CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE REGIONAL DE TOULOUSE

MEMOIRE

présenté en vue d'obtenir le

DIPLOME d'INGENIEUR CNAM

EN INFORMATIQUE

Par Valériane PENA

Implémentation de fonctions de travail en mode collaboratif dans une application de workflow procédural en vue de faciliter les échanges entre les différents profils utilisateurs.

Soutenu le 1er Juillet 2015

JURY

PRESIDENT : Anne Wei

MEMBRES : Thierry Millan (Responsable de la filière informatique CNAM Toulouse)

Pascal Dayre (Tuteur au CNAM)

Wilhem Joseph-François (Tuteur et chef de projet au CIRSO)

Florence Saint-Martin (Analyste Développeuse au CIRSO)

REMERCIEMENTS

Je remercie tout d'abord Sébastien Caillard sans qui je n'aurais pas fait mon stage au CIRSO. Il m'a permis de mettre un terme à ma longue recherche de stage en proposant ma candidature.

Je remercie aussi Jacques Subra et Wilhem Joseph François pour m'avoir fait confiance et accueilli chaleureusement dans leur équipe au CIRSO.

Je remercie ensuite Mylène Beaulieu qui m'a apporté un énorme soutien. En plus de m'avoir ouvert les bras, elle m'a accordé beaucoup de son précieux temps pour m'expliquer et me présenter autant de choses qu'il a été nécessaire pour ma compréhension et ma connaissance des tenants et aboutissants du sujet de stage.

Je tiens à remercier Monique Mur, Eric Pagnucco et Florence Saint-Martin pour leur participation active dans l'évolution du projet. Ils ont été présents tout au long du projet, lors des réunions, d'entretien de présentation du projet…

Un grand merci à Benjamin Christophe qui a toujours été présent pour expliciter tout ce qui avait trait au domaine technique.

Je remercie aussi Fabrice Debonnet et Alexis Debergue pour leur gentillesse et leur implication lors de nos échanges autour de Sametime.

A tous les membres du bureau et du pôle application collaboratives qui m'ont permis de me sentir à l'aise et m'ont bien intégrée au sein de l'équipe.

Je tiens à remercier Thierry Millan pour m'avoir soutenue dans la recherche de mon stage et dans la définition de mon sujet de mémoire. Un grand merci pour sa bienveillance et sa disponibilité.

Je remercie aussi Valérie Crouzil qui m'a guidé dans mes démarches administratives.

Enfin je tiens à exprimer toute ma reconnaissance à mon tuteur Pascal Dayre qui m'a guidé à travers ce long parcours au cours duquel j'ai souvent eu des doutes. Il m'a apporté une structure et des encouragements qui m'ont beaucoup aidé à donner jour à ce mémoire.

Je n'oublie pas non plus tous mes amis qui ont peut être eu l'impression d'être mis à l'écart depuis le début de cette aventure, je leur adresse un grand merci pour être restés à mes côtés malgré tout.

Avant de terminer, j'adresse toute ma gratitude à ma famille : à mes parents, mon papa pour l'importance qu'il a su accorder à mon projet dès le début, et à mes beaux parents pour leur aide dans la dernière ligne droite.

Enfin j'adresse une spéciale dédicace à mon conjoint et mes enfants qui ont subi mon « absence » et mon humeur mais m'ont apporté du bonheur et du réconfort en toute circonstance.

LISTE DES ABREVIATIONS

ACOSS	Agence Centrale des Organismes de Sécurité Sociale),
ALTAIR	Atelier Logiciel Transverse Associé à l'Informatique du Recouvrement
BPM	Business Process Management (ingénierie des processus métiers)
BPMN	Business Process Model Notation
BPMS	Business Process Management System (outil de gestion BPM)
CAF	Caisse d'Allocations Familiales
CAIRN	Conception, Analyse, Implémentation pour la RéNovation
CERTI	Centre d'Expertise Régional en Technologie de l'Information
CGSI	Comité de Gestion de Système d'Information
CIRSO	Centre Informatique recouvrement Sud Ouest
COG	Convention d'Objectifs et de Gestion
CPAM	Caisse Primaire d'Assurance Maladie
CSCW	Computer Supported Collaborative Work (travail collaborative assisté par ordinateur)
CU	Cas d'utilisation
DCU	Document des cas d'utilisation
DNS	Données Nominatives Sociales
ERP	Enterprise Resource Planing (progiciel de gestion intégré)
IHM	Interface homme machine
LDAP	Lightweight Directory Access Protocol (protocole d'accès aux annuaires légers)
MOA	Maitrise d'OuvrAge
MOE	Maitrise d'Œuvre
PAC	Pôle Applications Collaboratives
PIA	Plan Informatique Annuel
REST	REpresentational State Transfer

ROA	Ressources Oriented Architecture (Architecture Orientée Ressources)
RIDA	Relevé d'Informations, de Décision et d'Actions
RSI	Régime Social des Indépendants
SI	Système d'Information
SSO	Single Sign On (authentification unique)
TCAO	Travail Collaboratif Assisté par Ordinateur
URI	Uniform Resource Identifier
URSSAF	Unions de Recouvrement des cotisations de Sécurité Sociale et d'Allocations Familiales
W3C	Worl Wide Web Consortium
WATT	Worflow d'Assistance aux Tâches Techniques
WFMC	Workflow Management Coalition

GLOSSAIRE

Affaire	dans Watt, une affaire peut être définie comme l'ensemble des informations correspondant à un traitement à effectuer par un agent de l'organisme pour répondre à un cas soulevé par un cotisant. Une affaire équivaut donc au dossier physique constitué puis traité par le gestionnaire de comptes.
Awareness	(voir Présentiel)
Circuit	dans Watt, un circuit permet de standardiser le traitement des affaires ayant des caractéristiques communes (type de document générateur, tâches, ordonnancement des tâches). Il représente tout le cheminement possible d'un processus en fonction des tâches et de leurs états.
Client léger	application de type hôte qui s'exécute via un navigateur par exemple et sollicite un serveur pour faire appel aux services nécessaires
Client lourd	logiciel installé sur une machine locale
Client riche	permet de développer des applications de types client lourd
Conteneur web	un moteur de servlets qui prend en charge et gère les servlets : chargement de la servlet, gestion de son cycle de vie, passage des requêtes et des réponses ... (http://www.jmdoudoux.fr/java/dej/chap-servlets.htm)
Document	dans Watt, on désigne par document tout type de support physique (courriers,…) ou dématérialisé (fax, e-mail, appel téléphonique) qui est alors traité lors de la vie d'une affaire. On distingue leur nature en parlant de documents générateurs d'affaire à l'origine de la création d'une affaire et de documents à rattacher qui alimentent l'affaire. Chaque document possède un type permettant de l'identifier dans les processus métiers .
Listener	un terme anglais utilisé de façon générale en informatique pour qualifier un élément logiciel qui est à l'écoute d'évènements afin d'effectuer des traitements (wikipédia)
Multicast	une forme de diffusion d'un émetteur (source unique) vers un groupe de récepteurs. Les termes « diffusion multipoint » ou « diffusion de groupe » sont également employés. Les récepteurs intéressés par les messages adressés à ce groupe doivent s'inscrire à ce groupe. (wikipédia)
Perspective (java)	Une perspective est un ensemble de vues qui permet de travailler sur un thème donné. On a par exemple la perspective **Resource** qui permet de voir tout ce que contient le workspace, la perspective **Java** pour développer en Java, la perspective **CVS** pour gérer ses repositories, la perspective **Debug** pour le débogage, ... [ECL]
Présentiel	conscience de la disponibilité ou présence des autres membres du

	groupe
Servlet	une classe Java qui permet de créer dynamiquement des données au sein d'un serveur HTTP (wikipédia)
Système d'Information	un ensemble organisé de ressources qui permet de collecter, stocker, traiter et diffuser de l'information (wikipédia)
Tableitem (java)	classe SWT qui représente une ligne d'un objet de la classe Table
Thread (java)	un thread est donc une portion de code capable de s'exécuter en parallèle à d'autres traitements. Ils sont utiles dans bien des cas et parfois même nécessaires comme nous le verrons plus loin dans la section à propos de Swing. [ALW]
Vue (java)	les vues (ou views) sont des fenêtres graphiques d'Eclipse qui permettent d'afficher des informations : une vue "Package Explorer" va permettre de voir les packages Java du projet, une vue CVS History pour afficher l'historique d'un fichier sous CVS, ...etc. [ECL]
Widget	en informatique, le mot widget recouvre deux notions distinctes, chacune en relation avec les interfaces graphiques. Il peut alors être considéré comme étant la contraction des termes window (fenêtre) et gadget. Il peut désigner : un composant d'interface graphique, un élément visuel d'une interface graphique (bouton, ascenseur, liste déroulante, etc.) ou bien un widget interactif, un petit outil qui permet d'obtenir des informations (météo, actualité, dictionnaire, carte routière, pense-bête – en anglais post-it –, traducteur, etc.). (wikipédia)
Workflow	sert à décrire le circuit de validation, les tâches à répartir entre les différents acteurs d'un processus, les délais, les modes de validation, et à fournir à chacun des acteurs les informations nécessaires à l'exécution de sa tâche. (wikipédia)

Tables des matières

1 Contexte

Dans le cadre de mon cursus au CNAM, j'ai effectué un stage de 8 mois au CIRSO (Centre Informatique recouvrement Sud Ouest) pour valider mon diplôme d'ingénieur.

