

ESUP-DAYS **30**



En action à l'Université de Rouen Normandie

HISTORIQUE DU PROJET

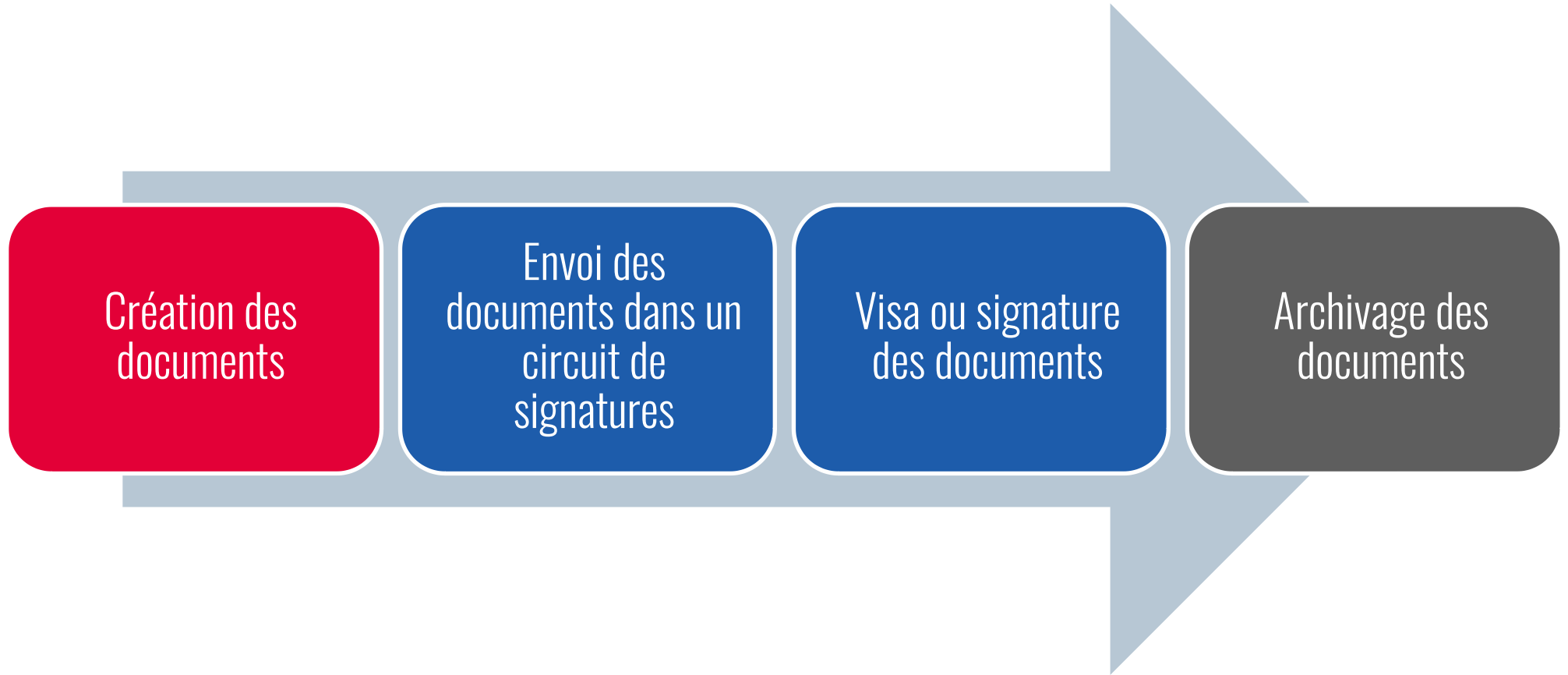
◆ Besoins initiaux

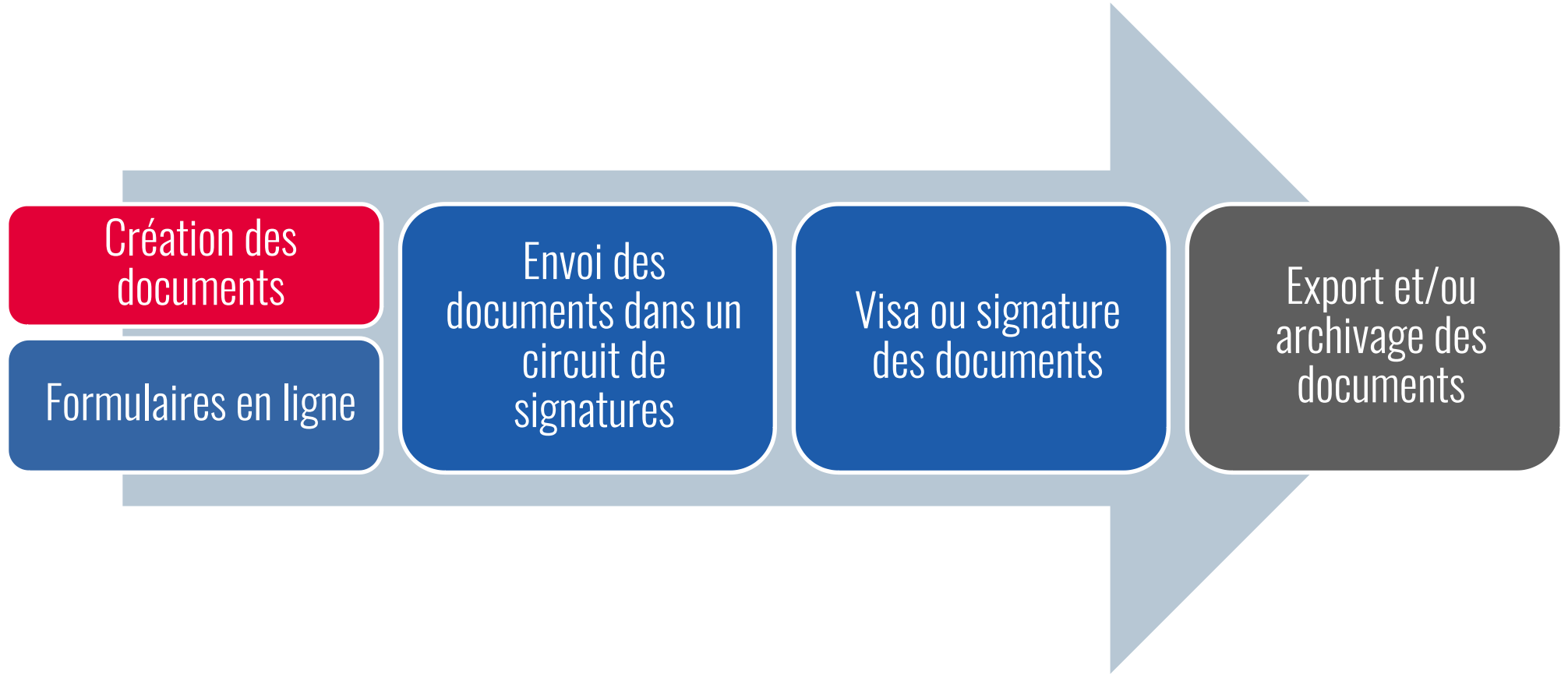
- Simplifier, normaliser et fluidifier les circuits de validation de documents
- Signer des documents avec ou sans certificat
- Tendre vers le **zéro papier**

◆ Esup-signature

- Un outil de signature et de gestion des circuits de signature
- Un outil de dématérialisation des documents et des parapheurs
- Un outil open-source







COUVERTURE FONCTIONNELLE

- ◆ Signature de documents (visas, tampons image, signatures électroniques PAdES, CAdES, XAdES)
- ◆ Création de circuits de signatures (parapheurs) paramétrables
- ◆ Import/Export automatique de documents (partage réseau SMB, GED CMIS)
- ◆ Reconnaissance automatique des champs et des méta-données des PDF
- ◆ Gestion d'alertes et de commentaires
- ◆ Mise à disposition d'APIs pour brancher des applications métiers
- ◆ Gestion des délégations
- ◆ Mise en ligne de formulaires PDF (PDF Forms)

◆ Délégations

- Signature (valable seulement pour le signature calligraphique)
- Création de documents ou demande de signature

