

04.Feb.2020

#esupdays29
#apereoparis20

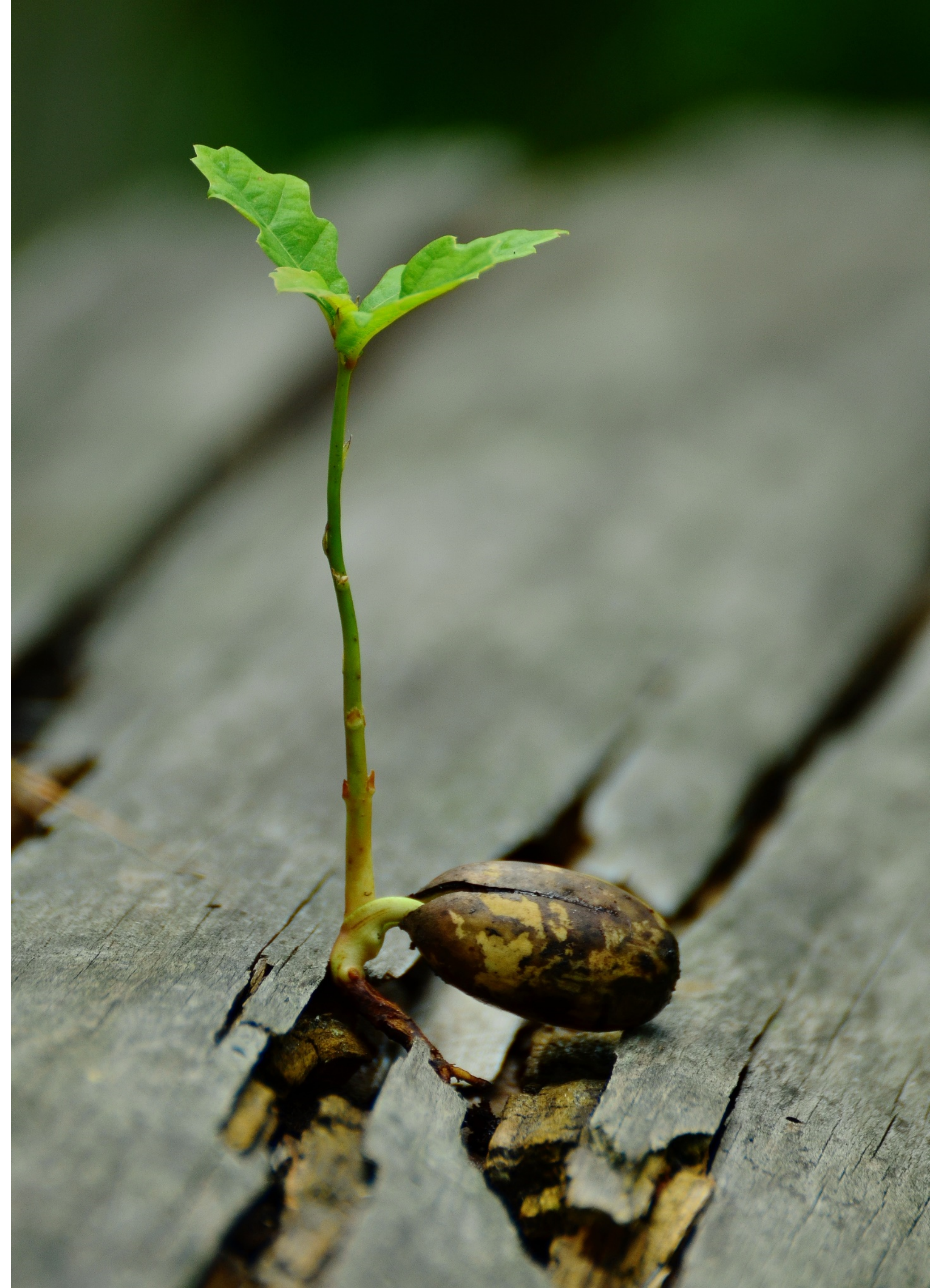


OUTIL DE SIGNATURE ÉLECTRONIQUE