1.1 L'entreprise d'accueil

1.1.1 Le CIRSO et la branche recouvrement

Le CIRSO est un organisme de la branche recouvrement de la Sécurité Sociale. Celle-ci est une des 4 entités de la sécurité sociale, aux côtés de la branche maladie et accidents (assurances), la branche vieillesse (retraites) et la branche famille (allocations). Sa mission se décompose en 2 parties :

- Collecter les cotisations et contributions sociales auprès d'environ 5 millions de cotisants (entreprises ou particuliers).
- Redistribuer ces fonds aux organismes de distribution de l'aide sociale comme les Caisses d'Allocations Familiales (CAF) ou les Caisses Primaire d'Assurance Maladie (CPAM).

Le recouvrement est composé de plusieurs organismes du territoire national qui assument différents rôles.

- L'ACOSS (Agence Centrale des Organismes de Sécurité Sociale), c'est la caisse nationale des Urssaf. Sa mission principale est de gérer la trésorerie du régime général de la Sécurité sociale. Elle assure également la direction des organismes du recouvrement.
- Les 22 URSSAF (Unions de Recouvrement des cotisations de Sécurité Sociale et d'Allocations Familiales), ce sont des organismes de droit privé chargés d'une mission de service public : collecter des cotisations et des contributions destinées au financement du régime général de la Sécurité sociale.
- Les 8 CERTI (Centre d'Expertise Régional en Technologie de l'Information), le CIRSO en fait parti. Ce sont les organismes informatiques de la branche.

1.1.2 Pôle applications collaboratives

Pour assurer ses fonctions, le CIRSO est découpé en 9 pôles : production, IST, assistance, accompagnement, applications collaboratives, applications métiers, Erisa, RH et marché, agence comptable.

J'ai effectué mon stage au sein du pôle applications collaboratives (ou PAC).

Le pôle PAC est organisé en 2 services. Le premier correspond aux activités de la gamme « Démat-Interne » qui concernent les applications de workflow et de dématérialisation interne, pour le cœur de métier. Le second correspond au secteur Documentaire et Collaboratif (Doc&Coll), qui gère les solutions de mobilité, les média-conférences, la messagerie et les applications documentaires et collaboratives. Le pôle PAC partage l'activité du secteur Doc& Coll avec le CNIR Paris (un des 8 CERTI de la branche recouvrement) [RACIRSO0714].

Mon projet m'a amené à collaborer avec les deux services du pôle.

1.2 Le fonctionnement MOE/MOA

1.2.1 La MOA Watt

La MOA Watt est une direction de l'ACOSS : la DPMA (Direction de la Production et de la Maitrise des Activités)

1.2.1.1 Plan pluriannuel de gestion de contrat

La MOA raisonne en fonction de plans pluriannuels de gestion de contrat : la COG (Convention d'Objectifs et de Gestion). Ces contrats sont négociés entre l'administration et l'état. Ils décrivent les objectifs pour une durée de 4 ans. Ils fixent un cadre de travail, une marge de progrès, une stratégie. La COG actuelle court de 2014 à 2018.

Les contrats sont déclinés localement, chaque organisme en signe un avec l'ACOSS. Le CIRSO a signé le sien en Octobre 1014.

1.2.1.2 Comités de Gestion de Système d'Information (CGSI)

Dans le cadre de la COG, les relations MOA/MOE s'organisent autour de CGSI. C'est le premier lien MOA/MOE qui rassemble la MOA, la MOE et des représentants des utilisateurs des organismes du réseau. Les organismes ont un fonctionnement collégial en réseau. Chacun a à charge de nommer des représentants qui vont siéger dans ces CGSI sur les applications pour lesquels ils sont volontaires.

La MOA est tenue de suivre le CGSI. Elle doit planifier et tenir des réunions, étudier des demandes (quelques soient leurs origine cf. paragraphe suivant), arbitrer quand cela est nécessaire.

1.2.1.3 L'expression des besoins

Les besoins sont exprimés par le **national** ou bien ils proviennent du **réseau par les organismes**.

- A l'ACOSS des directions métiers comportent des personnes chargées de travailler pour pouvoir mettre en forme des projets nationaux sur une demande majeure comme par exemple la rénovation du système d'information, ce qui relève des traitements des cotisants RSI[1], de la DNS[2]… Ces besoins gouvernementaux vont se décliner ensuite au niveau des applications impactées. C'est projet sont prioritaires sur les autres.
- Ce sont essentiellement des demandes utilisateurs qui émanent d'organismes qui sont adressées à la MOA, qui les arbitre et les décline à la MOE. Cette dernière fonctionne

[1] RSI : Régime Social des Indépendants, ce projet impacte Watt car des demandes proviennent des cotisants du RSI

[2] DNS : Déclaration Nominative Simplifiée : projet gouvernemental sur lequel le président de la république insiste beaucoup pour simplifier les démarches administratives

avec une application nommée CAP pour gérer la remontée des besoins utilisateurs. Chaque organisme peut y déposer des demandes. Ces demandes passent au travers d'un certain nombre de filtres au niveau interrégional et national, si elles en ressortent elles doivent être arbitrées.

Une récente innovation voit une nouvelle source de besoins émerger, **l'enquête de satisfaction.** Elle permet de faire remonter toute une série de besoins sur des périmètres applicatifs plus ciblés et de permettre que ces remontées soient les plus complètes possibles.

La note moyenne mesurée par l'enquête de 2013, de la satisfaction apportée par l'application Watt, a poussé à faire une enquête supplémentaire dans laquelle des observations très détaillées des utilisateurs ont été reçues. Cette matière a été très intéressante à exploiter pour arbitrer au niveau des besoins à mettre en œuvre. En conséquence, les résultats de l'enquête 2014 ont été meilleurs.

Cependant recourir à cette enquête demande un investissement important. Il faut avoir suffisamment de temps et de ressource pour la mener et faire des investigations complémentaires. Ce temps n'est pas toujours suffisant alors que la MOA demande de travailler en majorité sur des besoins nationaux.

1.3 Mon travail

1.3.1 Ma place au sein des équipes projet

Mon travail a consisté en une étude d'implémentation de fonctionnalités collaboratives dans l'outil de workflow Watt (cf. paragraphe 3.1). J'ai donc travaillé au sein du projet Watt avec les membres de l'équipe de développement pour la réalisation et ceux de l'équipe de MOE pour la conception.

Cependant, Sametime (cf. paragraphe 3.2), la solution fournissant les fonctionnalités collaboratives à intégrer dans le workflow est gérée par le service Doc&Coll. C'est lui qui définit et maintient son infrastructure. De ce fait, j'ai aussi collaboré avec eux pour concevoir l'étude.

Cette collaboration a apporté son lot de difficultés. En effet, il fallait tout d'abord trouver des créneaux où l'ensemble des acteurs concernés dans chaque équipe étaient disponibles. Il a ensuite fallu adapter le discours a tout le monde afin que les tenants et aboutissants soient compris de tous.

1.3.2 Mon positionnement entre la MOE et la MOA

1.3.2.1 Fonctionnement altéré

Jusque là (pendant mon stage), le fonctionnement MOA/MOE vu précédemment (cf. paragraphe 1.2.1) avait été mis en sommeil car il s'agissait d'une période transitoire entre deux COG. Les groupes CGSI ont été malmenés pour diverses raisons, notamment car les acteurs ont été très sollicités pour la régionalisation (cf. paragraphe 4.1) et un certain nombre d'enjeux informatiques qui nécessitaient une concentration de ressources pas forcément compatible avec ce genre de fonctionnement.

L'étude que j'ai menée a donc souffert de cette dégradation. Comme nous verrons dans la suite du document, j'ai dû participer à la définition du projet et à la production du cahier des charges. J'ai donc en partie endossé le rôle de MOA pour mener à bien ce projet. Nous verrons aussi que j'ai assumé le rôle de MOE en participant à la proposition de solution pour la mise en œuvre des exigences.

1.3.2.2 Définition du projet et émergence des exigences

J'ai pu faire émerger des besoins et des exigences grâce à l'enquête de satisfaction. Celle ci a permis d'identifier des besoins concernant une partie de mon étude (la partie sur les notifications cf. paragraphe 5.3.5) et de juger qu'il y a des évolutions fonctionnelles telles que celles sur lesquelles j'ai travaillé qui pourraient répondre à un réel besoin.

J'explique dans le chapitre 5, comment j'ai procédé pour définir les besoins et proposer des solutions fonctionnelles.

Il est prévu de remonter les besoins de communication inter utilisateurs de Watt dans le cadre du PIA[3] 2015 en rapprochant cette solution des besoins exprimés par l'enquête de satisfaction.

2 État de l'art

2.1 Informatique de gestion et Business Process Management (BPM)

2.1.1 Evolution du traitement des données

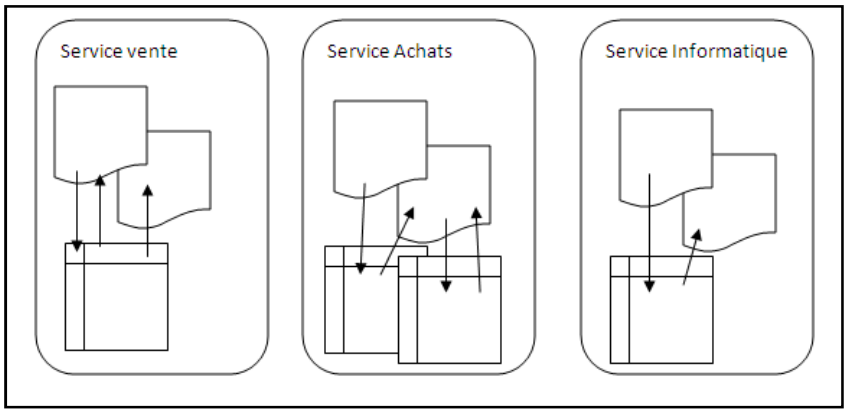


Figure 1 Gestion de l'information avant l'apparition des bases de données

La gestion des informations a connu avant l'apogée des nouvelles technologies, ère pré-base de données, des traitements « anarchiques ». Chaque service traite alors ses propres

[3] PIA : Plan Informatique Annuel. Il s'agit d'un livrable qui va retraduire les analyses des demandes qui ont été adressées, de l'arbitrage rendu en conséquence et qui informe la Direction Informatique des grands axes validés. Celle-ci va décider si oui ou non ces axes sont à mettre en œuvre et va donner les moyens de le faire.

informations et développe au cas par cas des programmes pour arriver aux fins souhaitées. Ce procédé a rencontré des limites assez rapidement avec en conséquence [**GO09**] une forte duplication des données et en corollaire, outre un coût de stockage important (à cette époque le stockage de données coutait une fortune), de nombreuses incohérences entre les duplications de la même donnée au sein des différents fichiers manipulés par les différents programmes.