◆ Ajout de pièces jointes dans les demandes

◆ Double export possible

- Vers répertoire de travail
- Vers une zone d'archivage

◆ One Time Password pour la signature des personnes externes

◆ Outils de mise en ligne de formulaires PDF

- Création d'un formulaire PDF à l'aide d'un outil standard (Acrobat Pro, Master PDF Editor...)
- Mise en ligne dans esup-signature
- Associé à un circuit de validation/signature
- Extraction des données saisies

◆ Fonctionnement

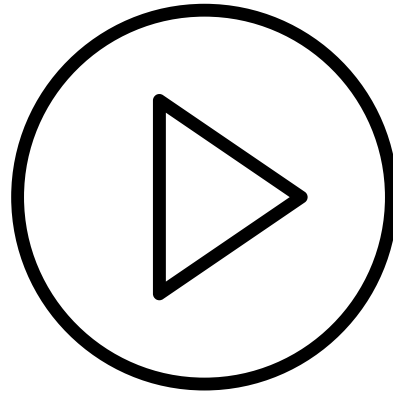
- S'appuie sur PDF.js et PDFBOX
- Un système de nommage des champs permet un pré-remplissage avec des données du SI

RETOUR D'EXPÉRIENCE

- ◆ Signature à la demande
 - Signer soi-même des documents
 - Demander des signatures
 - Signature calligraphique suffisante

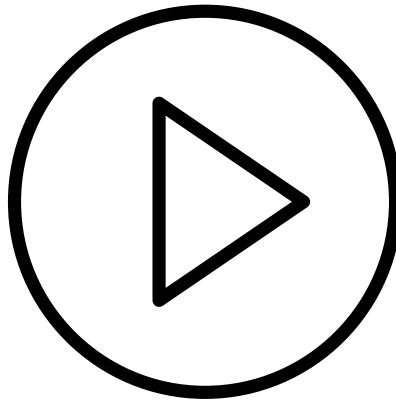
- ◆ En chiffres :
 - 1036 demandes signées depuis avril 2020

- ◆ Démo vidéo d'une demande simple

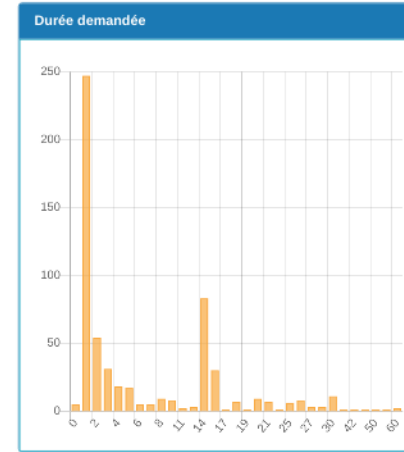
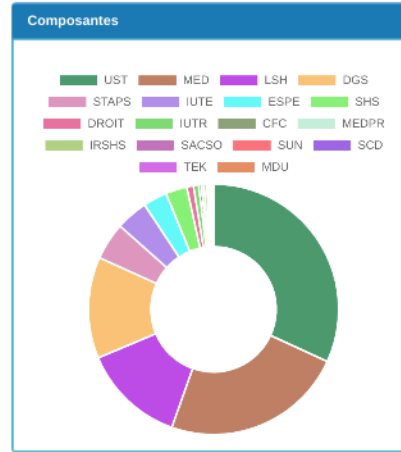
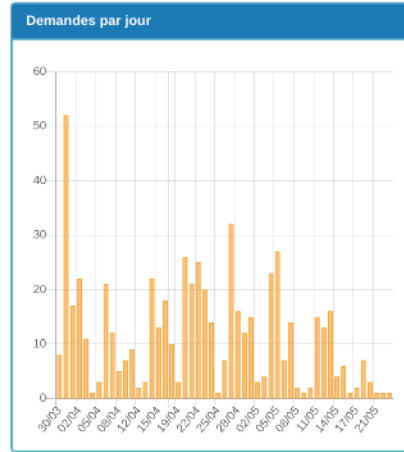
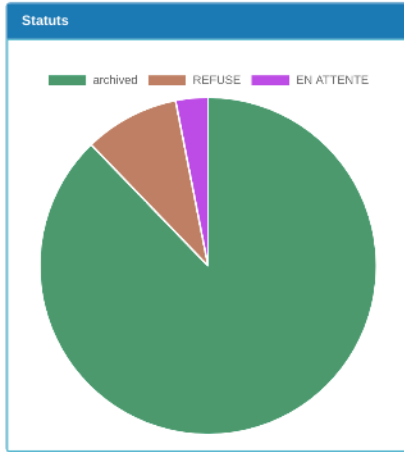