David Lemaigent -- Université de Rouen Normandie

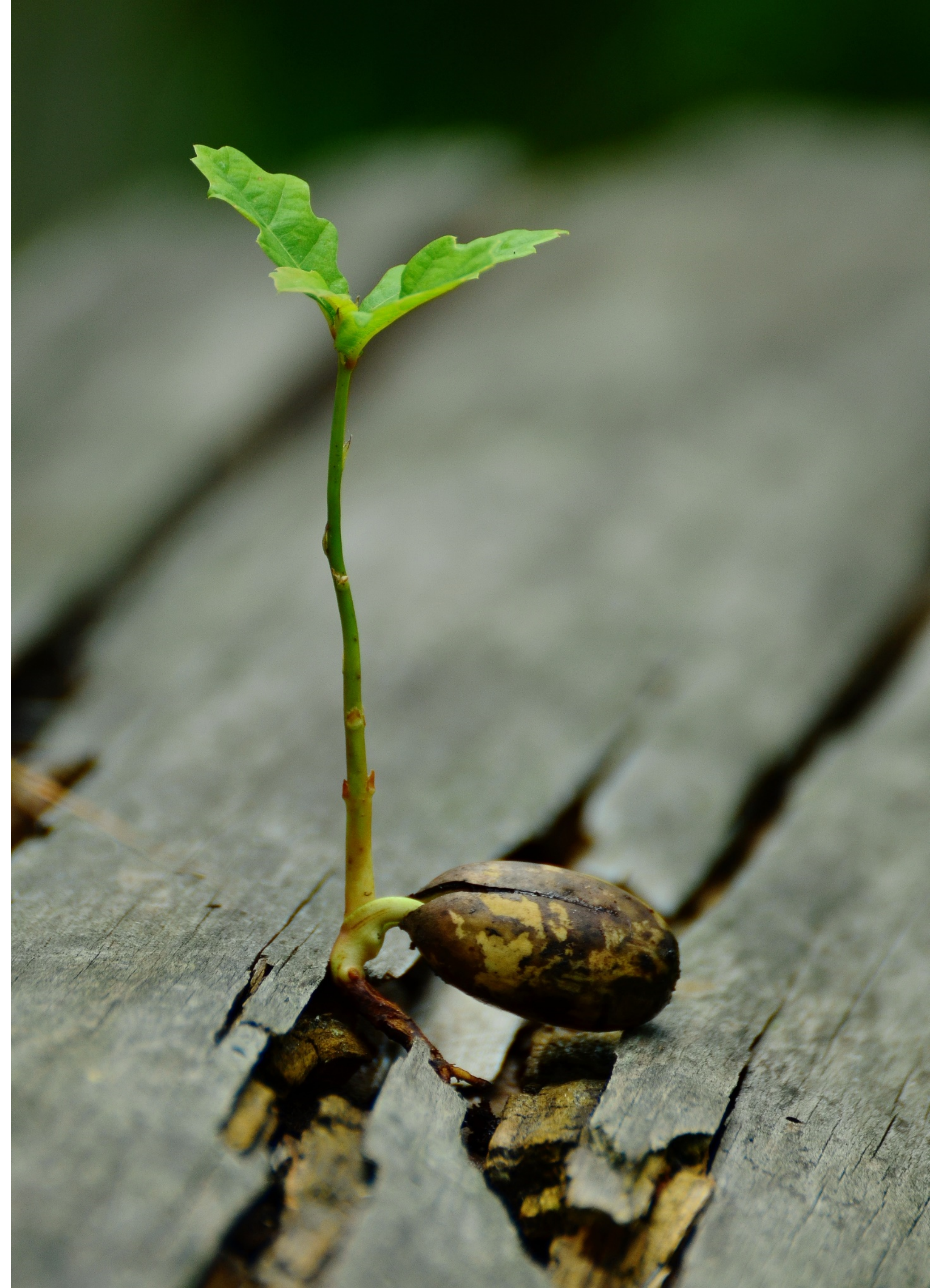
CONTEXTE DE L'UNIVERSITÉ DE ROUEN NORMANDIE