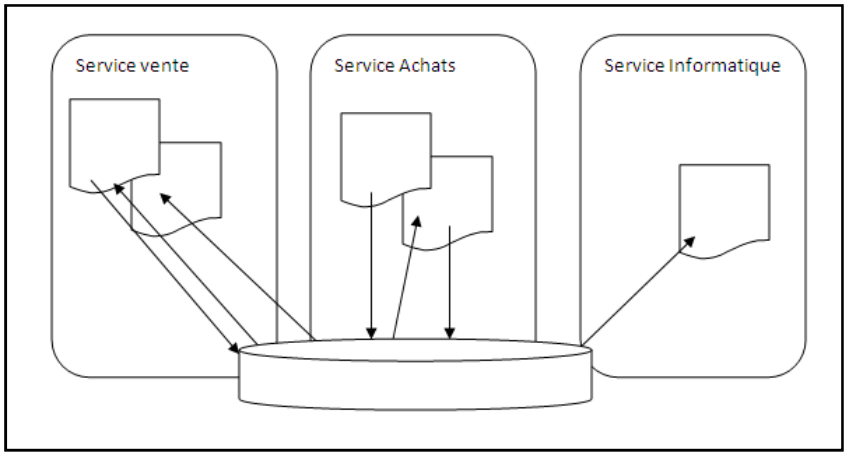


Figure 2 Mise en place des bases de données

La naissance des bases de données a permis de résoudre ces problèmes, leur but étant de gérer les données en seul et même endroit, de les organiser, structurer indépendamment des programmes.

Les ERP (Enterprise Resource Planning) sont les solutions informatiques mises en place par les entreprises pour gérer les bases de données. Ils reposent sur la promesse d'un traitement intégré et synchronisé des données [SL05].

2.1.2 L'apparition de la gestion par les processus métiers

A l'origine, les systèmes d'information ont été conçus pour supporter l'exécution de tâches individuelles. Les systèmes d'information d'aujourd'hui ont besoin de supporter et de manipuler les processus d'entreprise. Il ne suffit plus de se concentrer uniquement sur les tâches. Le système d'information doit également contrôler, surveiller et soutenir les aspects logistiques d'un processus métier. En d'autres termes, le système d'information doit également gérer le flux de travail grâce à l'organisation [SB10].

Un processus métier est défini dans [SB10] comme un ensemble d'activités qui sont exécutées en coordination dans un environnement organisationnel et technique. Ces activités réalisent en commun un objectif d'entreprise (commercial). Chaque processus métier est exécuté par une seule organisation, mais il peut interagir avec d'autres processus métier réalisées par d'autres organisations.

La figure ci-dessous, réalisée par la WfMC, regroupe tous les concepts associés au processus métier et résume ainsi son fonctionnement.

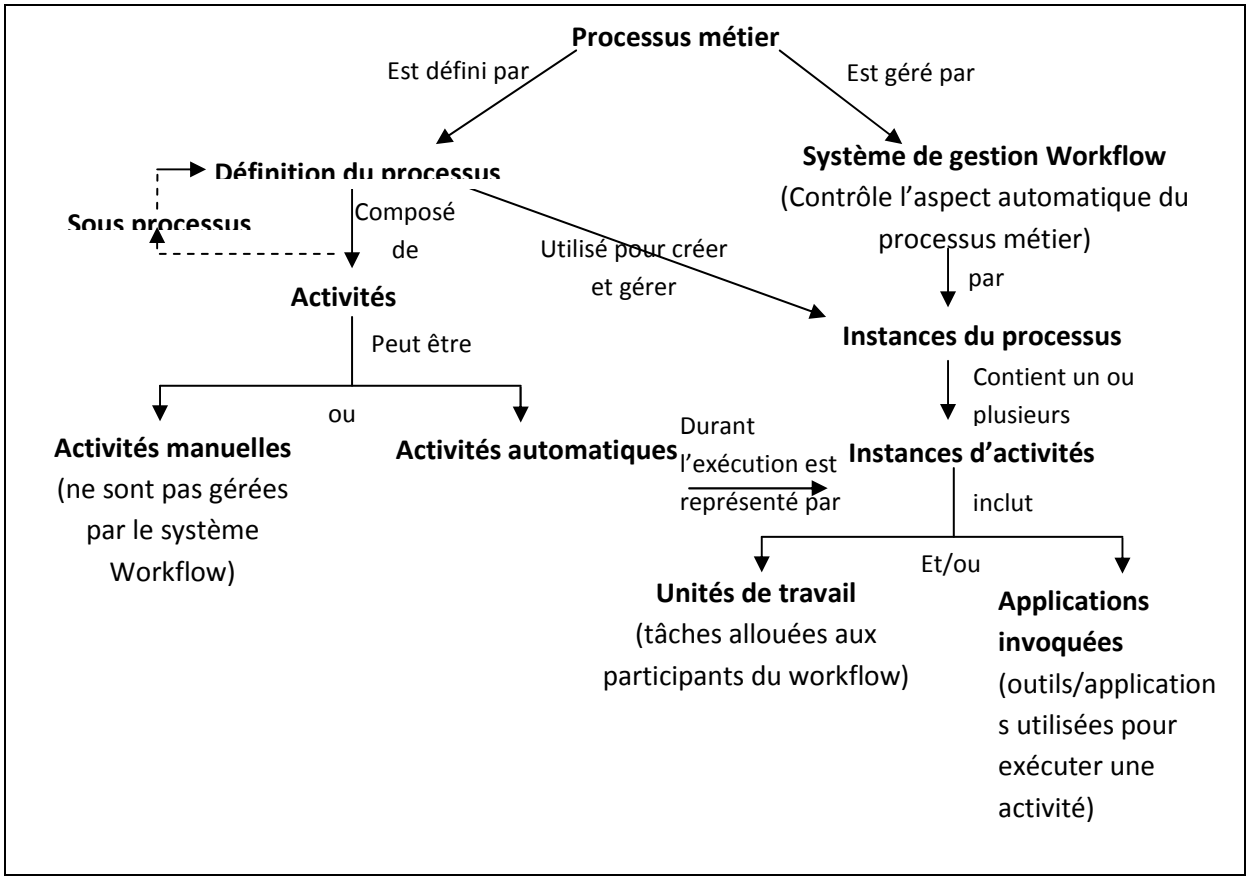


Figure 3 Relations entre les terminologies du processus [Wf99]

2.1.3 Le cycle de vie du développement des processus et but du BPM

Il existe une rupture entre la modélisation métier et la modélisation technique qui entraine des écarts entre ce qui est effectivement attendu et ce qui est réalisé. Au fur et à mesure des réalisations, les spécifications fonctionnelles et techniques évoluent et creusent d'avantages l'écart existant devenant de ce fait incohérentes entre elles. Après plusieurs cycles, il devient moins couteux et plus facile de redévelopper une nouvelle application plutôt que de modifier une énième fois l'implémentation.

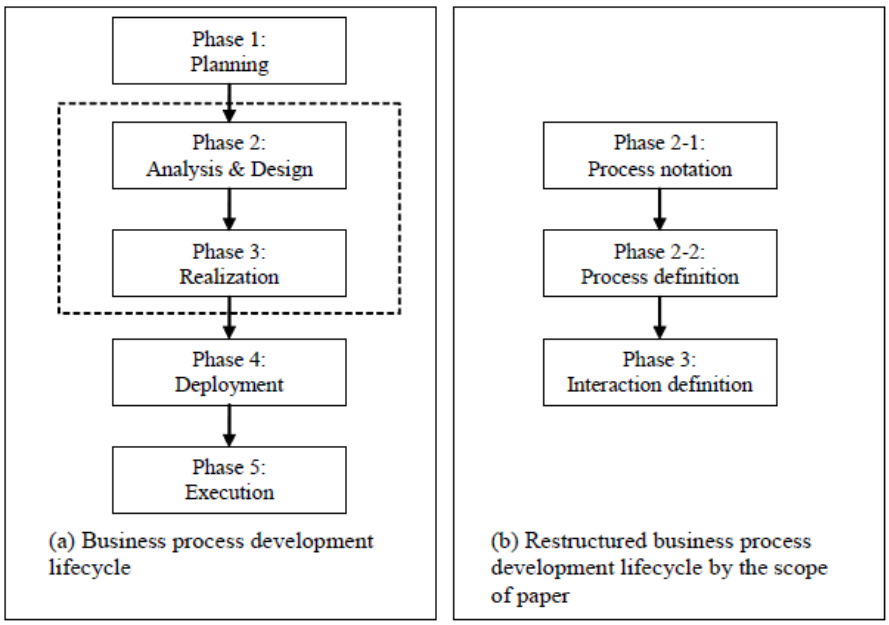


Figure 4 Cycle de vie du processus recherché avec la modélisation des processus métier [PA06]

Une solution à ce problème est la mise en place d'une modélisation des processus métier unique et compréhensible par toutes les parties concernées au long du cycle de vie du développement, c'est-à-dire les analystes, les développeurs, les utilisateurs et les décideurs. Le schéma (b) de la figure ci-dessus nous propose de segmenter les phases d'analyse et de réalisation pour mettre en place une définition des processus métiers. Elle se compose en trois phases :

- Une phase de notation où les processus seront définis à un haut niveau d'abstraction à l'aide de forme graphique pour permettre une compréhension du processus par tous les acteurs. Le concepteur [BO10] y décrit seulement la structure, les ressources nécessaires et des interfaces du processus grâce à des langages graphiques comme UML et BPMN.

