

▼ profile	
	id int(11) (auto increment)
	user_id int(11)
	username varchar(255)
	notification_email tinyint(1)
	notification_desktop tinyint(1)
	picture varchar(512)
	first_name varchar(255)
	last_name varchar(255)
	PRIMARY (id)
	UNIQ_8157AA0FA76ED395 (user_id)
	FK_8157AA0FA76ED395 (user_id) → user (id)
	UNIQ_8157AA0FA76ED395 (user_id) UNIQUE

Illustration 13 - Entité profil

On notifie l'ORM de la relation à double sens, ceci afin de permettre de récupérer le profil depuis l'utilisateur au moment de la levée de bouclier par l'authenticator Symfony.

Lors de l'accès aux pages protégées, l'authenticator vérifie l'utilisateur. Cet utilisateur est accessible par le container de service injecté dans le contrôleur servant la requête de la route protégée.

```
/**  
 * @ORM\OneToOne(targetEntity=Profile::class, mappedBy="user", cascade={"persist", "remove"})  
 * @Assert\Type(type="App\Entity\Profile")  
 */  
private $profile;
```

Illustration 14 - Annotation de l'entité User dans sa relation entretenue avec l'entité Profile

Au moment de la première authentification d'un utilisateur, ce dernier ne possède pas de profil, alors il faut lui en créer un et l'associer à l'entité utilisateur fraîchement authentifié.

```

* @Route("/profile", name="app_profile")
*/
public function index(EntityManagerInterface $em,
    ProfileRepository $profileRepository,
    UserRepository $userRepository,
    Request $request): Response
{
    $user = $userRepository->findOneBy(['email' => $this->getUser()->getUserIdentifier()]);
    $profile = $profileRepository->getProfileByUser($user);
    $profile = $profile?? (new Profile())->setUser($user);
    $form = $this->createForm( type: ProfileFormType::class, $profile);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()){
        $oldProfile = $profile;
        /** @var Profile $profile */
        $profile = $form->getData();
        $profile->setNotificationDesktop((bool) $oldProfile->getNotificationDesktop());
        $profile->setNotificationEmail((bool) $oldProfile->getNotificationEmail());
        $em->persist($profile);
        $em->flush();
    }

    return $this->render( view: 'profile/profile_home.html.twig', [
        'controller_name' => 'ProfileController',
        'profile_form' => $form->createView(),
        'picture' => $profile->getPicture(),
        'uuid' => $user->getUuid(),
        'csrf_token_string' => 'tokenString'
    ]);
}

```

Illustration 15 – Contrôleur de la page profile

Puis afficher le profil qui se décompose en un formulaire composé dont les champs sont éclatés :

- Un formulaire dont la validation opère sur l'entité User
- Un formulaire dont la validation opère sur l'entité Profile

Profile


in

Nom :
Nouailhaguet

Prénom :
Luc

Email :
luc.nouailhaguet@gmail.com

Nom d'utilisateur:
Nom d'utilisateur

Mot de passe :

Identifiant
 D769FB8D-1174-47AA-80C0-9AF481EC2E5B

METTRE À JOUR

[Conversations](#)

Illustration 16 - Vue de la page Profil

Nom, prénom, username et l'image de profile seront enregistré dans l'entité Profile.

Email et mot de passe seront enregistrés dans l'entité User.

ATTENTION : LA MODIFICATION DU MOT DE PASSE NE FONCTIONNE PAS ENCORE.

La modification de l'image s'effectue en ajax : en cliquant sur le crayon on affiche le formulaire d'envoi de fichier qui est géré par une route du contrôleur de profil.

Nom :	Nouailh
Prénom :	Luc
Email :	luc.nou
Nom d'utilisateur	

Illustration 17 - Formulaire d'envoi de l'image Profil

A partir du profil, on peut ajouter les ancrés vers les pages conversation, stockage et carnet de contacts.

Je me suis occupé d'ajouter l'ancre de la page conversation mais pas les autres.

Conversation :

Package conversation

On construit l'entité Conversation et on conceptualise la liaison entre une conversation et son propriétaire par une liaison de multiplicité 1-n avec l'entité Profile : une entité Profile peut posséder plusieurs entité Conversations.

De plus, on conceptualise la relation entre l'existence d'une archive et la conversation archivée à travers la présence d'un drapeau au sein de l'entité Conversation. La construction de l'objet archive étant pleinement dépendante de l'identité de la conversation, on peut définir l'objet archive comme un **value object** pris dans le sens défini par Eric Evans dans son ouvrage « La conception pilotée par le domaine ».

▼ conversation	
	id int(11) (auto increment)
	proprietaire_id int(11)
	titre varchar(255)
	create_datetime varchar(255)
	archived tinyint(1) = 0
	PRIMARY (id)
	FK_8A8E26E976C50E4A (proprietaire_id) → profile (id)
	IDX_8A8E26E976C50E4A (proprietaire_id)

Illustration 18 - Entité Conversation

La page de conversation est une composition de plusieurs éléments de présentation. Toutes les interactions sur cette page se feront par l'intermédiaire d'ajax. Tous les événements liés à la messagerie seront effectués via mercurie.

L'accès à la page de conversation transforme automatiquement l'acteur Utilisateur en acteur Participant. Ceci bien qu'il ne participe à aucune conversation pour le moment.