- ◆ Ni GED, ni système d'archivage mais des outils multiples
 - ◆ Stockage : homedirs, groupdirs, maildirs, drive
 - ◆ Intranet
 - ◆ Applicatifs métiers
- ◆ Procédures variables et indéfinies (informatiquement parlant)
 - ◆ Impression
 - ◆ Signature manuscrite
 - ◆ Numérisation
- ◆ Signatures électroniques (X509, RGS **)
 - ◆ Pour les marchés (via Adobe Reader)



OBJECTIFS DE L'UNIVERSITÉ DE ROUEN NORMANDIE

- ◆ Tendre vers le **zéro papier**
- ◆ Simplifier, normaliser et fluidifier les circuits de validation de documents
 - ◆ Vers le parapheur électronique
- ◆ Signer des documents avec ou sans certificat
 - ◆ En fonction de la criticité des documents
 - ◆ Pas de recours systématiques aux certificats électroniques



PROJET DÉMATÉRIALISATION À L'UNIVERSITÉ DE ROUEN NORMANDIE

- ◆ Un outil de dématérialisation des documents
- ◆ Un outil de signature électronique et de circuits de signature
- ◆ Un outil de gestion électronique de documents
- ◆ Un système d'archivage

PÉRIMÈTRE DU PROJET

- ◆ **Types de documents pressentis pour intégrer le projet**
 - ◆ Marchés (notification, rejets, actes d'engagement, avis d'attribution)
 - ◆ Ordres de mission
 - ◆ Virements et facturations internes
 - ◆ Entretiens professionnels
 - ◆ Demandes de téléphonie mobile
 - ◆ Demandes de Wi-Fi pour l'organisation des colloques
 - ◆ Autorisations de cumul
 - ◆ Conventions
 - ◆ Diplômes

Création des documents

Envoi des documents dans un circuit de signatures

Visa ou signature des documents

Archivage des documents

◆ Outils du marché

- ◆ Clé en main mais...
- ◆ Peu de maîtrise en interne des outils (nouvelle commande pour chaque nouveau circuit ?)
- ◆ Tarification par lots de signatures, au certificat ou à l'horodatage
- ◆ Pas toujours adapté à l'enseignement supérieur (faible intégration au SI)
- ◆ Certificat électronique requis (pas de visa CAS ou de signature calligraphique)
- ◆ Ou sont hébergés les documents, quid de la réversibilité ?

◆ Esup-Signature

◆ S'appuie sur les standards

- Respecte le règlement eIDAS grâce à DSS Signature
- Exports SEDA (Archives de France)
- PDF/A