- Une phase de définition du processus : lors de cette phase le concepteur va transcrire les notations vers un langage textuel (comme XPDL, WS-BPEL) dans lequel nous trouverons la définition des activités, des ressources…

- Une phase de définition des interactions : cette phase va s'intéresser à définir les interactions entre différentes organisations sur un même processus métier. Cette définition se fera au moyen d'un langage (comme WS-BPEL) dans lequel nous retrouverons des informations sur le format des messages échangés, des ports de communication, des protocoles de transports utilisés…

[SL05] nous montre dans le schéma ci-dessous une des problématiques techniques posée par l'évolution des ERP vers le BPM. Il faut extraire la logique métier des applications pour en faire une couche à part entière aux côté des couches « présentation », « application » et « données » qui définissent aujourd'hui la structure des architectures n-tiers. Cette extraction

doit permettre d'orchestrer les services du processus interne à l'entreprise et de chorégraphier l'enchainement des opérations interentreprises.

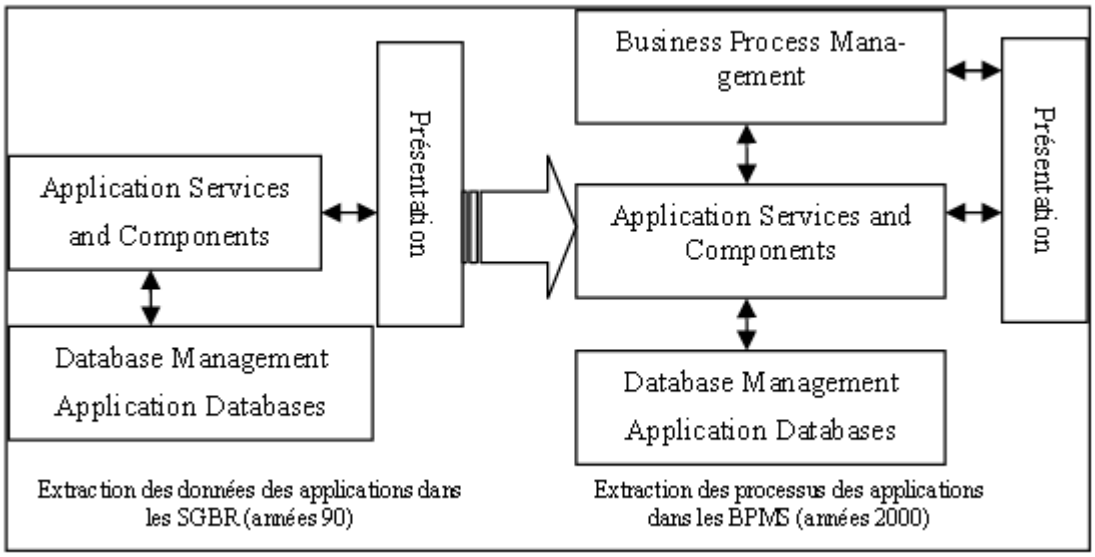


Figure 5 Changement de paradigme introduit par le BPM dans le SI [SL05]

2.1.4 Systèmes de gestion de workflows

Le workflow peut être défini [BE09] comme étant " l'automatisation, totale ou partielle, d'un processus métier, dans lequel les tâches passent d'un participant à l'autre pour être effectuées, selon un ensemble de règles procédurales". Il organise les processus métiers de deux façons : il définit l'ordre dans lequel les tâches doivent être accomplies, et stipule les étapes et les points de contrôle où interviennent des êtres humains.

Les systèmes de gestions de workflow, aussi trouvés sous le nom de BPMS (Business Process Management System), vont [**GO09**] définir, créer et gérer l'exécution de workflow grâce à l'utilisation d'un logiciel capable d'interpréter les définitions des processus, d'interagir avec les participants et, lorsque cela est requis, d'invoquer les outils et les applications.

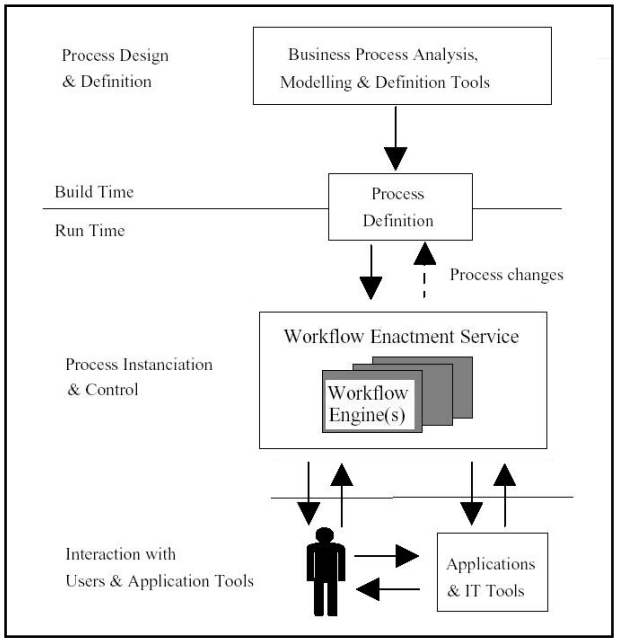


Figure 6 Système de gestion de workflow [Wf99]

Ces technologies vont donc s'appuyer sur les définitions élaborées lors de la phase de modélisation pour les implémenter vers les applications et les utilisateurs. La Figure 6 résume les différentes phases abordées jusqu'ici. On retrouve au sommet la partie « Process design et definition », c'est la phase lors de laquelle ont été analysés les processus et retranscrits sous forme de graphes. La partie centrale « build time/Run time » correspond à la phase où le processus métier est traduit sous forme de language textuel, par exemple sous la forme d'un fichier WS-BPEL. C'est ce document qui sera interprété par les moteurs de workflow (dans la partie basse) pour permettre l'exécution du processus.

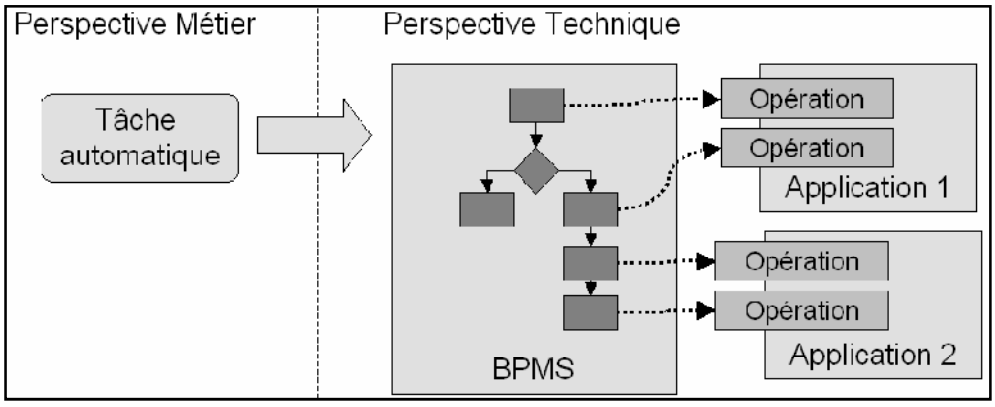


Figure 7 Flux d'activités des processus métiers [BR08]

La Figure 7 nous fournit une autre illustration du cheminement du processus métier entre sa conception et sa mise en application d'un point de vue technique. Dans un premier temps, les concepts introduits par le processus vont être [**GO09**] concrétisés à l'aide de notation BPMN et XPDL par exemple. La première est caractéristique des notations graphiques utilisées à l'interface entre les hommes du métier et les concepteurs informatique, la seconde des formalismes opérationnels exprimés dans un dialecte XML pour assurer la mise en œuvre des processus en intranet et extranet.

2.1.5 Limites

Dans [SL05], les auteurs constatent que les « méthodes de modélisations sont performantes pour décrire des procédures de production mais qu'elles ne permettent pas de représenter la façon dont les gens s'ajustent mutuellement pour s'adapter à des situations changeantes ».

Ils expliquent que le lien entre le BPM et le système d'information pose problème. Tout d'abord d'un point de vue technique, avant qu'un processus arrive à l'état d'exécutable, les techniciens doivent renseigner un nombre importants d'informations techniques. De ce fait, se passer d'une phase d'implémentation leur paraît illusoire. Puis d'un point de vue organisationnel où l'apport du BPM amène à repenser toute l'organisation de l'entreprise pour prendre en compte les activités centrées autour du BPM et du système d'information.

Enfin, ils ajoutent que « le processus est une représentation incontournable de la firme, mais qui complète bien d'autres formes de représentations de l'activité collective et de sa valorisation ». Leur démarche est alors « d'insister sur la nécessiter d'articuler les processus aux interactions humaines qui les rendent performants ».

