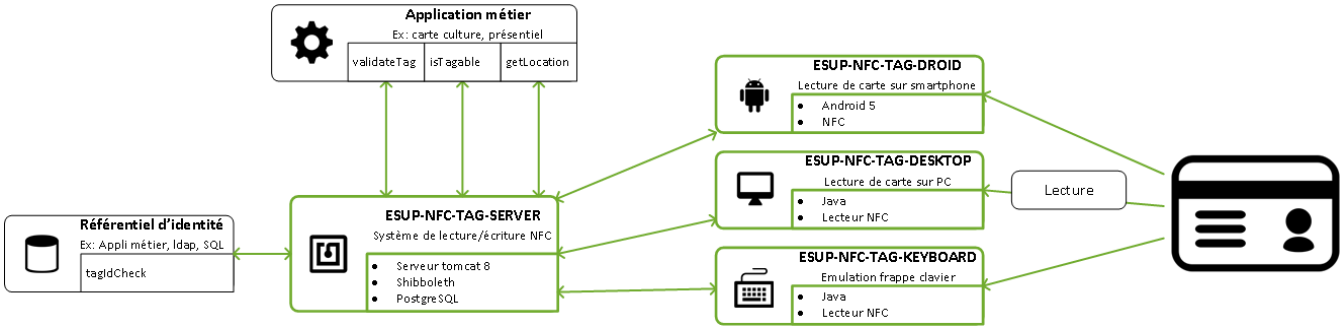
ESUP-NFC-TAG-SERVER

Introduction

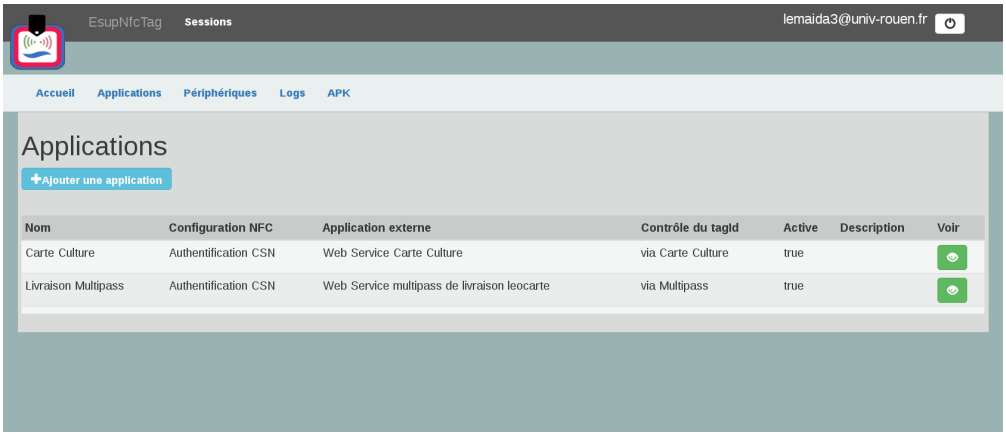
Ce projet vise à permettre et faciliter le développement de services autour des cartes NFC dites "multiservice". Il propose une architecture standardisée et connectée autour du badgeage d'une carte présentant un identifiant (CSN ou identifiant codé en Desfire) correspondant à une carte valide d'un individu connu du système d'information.

Esup-nfc-tag-server permet d'utiliser comme lecteur/borne de badge NFC :

- un ordinateur + lecteur usb NFC : ESUP-NFC-TAG-DESKTOP, ESUP-NFC-TAG-KEYBOARD
- un smartphone Android avec NFC : ESUP-NFC-TAG-DROID
- ou encore éventuellement un Arduino : ESUP-NFC-TAG-ARDUINO



L'application Esup-nfc-tag-server est développée en Spring (ROO) et tourne sur Tomcat.