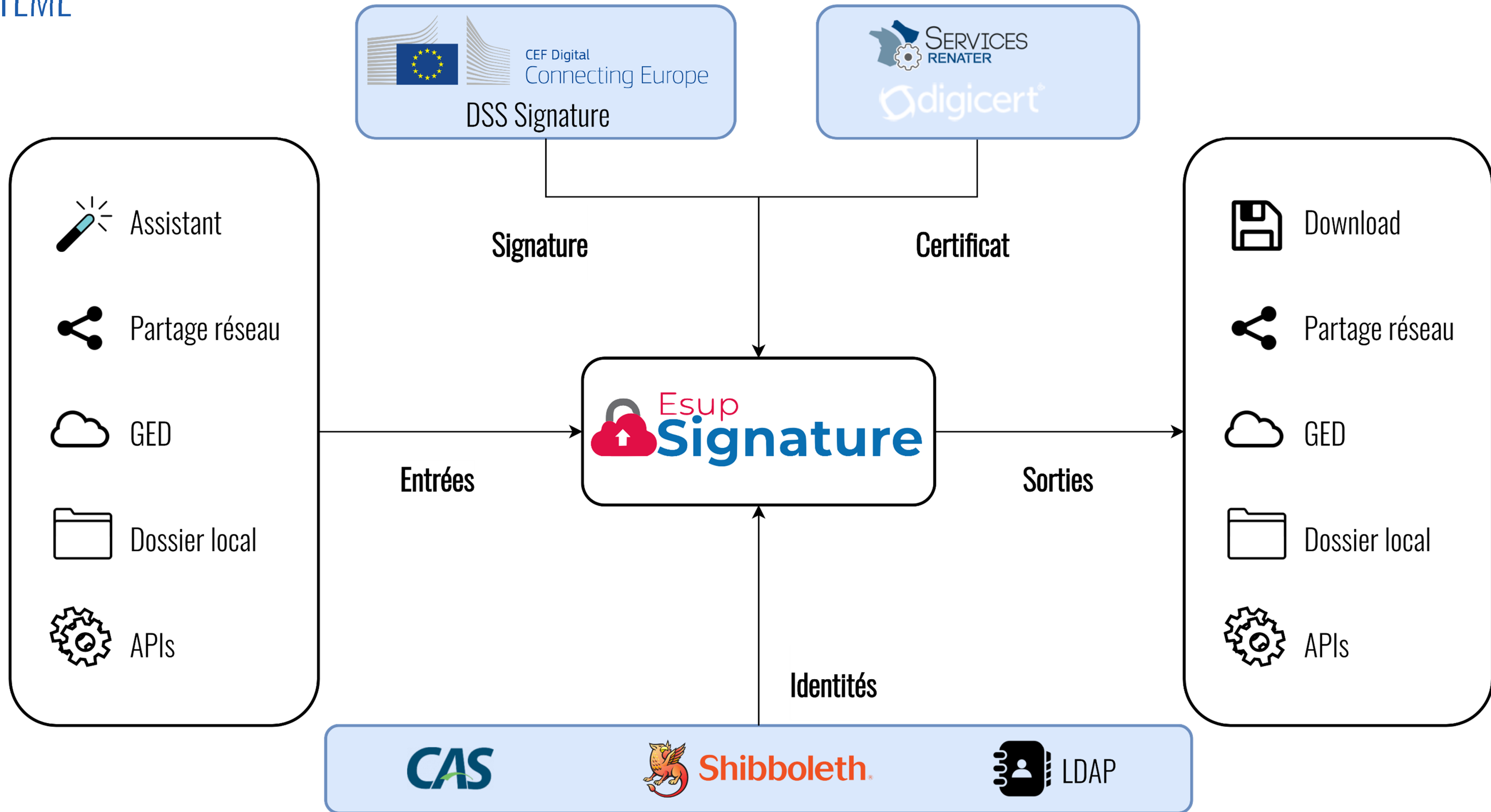
◆ Outil, libre et open source, évolutif, documenté et partagé

◆ Adapté à l'ESR

- Fédération d'identités,
- Authentification CAS/Shibboleth
- LDAP/SUPANN

◆ Utilisation de tous types de certificats (TCS RENATER ou autres)


◆ Hébergé localement



LE PROJET « DIGITAL SIGNATURE SERVICE »

- ◆ Mis à disposition par la Commission Européenne
- ◆ Open Source - <https://github.com/esig/dss>
- ◆ Conforme au règlement eIDAS
- ◆ Développé en Spring
- ◆ Documenté - <https://ec.europa.eu/cefdigital/DSS/webapp-demo/doc/dss-documentation.html>

CERTIFICATS TCS DE RENATER

- ◆ <https://www.digicert.com/sso>
- ◆ Connexion avec un compte d'établissement
- ◆  Non qualifiés par le RGS
- ◆ Procédure présentée lors de l'atelier GED de mars 2019
<http://esupportail.github.io/presentations/esup-signature-atelier-ged-2019/videos/esupdemcert.mp4>



COUVERTURE FONCTIONNELLE

- ◆ Signature de documents (visas, tampons image, signatures électroniques PAdES, CAdES, XAdES)
- ◆ Circuits de signatures paramétrables (ie. parapheurs)
- ◆ Import/Export automatique de documents
 - ◆ Partage réseaux (via SMB), GED (via CMIS)
- ◆ Reconnaissance automatique des champs de signature (PDF)
- ◆ Intégration de la librairie sedalib (Vitam) pour l'export SEDA
- ◆ Gestion d'alertes et de commentaires
- ◆ Mise à disposition d'APIs pour brancher des applications métiers