Nous verrons plus loin (cf. paragraphe 2.3.1), qu'ils proposent une solution qui passe par une modélisation des pratiques de travail.

2.2 Applications collaboratives

2.2.1 Définition

La notion d'application collaborative est souvent retrouvée sous le mot « Groupware » ou moins fréquemment dans sa traduction française « collecticiel ». Dans [EGR91], Clarence Ellis définit le groupware comme « un système informatisé qui soutient le travail des groupes de personnes pour un but commun et qui fournit une interface pour un environnement partagé ».

Le courant de recherche CSCW (Computer Supported Collaborative Work), en français TCAO (Travail Coopératif Assisté par Ordinateur) a émergé au milieu des années 80. Ce domaine propose des modèles de représentation des activités coopératives. L'AFCET [AF94] définit cette discipline comme suit : « la collectique (ou CSCW) regroupe l'ensemble des techniques et des méthodes qui contribuent à la réalisation d'un objectif commun à plusieurs acteurs, séparés ou réunis par le temps et par l'espace, à l'aide de tout dispositif interactif faisant appel à l'informatique, aux télécommunications et aux méthodes de conduite de groupe ».

Les groupware sont ainsi des outils supportant les pratiques d'activités collaboratives regroupées dans le domaine de recherche des TCAO.

2.2.2 Inventaire des outils et classification

Les outils découlant des pratiques des TCAO sont nombreux, en voici une liste non exhaustive : workflow, agenda partagé, outils bureautiques partagés (accessibles à plusieurs en même temps : traitement de texte, tableur…), webconférence, progiciel de gestion intégré (ou ERP), plateforme collaboratives, blogs, wiki, forum, listes de diffusion, listes de discussion, mails, fax, téléphone, messagerie instantanée, gestion électronique de documents, tableaux blancs, flux RSS, visioconférence, podcast, vidéo en ligne…

Leur choix peut se faire selon plusieurs critères, en fonction du lieu, du moment, de la taille du groupe, de l'objet du travail et/ou de l'ergonomie.

2.2.2.1 Le trèfle des 3C

Le groupe SCOOP a proposé dans [BLC95] une classification fonctionnelle sous la forme du trèfle des 3C qui s'inspire des travaux d'Ellis. Ce trèfle repris ci-dessous dans la Figure 8 par Gilles Balmisse dans [BA04].

Ce modèle regroupe les services trouvés dans les groupware dans trois espaces : la production (ou collaboration), la coordination et la communication.

- La production ou collaboration : l'espace de production est celui où on va trouver des outils de travail permettant la participation d'une ou plusieurs personnes pour aboutir à des projets, objets, œuvres communes. L'outil de gestion de contenu comme wikipédia offre cette prestation.

- La coordination : la fonction de coordination permet d'identifier et organiser les actions, tâches, séquences, ressources… L'outil de workflow est adéquat pour remplir cette fonction. Contrairement à l'espace de production qui offre une vue statique du système, celui-ci donne une vue dynamique.

- La communication : cet espace assure aux acteurs la possibilité d'échanger des informations sous forme par exemple de messages électroniques ou par messagerie instantanée.

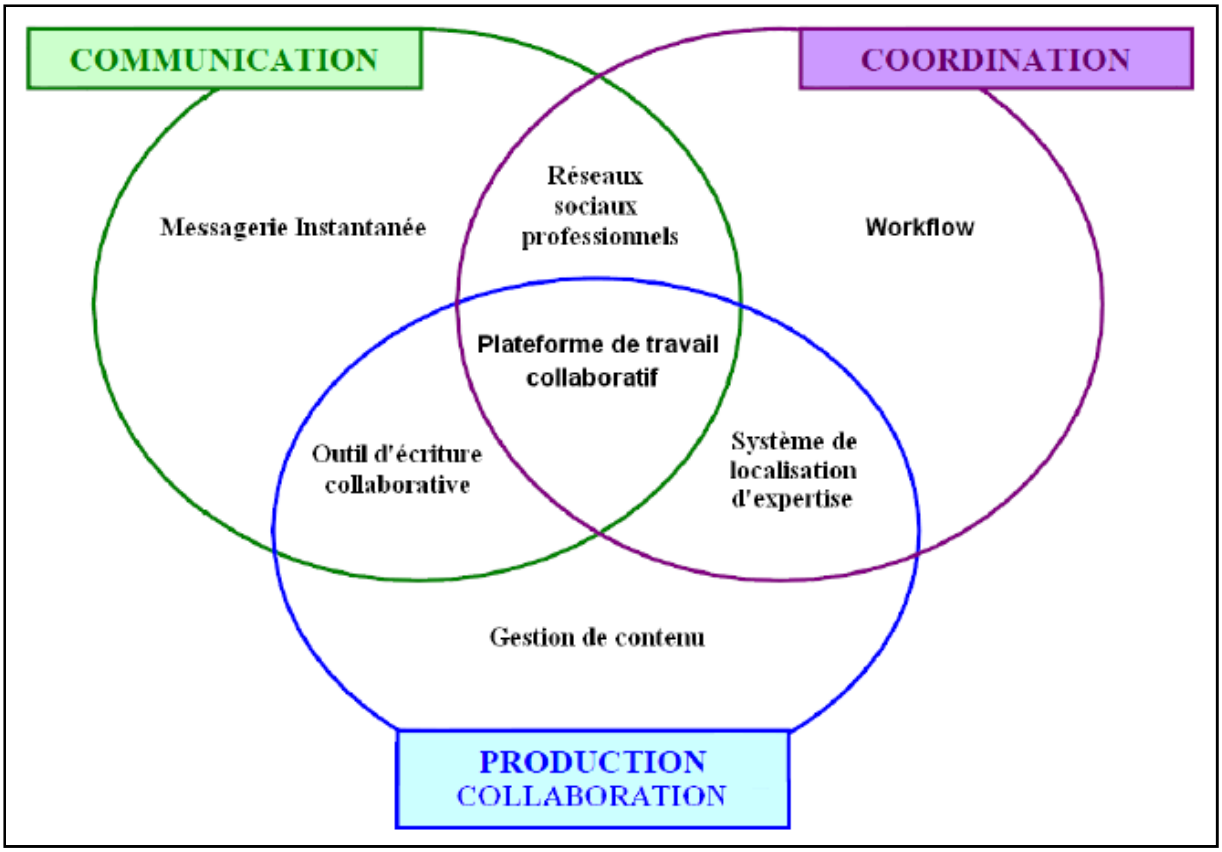


Figure 8 Le trèfle fonctionnel des 3C [BA04]

Comme on peut le voir sur le modèle du trèfle, les trois espaces s'entrecroisent. Certains outils se peuvent proposer des services d'un ou plusieurs espaces. Par exemple, l'outil d'écriture collaborative donne la possibilité de partager des informations entre plusieurs acteurs (espace communication) avec pour finalité un document (espace production). Une plateforme de travail collaboratif rassemble quand à elle les trois fonctionnalités.

2.2.2.2 Classification spatio-temporelle

Une autre classification proposée pour les applications collaboratives est la classification spatio-temporelle.

Elle a d'abord été abordée par Johansen (chercheur affilié au MIT –Massachusetts Institute of Technology-) en 1988 sous la forme d'une matrice « Temps-Lieu» (voir Figure 9 ci-dessous). Elle prend en compte les facteurs de lieux et de moment. Le premier critère définit la distance entre deux ou plusieurs acteurs. Dans le cas où les acteurs sont face à face on considère qu'ils sont sur le même lieu, mais ils peuvent être séparés par un mur, un étage, ou même se trouver dans différents bâtiments, ville ou pays. L'autre critère détermine si les acteurs collaborent en même temps de manière synchrone ou en différé (asynchrone).

Ainsi dans le cas d'une communication synchrone locale, nous pouvons organiser une réunion dans une salle alors qu'à distance elle nécessitera un outil de type chat, visioconférence, téléphone… La communication asynchrone locale peut se matérialiser sous la forme d'un

post-it laissé sur le bureau d'un collègue tandis que lorsque les acteurs sont distants nous pourrons utiliser des forums, liste de diffusions, mails…

TEMPS / LIEU	Même (Synchrone)	Différent (Asynchrone)
Même (Local)	Face à face (salle de conférence, salle de presse)	Interaction asynchrone (programmation de projet, outil de coordination)
Différent (Distant)	Synchrone distribué (Editeurs partagés, interfaces vidéos)	Asynchrone distribué (email, weblog)

Figure 9 Matrice Temps Lieu

Cette matrice est ensuite renommée matrice « Temps-Espace » par Clarence Ellis puis reprise dans les travaux de Grudin [GR94] (voir Figure 10 ci-dessous). Grudin rajoute une notion de prévisibilité sur chacun des axes. Elle permet de prendre en compte si le lieu et le moment sont prévisibles ou pas.

TEMPS / ESPACE	Même	Différent prévisible	Différent Imprévisible
Même	Simplification de réunion	Relais de travail	Salle d'équipe de projet
Différent prévisible	Télé conférence	Courrier électronique	Editeur collaboratif
Différent Imprévisible	Séminaire multi-rôle interactif	Forum électronique	Workflow

Figure 10 Matrice Espace Temps

2.3 Méthodes et représentation des phases de travail coopératif associées aux workflows

2.3.1 Une proposition de modélisation [SL05]

Partants du constat que « les outils de BPM ne permettent pas de représenter les communications informelles qui se déroulent dans les situations de travail » [CSSH98], les auteurs de [SL05] se sont intéressé aux modèles du CSCW ainsi qu'à plusieurs propositions et analyses menées en amont pour proposer une modélisation descriptive des pratiques de travail. Pour se faire, ils associent « différents modèles, qui permettront de représenter le

travail au sein d'un groupe de manière plus détaillée que le BPM, mais moins détaillée, et moins en profondeur que les modèles cognitifs ». Ils ajoutent dans leurs modèles des dimensions physiques, cognitives et sociales.