- ◆ Formulaire de demande de déplacement
 - A partir du PDF ministériel
 - Ajout des champs et création d'un circuit de signatures (validation n+1 et signature du Président)
 - Mise en place d'un tableau de bord
 - En chiffres : 581 demandes de déplacement traitées

- ◆ Démo vidéo de la saisie d'un PDF en ligne (attestation de déplacement COVID)



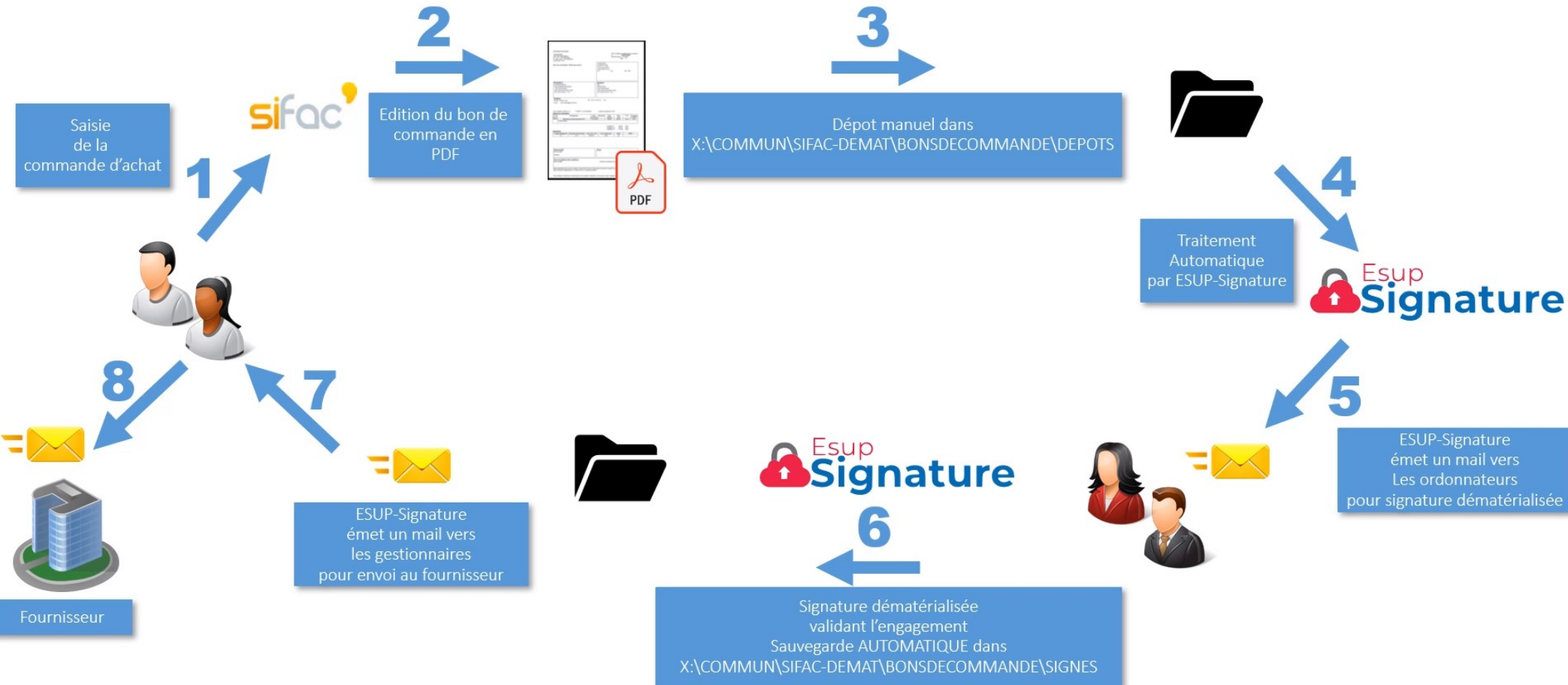
◆ Extractions et statistiques à partir des données des formulaires