◆ 3 types de signature

- ◆ Validation ou visa (garantie par l'authentification CAS, Shibboleth et par la journalisation en base)
- ◆ Signature calligraphique
 - Apposition d'une image (exclusivement pour les documents PDF)
- ◆ Signature électronique avec horodatage
 - Certificat électronique (keystore lié au profil)
 - Support cryptographique

◆ 3 formats de signature électronique supportés

- ◆ PAdES : adapté aux documents PDF (signature visuelle possible)
- ◆ CAdES : les documents signés sont encapsulés dans un zip avec une signature PKCS#7
- ◆ XAdES : signature encapsulés au format XML

CIRCUITS DE VALIDATION

- ◆ Une demande de signature suit un circuit
- ◆ Une demande peut contenir plusieurs documents
 - Tous les documents doivent être signés/visés pour changer d'étape
- ◆ A chaque étape, un ou plusieurs participants en parallèle
 - Une seule signature
 - Toutes les signatures
- ◆ Possibilité de rédiger des commentaires
- ◆ Assistant de création / enregistrement de modèles

Provenance des documents

Solutions possibles

Documents bureautiques sur le poste de travail ou issu d'une numérisation



Création d'une demande de signature via l'assistant

Documents générés par une application "maison"



Utilisation des APIs

Documents stockés sur un partage réseau ou sur une GED (Nuxeo, Alfresco, SharePoint)



Une tâche planifiée d'Esup-Signature les intègre automatiquement dans un circuit prédéfini

RÔLES

◆ Gestionnaires

- ◆ Création de circuits de signature réutilisables
- ◆ Paramétrage des entrées/sorties automatiques des documents

◆ Utilisateurs

- ◆ Création des demandes de signature
- ◆ Signature ou visa des documents
- ◆ Gestion du profil utilisateur
 - Image de la signature
 - Keystore
 - Préférences d'alertes



PARIS

04.Feb.2020

#esupdays29

#apereoparis20

ESUP-SIGNATURE

DÉMONSTRATION

ESUP-SIGNATURE

MISE EN ŒUVRE

- ◆ Application Spring Boot
- ◆ 1 fichier de configuration
- ◆ Code disponible sur GitHub

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

- ◆ Documentation wiki Esup

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

- ◆ Site de démonstration

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

```
applyEndOfStepRules(signRequest)
}
}
public Document nexuSign(SignRequest signRequest, User user, AbstractSignatureForm signatureForm) {
    logger.info(user.getId() + " launch nexu signature for signRequest : " + signRequest.getId());
    DSSDocument dssDocument;
    if (signatureForm.getClass().equals(SignatureMultipleDocumentsForm.class)) {
        dssDocument = signService.signDocument((SignatureMultipleDocumentsForm) signatureDocumentForm);
    } 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().getMimeTypeString());
}
}
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());
        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)) {
            AbstractSignatureParameters aSICWithCAdESSignatureParameters = new ASICWithCAdESSignatureParameters();
            parameters = aSICWithCAdESSignatureParameters.aSIC().setContainerType(ASICContainerType.ASIC_E);
        } else if (signatureForm.equals(SignatureForm.XAdES)) {
            AbstractSignatureParameters aSICWithXAdESSignatureParameters = new ASICWithXAdESSignatureParameters();
            parameters = aSICWithXAdESSignatureParameters.aSIC().setContainerType(ASICContainerType.ASIC_E);
        } else if (signatureForm.equals(SignatureForm.PAdES)) {
            parameters = signService.fillVisibleParameters((SignatureDocumentForm) signatureDocumentForm);
        }
        if (signatureForm.equals(SignatureForm.PAdES)) {
            step = "Signature du document";
        }
    } catch (Exception e) {
        logger.error(e.getMessage());
    }
}
```


TODO LIST

- ◆ Signature des extérieurs
 - ◆ Boucle SMS
 - ◆ FranceConnect
 - ◆ Signature de certification
- ◆ Du multi-tenant
- ◆ Export multi destinations
- ◆ Délégations



MERCI