- Modèle d'agent (acteurs impliqués)
- Modèle d'activités (comportement des acteurs et des objets dans leur environnement)
- Modèle d'objets (objets qui constituent l'environnement des acteurs)
- Modèle de contexte (représentation du contexte dans lequel se déroulent les activités)
- Modèle temporel (contraintes sur les activités)
- Modèle de connaissance (actions de production des connaissances)
- Modèle de communication (action de communication entre acteurs et objets)

La démarche des auteurs est de proposer un cadre de modélisation des pratiques de travail, certaines dimensions ont déjà été testées mais la « modélisation des comportements humains dynamiques s'inscrivant dans des environnements de travail eux-mêmes dynamiques n'en est encore qu'à ses débuts ».

2.3.2 Une autre démarche de modélisation [BBD04]

Pour les auteurs de [BBD04], « il est parfois difficile pour un concepteur de disposer d'informations à la fois complètes et pertinentes, au moment opportun, d'où une nécessité d'interaction avec les autres acteurs ».

Ils s'inspirent du concept de situation pour proposer une modélisation. Celle ci vient enrichir les modèles basés sur les processus métiers tout en apportant « une aide pour la compréhension des différents mécanismes qui régissent les activités collaboratives ».

Au concept de situation, les auteurs de [BBD04] proposent la définition suivante : « la situation est un ensemble d'entités et d'interactions (de différentes natures) qui caractérise de façon globale l'environnement externe dans lequel l'acteur mobilise sa compétence ». Ils identifient trois niveaux de description d'une situation : la **situation concrète** (ensemble des entités réelles existantes), la **situation observable** (ensemble des entités et relations observables par un acteur) et la **situation utile** (ensemble des entités et relations observées et considérées comme pertinentes par un acteur à un moment donné). Ils y associent une autre forme : la **situation modélisable** « qui contient l'ensemble des entités et relations observées ou explicitées par un modélisateur externe qui a une vision globale de l'environnement ».

La modèle de situation proposé (voir Figure 11) comporte un ensemble **d'entités E** reliées par des **relations R**. Il couvre les différents aspects de l'environnement de travail (structurel, opérationnel et dynamique).

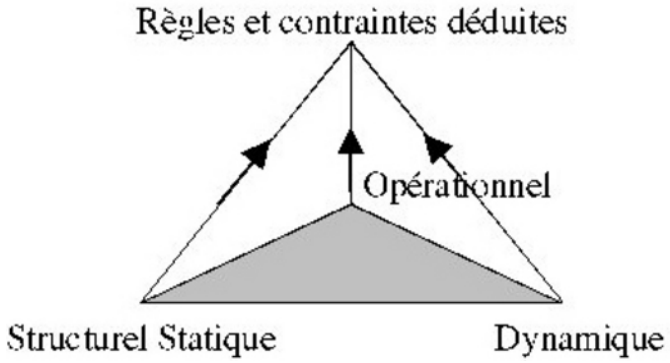


Figure 11 Les différentes facettes du modèle de la situation [BBD04]

Les **entités** se déclinent sous plusieurs formes, les deux principales sont les **entités de base EB** (acteurs humains et matériaux) et **interactionnelles EI** (représentent un lien entre entités). Elles sont définies par leur type (base ou interactionnel), leur état (actif/inactif) et leur poids selon le degré de pertinence pour un acteur par rapport à son activité (0 ou 1).

Il y a trois types d'entités interactionnelles :

- opérationnelles : qui fait quoi

- communautaires : établissent un lien entre entités mères et entités filles

- transactionnelles : pour renseigner sur la nature des échanges que peuvent avoir les acteurs dans une situation donnée, 4 formes possibles : communication, coopération, coordination, collaboration

Une **relation** Rij relie une entité (de base ou interactionnelle) Ei à une entité interactionnelle Ej. La relation est définie par les deux entités liées, son rôle (acteur, client, objet, support, manager) et son poids.

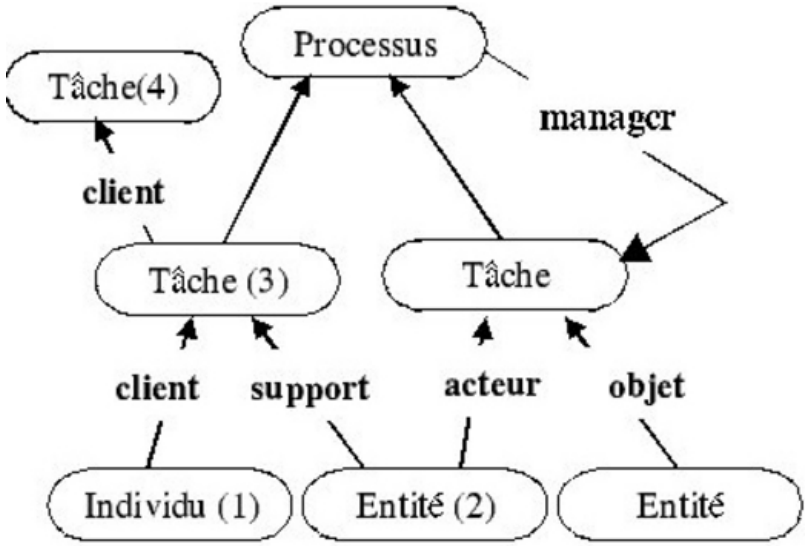


Figure 12 Les rôles dans les entités opérationnelles [BBD04]

Dans l'exemple ci-dessus, la relation R_{13} relie l'entité Individu 1 à l'entité interactionnelle 3 avec pour rôle client, elle est définie comme suit : R_{13}= {E_1 ; E_3 ; client ; poids}.

2.3.3 La méthode OSSAD

Le but de [SN96] est de définir une méthode d'analyse et de conception associant le workflow et autres situations du travail coopératif, et de modéliser les interactions (synchrones, asynchrones, formelles ou informelles) entre coopérants. Après avoir étudié plusieurs méthodes usuelles comme Merise, SADT[4], SART[5]… pour lesquelles l'aspect organisationnel n'est représenté que partiellement, l'auteur s'est tourné vers la méthode OSSAD[6].

La méthode OSSAD a été conçue dans le cadre du programme ESPRIT[7]conduit de 1985 à 1990 par une équipe multinationale et multidisciplinaire de consultants. Cette méthode inclut les acteurs dans le système à concevoir, elle est tournée vers l'organisation des hommes plutôt que celle des données et des traitements. Elle permet d'analyser comment différentes personnes coordonnent leurs tâches en vue de fournir un résultat global.

OSSAD se base sur plusieurs modèles. Un modèle abstrait qui représente les objectifs et répond au QUOI ? et POURQUOI ? Un autre modèle descriptif qui décrit les moyens pour atteindre ces objectifs et répond aux questions COMMENT ? QUI ? et QUOI ?

Le modèle abstrait permet de cartographier les processus. « Il se base sur un découpage de l'organisme en sous systèmes aux objectifs cohérents (FONCTIONS) décomposables à leur tour en utilisant le principe du zoom. Au niveau le plus détaillé de l'analyse, les fonctions non décomposées sont nommées ACTIVITES» [SN96]. L'exemple ci-dessous représente le modèle abstrait de la demande de prêt d'un client.

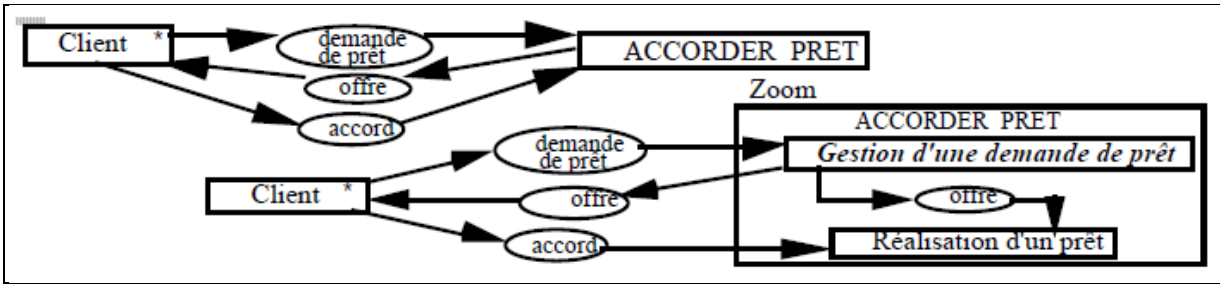


Figure 13 Exemple de modèle abstrait (Source [SN96])

La transition entre le modèle abstrait et le modèle descriptif se fait par une matrice Activité/Rôles (voir Figure 14). Elle permet de définir quel rôle (concept descriptif) remplit quel activité (concept abstrait).

[4] SADT : *Structured Analysis and Design Technique*
[5] SART : *Structured Analysis for Real Time*
[6] OSSAD : *Office Support System Analysis and Design*
[7] ESPRIT : *European Strategic Programme for Research in Information Technology*

	Client *	Agent service prêts	Chef service prêts	Agent comptes courants
Gestion d'une demande de prêt	X	X	X	
Réalisation d'un prêt	X	X	X	X

Figure 14 Matrice Activité/Rôle [SN96]

Enfin, le modèle descriptif décrit les moyens et les ressources technologiques de l'organisation. Il est composé de trois types de modèles : rôles, procédures et opérations.