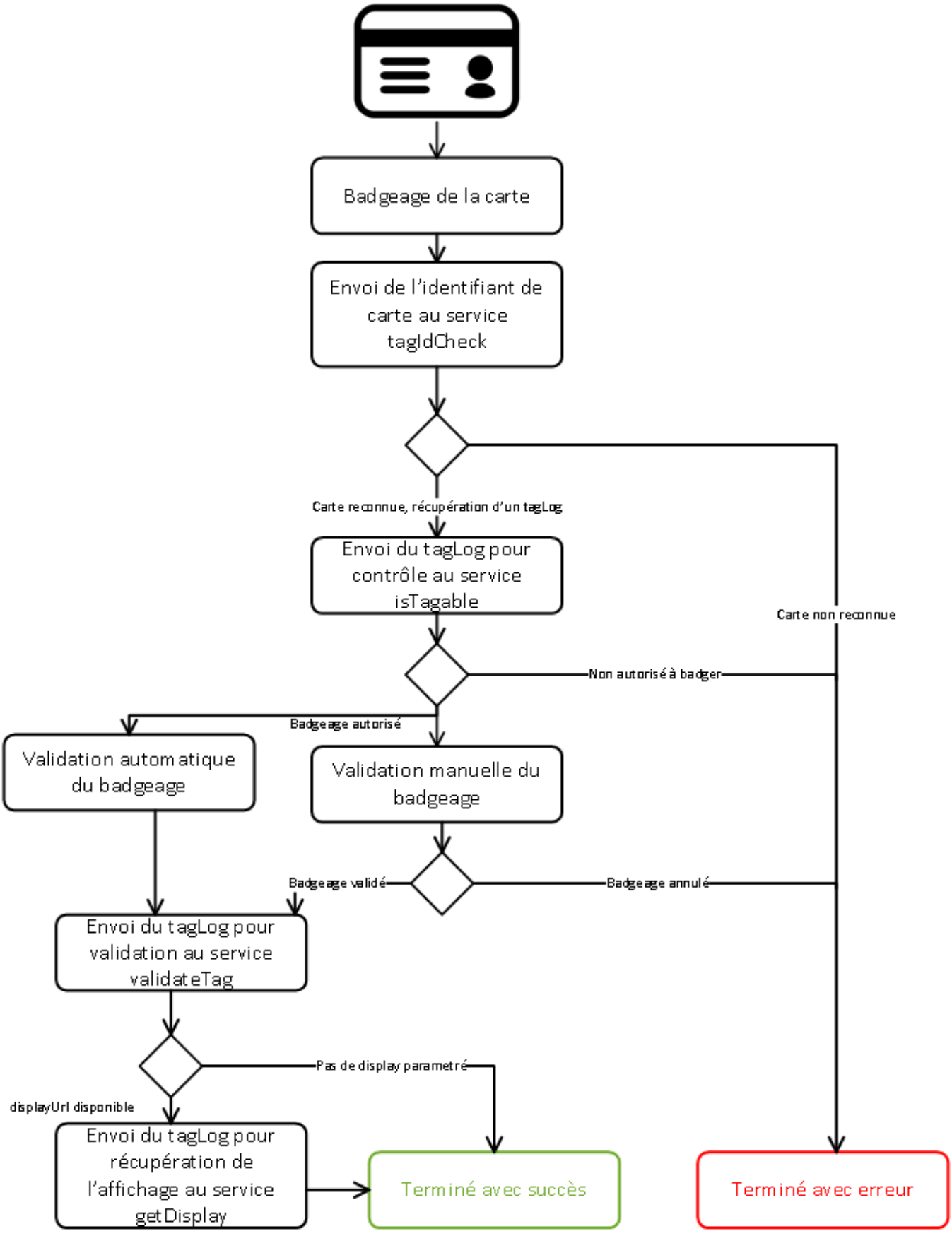


- Introduction
- Badgeage (tag_log)
- Applications
- Les salles (locations)
- Périphériques (devices)

Badgeage (tag_log)

Esup-nfc-tag-server à été pensé pour être souple et s'intégrer dans différents systèmes d'information. C'est pourquoi l'action de badger a été séparée en plusieurs étapes :

1. Lecture de l'identifiant de carte
2. Rechercher de l'identifiant de la carte dans un des référentiel du SI (tagIdCheck), puis création d'un tag_log au statut "none"
3. Interrogation d'une application métier pour vérifier l'autorisation pour cette carte (isTagable)
4. Validation du badgeage et lancement d'une procédure métier (validateTag), puis modification du statut du tag_log
5. En option récupération d'éléments à afficher (getDisplay)


Badgeage de la carte
Envoi de l'identifiant de carte au service tagIdCheck
Carte reconnue, récupération d'un tagLog
Envoi du tagLog pour contrôle au service isTagable
Carte non reconnue
Non autorisé à badger
Badgeage autorisé
Validation automatique du badgeage
Validation manuelle du badgeage
Badgeage validé
Badgeage annulé
Envoi du tagLog pour validation au service validateTag
Pas de display parametré
displayUrl disponible
Envoi du tagLog pour récupération de l'affichage au service getDisplay
Terminé avec succès
Terminé avec erreur