◆ Signature des bons de commandes

- Issus de SIFAC
- Un script maison ajoute des méta-données PDF au BdC (créateur, unité budgétaire, ordonnateur et chemin de destination)
- Esup-signature intègre le BdC et configure le circuit automatiquement
- Périmètre de test DSI

RETOUR D'EXPÉRIENCE



- ◆ Une page wiki reprend les demandes d'évolutions des établissements

<https://www.esup-portail.org/wiki/pages/viewpage.action?pageId=906821633>

- Amélioration des alertes (relances)
- Traitement par lots
- Archivage SAE

- ◆ Code disponible sur GitHub

<https://github.com/EsupPortail/esup-signature>

- ◆ Jeu de test d'intégration

- ◆ Documentation wiki Esup

<https://www.esup-portail.org/wiki/display/SIGN>

- ◆ Site de démonstration

<https://esup-signature-demo.univ-rouen.fr>

```
public Document reqSign(SignRequest signRequest, User user) {
    Logger.info(user.getUsername() + " launch nexu signature for signRequest : " + signRequest);
    DSSDocument dssDocument;

    if (signatureDocumentForm.getClass().equals(SignatureMultipleDocumentsForm.class)) {
        dssDocument = signService.signDocument((SignatureMultipleDocumentsForm) signatureDocumentForm, parameters);
    } else {
        dssDocument = signService.nexuSignDocument((SignatureDocumentForm) signatureDocumentForm, parameters);
    }

    InMemoryDocument signedDocument = new InMemoryDocument(DSSUtils.toByteArray(dssDocument), dssDocument.getName(), dssDocument.getMimeType());

    return addSignedFile(signRequest, signedDocument.openStream(), dssDocument.getName(), signedDocument.getMimeType());
}

public Document certSign(SignRequest signRequest, User user, String password, boolean addDate, boolean visual) throws EsupSignatureException {
    SignatureForm signatureForm;
    List<Document> toSignFiles = new ArrayList<>();
    for (Document document : getToSignDocuments(signRequest)) {
        toSignFiles.add(document);
    }
    step = "Initialisation de la procédure";
    try {
        step = "Déverrouillage du keystore";

        SignatureTokenConnection signatureTokenConnection = userKeystoreService.getSignatureTokenConnection(user.getKeystore().getInputStream(), password);
        CertificateToken certificateToken = userKeystoreService.getCertificateToken(user.getKeystore().getInputStream(), password);
        CertificateToken[] certificateTokenChain = userKeystoreService.getCertificateTokenChain(user.getKeystore().getInputStream(), password);

        step = "Formatage des documents";

        AbstractSignatureForm signatureDocumentForm = signService.getSignatureDocumentForm(toSignFiles, signRequest, visual);
        signatureForm = signatureDocumentForm.getSignatureForm();
        signatureDocumentForm.setEncryptionAlgorithm(EncryptionAlgorithm.RSA);

        step = "Préparation de la signature";

        signatureDocumentForm.setBase64Certificate(Base64.encodeBase64String(certificateToken.getEncoded()));
        List<String> base64CertificateChain = new ArrayList<>();
        for (CertificateToken token : certificateTokenChain) {
            base64CertificateChain.add(Base64.encodeBase64String(token.getEncoded()));
        }
        signatureDocumentForm.setBase64CertificateChain(base64CertificateChain);

        AbstractSignatureParameters parameters = null;
        if (signatureForm.equals(SignatureForm.CADES)) {
            AS1CMTXCAESSignatureParameters as1CMTXCAESSignatureParameters = new AS1CMTXCAESSignatureParameters();
            parameters = as1CMTXCAESSignatureParameters.asIC().setContainerType(AS1ContainerType.AS1C_E);
        } else if (signatureForm.equals(SignatureForm.XADES)) {
            AS1CMTXCAESSignatureParameters as1CMTXCAESSignatureParameters = new AS1CMTXCAESSignatureParameters();
            parameters = as1CMTXCAESSignatureParameters.asIC().setContainerType(AS1ContainerType.AS1C_E);
        } else if (signatureForm.equals(SignatureForm.PADES)) {
            parameters = signService.fillVisibleParameters(parameters);
        }
    } catch (Exception e) {
        throw new EsupSignatureException(e);
    }
}
```


MERCI