Les modèles de rôles et procédures représentent la circulation des ressources d'information. Ils permettent de faire le lien entre les rôles pour le premier et entre les procédures pour le second. « Ils élaborent une représentation statique de l'organisme : aucun élément chronologique n'y figure » [SN96].

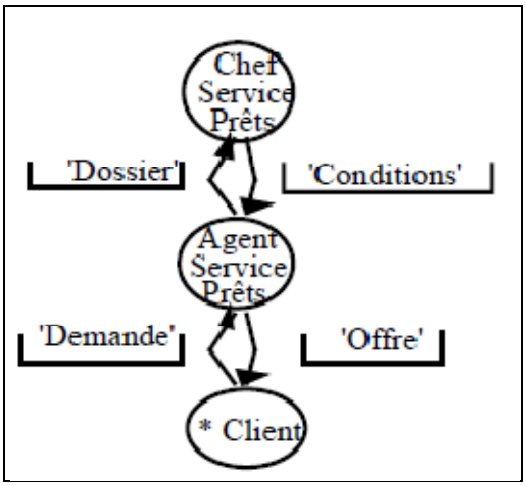


Figure 15 Modèle descriptif de rôles [SN96]

Le modèle descriptif d'opérations schématise les tâches et permet de faire le lien avec la matrice Rôle/Action. C'est le modèle le plus détaillé, on y trouve qui fait quoi et dans quel ordre.

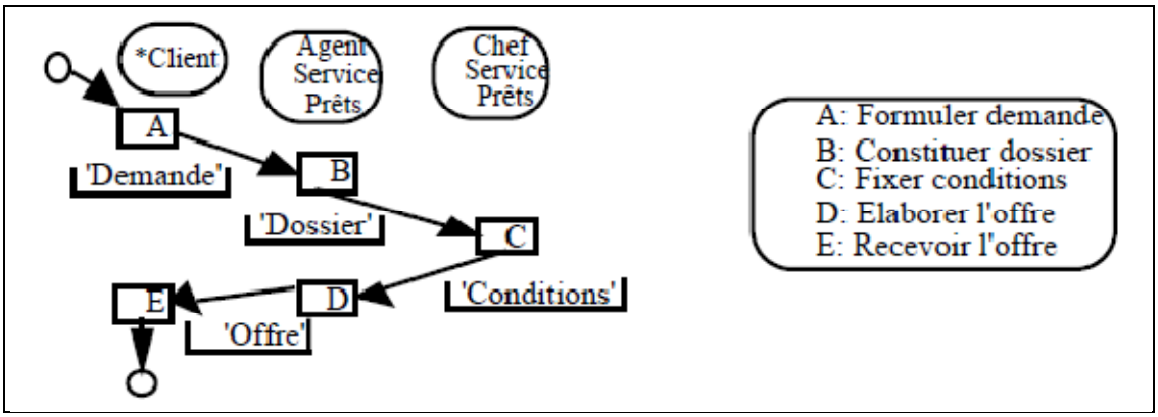


Figure 16 Modèle descriptif d'opérations [SN96]

La méthode OSSAD est présente dans les solutions : Win Design et Workey (C-Log).

2.4 Des outils de gestion des processus métiers

Comme nous venons de le voir dans le précédent paragraphe, des méthodes pour la modélisation des processus métiers ont été proposées. J'ai cité précédemment Workey qui est une solution qui utilise la méthode OSSAD mais il existe tout un beaucoup d'autres outils qui vont permettre de modéliser les processus métiers. Ces outils de gestions (appelés BPMS) sont généralement plus complets, ils permettent la modélisation des processus métiers mais aussi leur intégration, exécution et pilotage. Leur fonctionnement est détaillé dans le paragraphe 2.1.4.

2.4.1 Bonita BPM

Bonita BPM est un des outils qui nous vient immédiatement à l'esprit lorsque nous parlons de BPMS. Il fait parti des premiers outils apparus sur le marché et est open source. Sa version actuelle est Bonita BPM 6.5.2. Il est composé de trois modules : Studio, Engine et Portal.

Bonita BPM Studio permet de modéliser les processus, définir les données, ajouter des connecteurs vers les systèmes tiers (messagerie, progiciel de gestion intégré, GED, base de données, etc.) et créer les formulaires. Il utilise le langage de notation BPMN.

Bonita BPM Engine est le moteur de workflow de la suite. Il va gérer le déroulement des processus métiers.

Enfin Bonita BPM Portal va permettre aux utilisateurs de suivre les processus métiers et les dérouler. Ils vont ainsi pouvoir les surveiller et gérer les erreurs.

2.4.2 Watt

Watt, l'outil de workflow procédural des URSAFF est construit en plusieurs modules comme Bonita BPM. Une présentation plus détaillée de l'outil est disponible au chapitre 3.1. Développé initialement sous Lotus Notes, il a été décidé de le refondre sous Eclipse RCP en 2010 après une étude des divers produits BPMS sur le marché. A ce moment là, les décisionnaires du projet n'ont pas trouvé d'outils qui les satisfassent totalement. Ils ont donc décidé de le recréer entièrement en Java.

On y retrouve comme dans Bonita des interfaces de modélisation et de pilotage et traitement des processus.

Une première interface appelée Watt Graph (contenue dans le module Watt Administration) permet la modélisation des processus métiers. Elle utilise la notation UML.

Watt Pilot et Watt Traitement permettent elles de surveiller les processus en cours et de les traiter.

Des modules complémentaires permettent de gérer l'injection de nouveaux documents dans le workflow et le paramétrage technique et fonctionnel de l'application.

2.5 Orchestration et chorégraphie

Plusieurs organisations peuvent collaborer sur des processus métier commun. Les systèmes de gestion des processus métiers comme Bonita sont une solution à l'intégration des processus au sein d'une même entreprise. Ils vont permettre de passer de processus métier à leur implémentation par une couche de service (SOA, REST…). Deux concepts sont définis pour organiser les processus : l'orchestration et la chorégraphie.
La partie orchestration traite alors l'enchainement de services web, sans intervention humaine.

L'orchestration de services permet de spécifier l'ordre d'exécution des services, et de décrire le flot de contrôle du processus qui sera géré par un moteur d'exécution [GO09]. Dans le schéma de la Figure 17, l'orchestration correspond aux activités internes aux organisations.

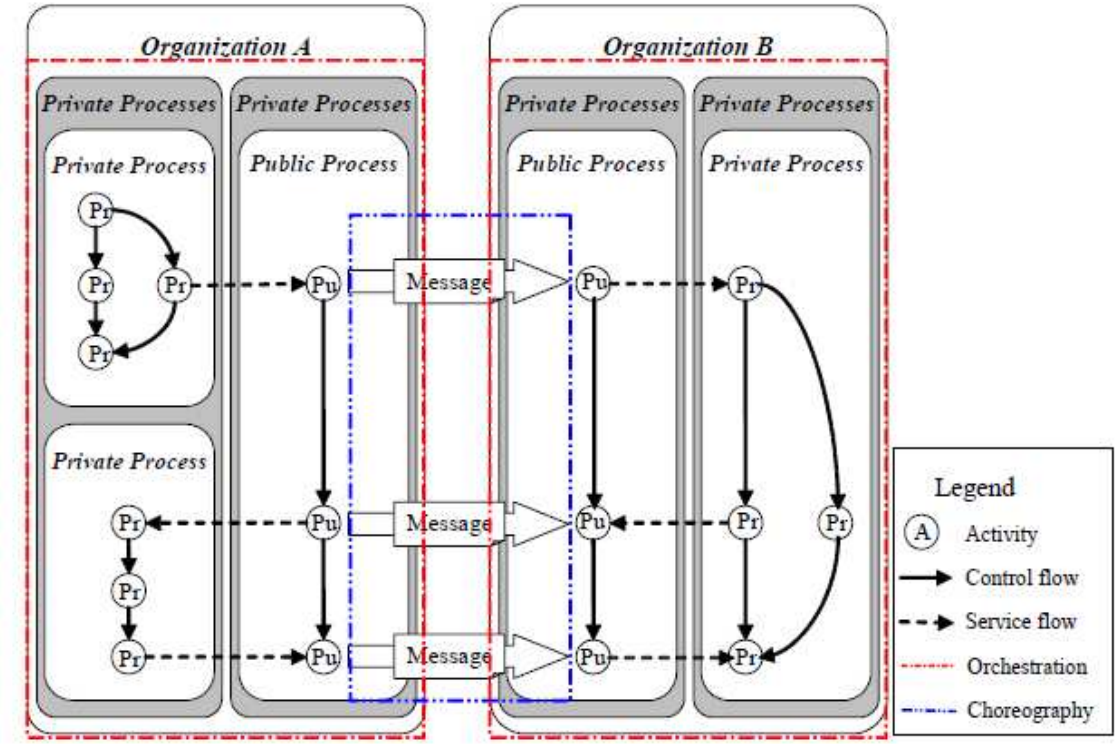


Figure 17 Organisation des services web [JU06]

La chorégraphie se place dans une vision plus collaborative dans laquelle plusieurs processus échangent des messages en ayant une vue commune des différentes interactions, des différents messages et des activités de chaque partenaire [GO09]. Cette partie correspond sur la Figure 17 aux flèches pleines représentant des messages échangés entre les partenaires.

2.6 Conception : architecture REST

Les systèmes de gestion des processus métiers sont une solution à l'intégration des processus au sein d'une même entreprise. Dans le cas d'une collaboration entre plusieurs partenaires, client et fournisseurs par exemple, les services web ont été appréhendés pour échanger des messages et partager des informations hors de l'entreprise.