Lors du badgeage un objet tag_log est créé au moment du tagIdCheck (si l'identifiant de carte est retrouvé dans le référentiel). Le tag_log représente la date (authDate), le lieu (location), et l'identité (eppn) du badgeage.

```
public class TagLog {
    private String desfireId;
    private String csn;
    private String eppn;
    private String firstname;
    private String lastname;
    private String numeroId;
    private String eppnInit;
    private String applicationName;
    private String location;
    private Status status;
    private Date authDate;

    public enum Status {
        none, valid, cancel
    }
}
```

A sa création le tag_log porte le statut "none", il passera à "valid" si le procéssus va jusqu'au bout et si tout se passe bien. Si le badgeage est annulé par l'utilisateur le status passe à "cancel". Dans les autres cas il restera au statut "none".

Dans esup-nfc-tag **eppnInit** représente toujours le gestionnaire (celui qui badge). **eppn** représente l'identifiant du propriétaire de la carte badgée.

Applications

Esup-nfc-tag-server propose de gérer des "Applications" qui représentent la conjonction de 3 éléments:

1. Une configuration d'authentification sur la carte NfcAuthConfig (pour faire simple CSN ou DESFIRE)
2. Un service de recherche de l'indentifiant TagIdCheckApi
3. Les urls des web-services métiers AppliExtRestApi comportant: isTagableUrl, validateTagUrl, getLocationsUrl et eventuellement displayUrl

Exemple d'une application de controle de présence :

1. *Configuration NFC : Desfire*
2. *Contrôle du tagId : via LDAP*
3. *Application externe : web service controle de présence*

Autre exemple d'une application de carte culture :

1. *Configuration NFC : CSN*
2. *Contrôle du tagId : via carte culture*
3. *Application externe : web service carte culture*

D'origine Esup-nfc-tag-server est fourni avec les implémentations suivante :

- pour NfcAuthConfig : CsnAuthConfig (lecture simple du CSN), DesfireReadConfig (Lecture d'une application Desfire, voir ici), DesfireWriteConfig et DesfireUpdateConfig (configurations spécifiques pour ESUP-SGC)
- pour TagIdCheckApi : TagIdCheckLdap (recherche LDAP), TagIdCheckSql (requete SQL dans une base métier), TagIdCheckRestWs (recherche via un web service)
- pour AppliExtRestApi : AppliExtRestWs (communication avec les applications métier en web services REST)

A voir :

Implémentation du Web Service TagIdCheck

Implementation du webService AppliExtRestWs

En option il est possible de préciser si, pour une application donnée, une validation par le gestionnaire (validation manuelle) est nécessaire. Il est aussi possible de déclarer une application active ou inactive, ceci ayant pour conséquence de rendre l'application ainsi que les salles (locations) non visible (cependant elle reste fonctionnelle).

Les salles (locations)

La notion de salles permet de proposer des applications "multi-salles". Dans le cas d'un dispositif carte culture, par exemple, les salles sont les différentes salles de spectacle. Pour une application de contrôle de présence il pourrait s'agir de salles de classe.

La liste des salles est donc fournie par l'application métier sous forme d'une liste de String au format JSON, accessible par esup-nfc-tag-server, en échange de l'eppn du gestionnaire (eppnInit). L'application métier gère de son coté les salles disponibles en fonction des droits du gestionnaire qui utilise l'application.

```
["Salle 1", "Salle 2"]
```

Périphériques (devices)

Chaque périphérique (Android, Appli Java, Arduino, Emulation clavier, encodeur ...) est enregistré dans esup-nfc-tag-server. Dans la plupart des cas les périphériques sont créés automatiquement lors du démarrage des applications clientes.

Il est nécessaire de créer manuellement des applcations dans le cas d'esup-nfc-tag-keyboard et pour les arduinos.

```
public class Device {
    private String numeroId;
    private boolean validateAuthWoConfirmation;
    private String eppnInit;
    private String imei;
    private String macAddress;
    private String userAgent;
    private String location;
    private Application application;
}
```

Le numéro d'identification du périphérique (numéroId) est utilisé lors du badgeage pour retrouver le lieu du badgeage (location), l'application et le gestionnaire (eppnInit). On recupère aussi le paramètre de validation automatique (validateAuthWoConfirmation).

Installation ESUP-NFC-TAG-SERVER