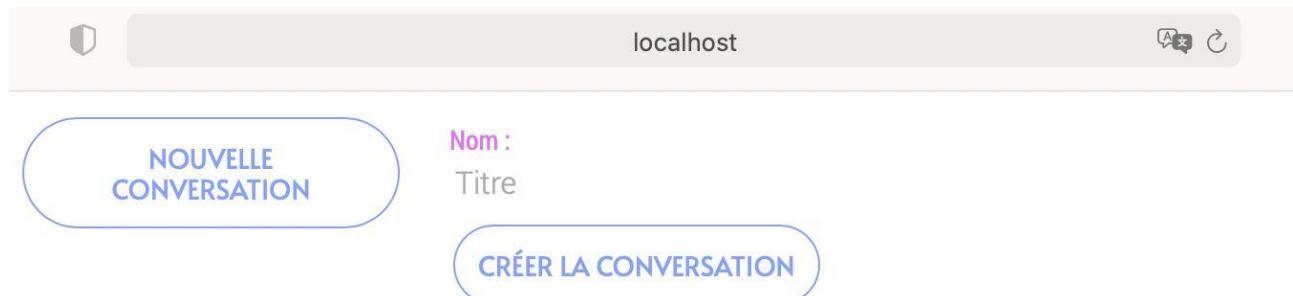


Illustration 19 - Créer une nouvelle conversation

En cliquant sur le bouton « Nouvelle conversation », le contrôleur de conversation construit le formulaire de nouvelle conversation et le renvoie sous forme sérialisée pour injection dans le navigateur client.

```
* @Route("/nouvelle_conversation", name="nouvelle_conversation")
* @param Request $request
* @return Response
*/
public function newConversation(Request $request): Response
{
    // get the new conversation form and restitute as an ajax response.
    $conversation = new Conversation();
    $nouvelleConversationForm = $this->createForm( type: NouvelleConversationFormType::class, $conversation);
    $renderedTemplate = $this->render( view: 'conversation/nouvelle_conversation.html.twig', [
        'nouvelle_conversation_form' => $nouvelleConversationForm->createView()
    ]);

    return (new JsonResponse([
        'html' => $renderedTemplate->getContent()
    ], status: 200));
}
```

Illustration 20 - Contrôleur nouvelle conversation

Une fois la nouvelle conversation enregistrée, elle devient disponible au propriétaire en tant que participant et il peut y inviter d'autres profils et y poster des messages.

Bien que je n'aie pas eu le temps d'aller plus loin, j'aimerais faire une remarque concernant la conceptualisation de l'objet invitation.

Du point de vue du contenu, une invitation de modélise comme un objet accessible. Mais du point de vue de la conception c'est une classe d'association liant deux participants et une conversation.

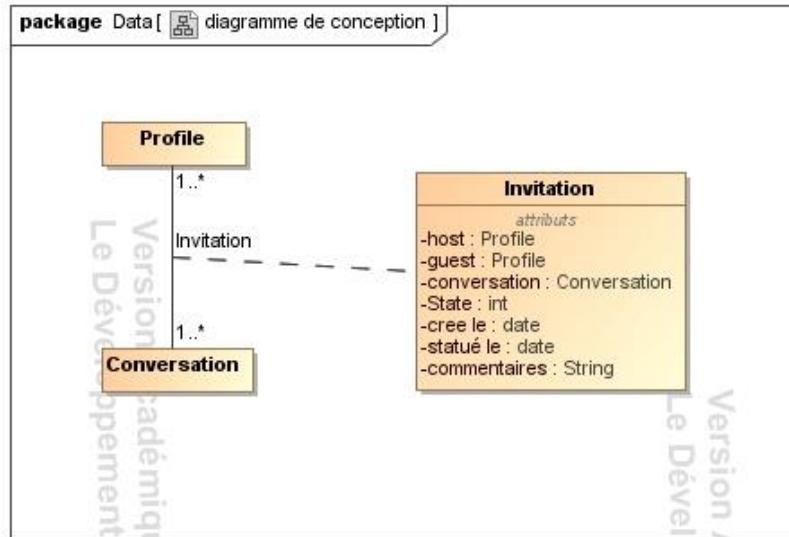


Illustration 21 - Diagramme de conception de l'objet Invitation

Références :

Bibliographie :

- UML 2.5 PAR LA PRATIQUE : ETUDES DE CAS ET EXERCICES CORRIGÉS.
PASCAL ROQUES
- APPLYING UML AND PATTERNS : AN INTRODUCTION TO OBJECT-ORIENTED ANALYSIS AND DESIGN AND ITERATIVE DEVELOPMENT
CRAIG LARMAN
- LES CAHIERS DU PROGRAMMEUR, UML2 : MODÉLISER UNE APPLICATION WEB
PASCAL ROQUES
- DOMAIN DRIVEN DESIGN
ERIC EVANS

Sites internet :

- <https://www.omg.org/>
- <https://symfony.com/>
- <https://martinfowler.com/>
- <https://nodejs.org/>
- <https://getcomposer.org/>
- <https://packagist.org/>
- <https://babeljs.io/>
- <https://webpack.js.org/>
- <https://docs.docker.com/>
- <https://www.w3.org/TR/webrtc/>
- <https://les-tilleuls.coop/fr/blog/article/mercure-un-protocole-pour-pousser-des-mises-a-jour-vers-des-navigateurs-et-app-mobiles-en-temps-reel>

Site du projet :

<https://github.com/012kirby210/nfe114-project>