REST propose un style d'architecture pour la mise en œuvre de services web. Nous verrons plus loin que cette architecture sert de base à la solution utilisée pour l'implémentation des fonctionnalités collaboratives dans Watt.

2.6.1 REST : généralités

REST (REpresentational State Transfer) n'est pas un format, un standard ou un protocol mais un style architectural inventé par Roy Fielding en 2000 [FR00]. Il permet de construire des applications (web, internet, services web) en utilisant le protocole http et la syntaxe universelle URI pour adresser les ressources.

Son utilisation prime sur celles d'autres protocoles existants tels que SOAP. Les acteurs Google, Amazon, Flickr, Twitter, Facebook et bien d'autres encore s'en servent pour intégrer leurs services dans les applications des utilisateurs.

Le terme RESTfull est souvent employé pour désigner une architecture qui respecte les principes de l'architecure REST.

Dans les outils existants pour appeler des services REST, nous retrouvons CURL, Postman (extension de Chrome), POSTER (plugin FIREFOX) et SOAPUI. Plusieurs plateformes permettent de développer ou appeler des services REST : JAX-RS pour Java, Php, .Net, Python…

2.6.1.1 Propriétés

Roy Fielding a énoncé les contraintes suivantes dans sa thèse pour composer le modèle REST :

- Client-serveur : la problématique d'interface côté client est séparée de celle de gestion des données par le serveur pour augmenter la portabilité de l'interface mais aussi permettre à chaque partie d'évoluer indépendamment.

- Sans état : une requête du client vers le serveur ne doit prendre en compte que l'état dans lequel se trouve le client à l'instant où le message est envoyé. Elle ne peut pas utiliser de contexte lié au serveur.

- Mise en cache : « Les contraintes de cache imposent que les données d'une réponse à une requête soient marquées, implicitement ou explicitement, comme pouvant être mises ou non en cache ». [FR00]

- Une interface uniforme : cette contrainte agit selon 4 règles essentielles, l'identification des ressources, la manipulation des ressources par des représentations, des messages auto-descriptifs et l'hypermédia comme moteur de l'état de l'application.

- Système en couche : le client ne se préoccupe que des composants avec lesquels il est en communication.

- Code à la demande (facultatif) : « possibilité pour les clients d'exécuter des scripts obtenus depuis le serveur. Cela permet d'éviter que le traitement ne se fasse que du côté serveur et permet donc de faire évoluer les fonctionnalités du client au cours du temps. » [WIK]

2.6.1.2 Architecture orientée ressources

La terminologie d'architecture orientée ressource (ou ROA) est proposée par les auteurs de [SL07]. Cette architecture soumet des bonnes pratiques REST appliquées aux services web. REST essaie de nous amener à penser au web comme à un ensemble de **ressources** et non de pages web qui ont plusieurs **représentations** possibles. Le client et le serveur communiquent par le biais de messages utilisant le protocole http. Des **méthodes** sont utilisées dans ces messages selon leur nature. Ces concepts principaux sont explicités ci dessous.

- Ressources

Les ressources sont tous les objets identifiables dans un système (fichier, image, vidéo, personne, valeur d'un objet, groupement d'objets…). Chacune possède une URI unique. Elles peuvent cependant en avoir plusieurs car celle-ci est construite de façon hiérarchique.

Figure 18 Exemple d'URIs [BAR]

Dans l'exemple de la figure ci-dessus, chaque point suivant est une ressource du système :

- Le livre « Le prisonnier d'Azkaban »
- Tous les livres Harry Potter
- Tous les livres d'aventure
- Tous les livres sont chacun une ressource du système.

Ils vont permettre de construire les adresses de façon hiérarchique. C'est-à-dire que tous les livres seront représentés à l'adresse /books/, tous les livres d'aventure à l'adresse /books/aventure, les livres de sciences fiction qui sont une collection de livres autres que ceux d'aventure pourraient être à l'adresse /books/SF/ …

Le livre « Le prisonnier d'azkaban » peut être demandé par le biais de 2 URI :

- « /books/aventure/harrypotter/2 »
- « /books/aventure/harrypotter/the_prisoner_of_azkaban »

Il pourrait exister un ressource « dernier livre ajouté » avec pour URI « /books/dernierlivre ». Si le livre cité précédemment est le dernier à avoir été ajouté, il serait alors aussi accessible par cet URI.

- Méthodes

Une fois les ressources identifiées, il faut définir les méthodes (ou verbes en http) que chacune peut supporter. Contrairement à une architecture RPC, elles ne sont pas infinies. REST reprend les 4 méthodes élémentaires de HTTP : POST, GET, PUT et DELETE. Elles sont souvent assimilées au terme CRUD (CREATE, RETRIEVE, UPDATE, DELETE) utilisé en base de données.

- ✓ POST (~ CREATE) pour créer une nouvelle ressource
- ✓ GET (~RETRIEVE) pour obtenir la représentation d'une ressource
- ✓ PUT (~UPDATE) pour mettre à jour la ressource
- ✓ DELETE (~DELETE) pour supprimer la ressource

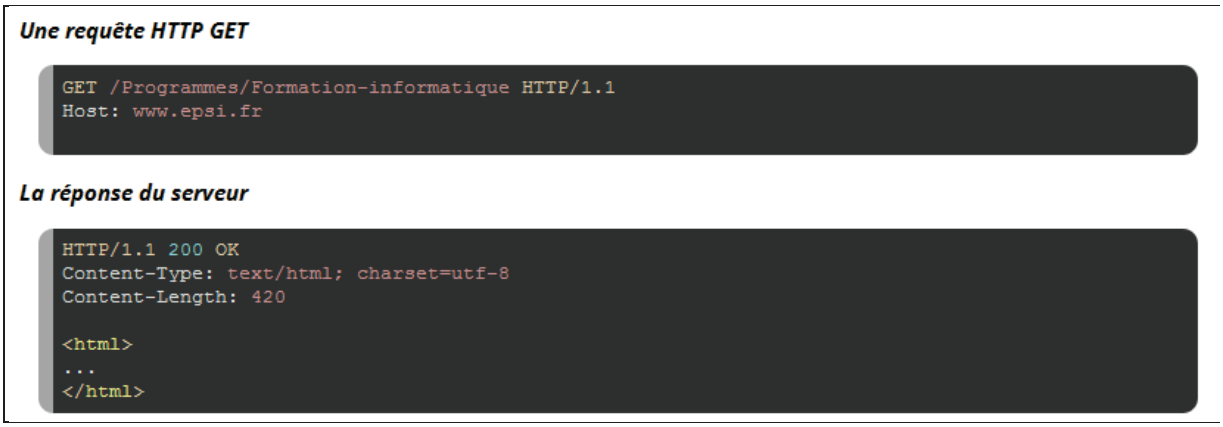


Figure 19 Exemple d'utilisation de la méthode GET [SPO]

Il y a deux propriétés importantes pour décrire les méthodes HTTP qui ressortent dans l'utilisation de REST de HTTP : la sureté et l'idempotence (RFC2616).

- ✓ Méthode **sûre** : l'état de la ressource ne doit pas être modifié. La méthode GET est sûre. Les autres (PUT, POST, DELETE) ne le sont pas puisqu'elles suppriment ou modifient la ressource.

- ✓ Méthode **idempotente** : l'état de la ressource reste identique quelque soit le nombre de fois où est envoyée la requête. Par exemple la méthode GET pour obtenir la représentation d'un livre n'affectera pas l'état de la ressource sur le serveur. Nous pouvons envoyer plusieurs fois la même requête sur le serveur nous obtiendrons toujours le même résultat. Les méthodes PUT et DELETE le sont aussi car la mise à jour sera toujours la même si la même requête est envoyée au serveur plusieurs fois d'affilé alors que la suppression de la ressource aura été effective dès la première requête, sa réémission ne modifiera plus la ressource puisqu'elle sera supprimée. Par contre une méthode POST n'est pas forcément idempotente du fait de la création d'une ressource.

A chaque requête reçue, le serveur adresse une réponse avec le code retour et la représentation de la ressource. Voici les codes les plus utilisés dans le cadre de REST.

Code retour	**Statut**
200	Succès et représentation renvoyée par le serveur
201	Ressource créée
204	Succès, pas de représentation
301	La ressource demandée a changé d'URI, la nouvelle URI peut être indiquée dans l'en-tête de réponse
400	Erreur, syntaxe erronée
410	La ressource n'existe plus

- Représentation

La représentation est le format informatique sous lequel est obtenue la ressource. Cela peut être une image, une page html, un document XML…

2.6.1.3 Exemple simple de mise en œuvre d'une interface RESTfull [JOE]

Dans [JOE], l'auteur J. Gregorio nous propose de se poser les questions suivantes pour créer un service REST : quelles sont les ressources ? Quelles sont les représentations ? Quelles sont les méthodes supportées par chaque ressource ? Quels sont les codes statut qui doivent être retournés?

- ✓ Quelles sont les ressources ?

L'auteur dresse la liste de toutes les ressources puis identifier leur URI. Son exemple s'intéresse à un annuaire d'entreprise qui contiendra le nom de chaque employé, le titre et

l'information de contact. Il identifie alors deux ressources, donc deux types d'URI : /employe et /tous_les_employes.

- ✓ Quelles sont les représentations ?

Il est ici question de la représentation choisie. Pour son interface, l'auteur choisit le XML.

Format d'employé

Pour les besoins de l'illustration —Action d'éclairer, d'illustrer par des explications, des exemples.—, je vais créer un nouveau format XML pour cette information, ce qui signifie seulement que je triche pour les besoins de la cause.

```
<employe xmlns='http://example.org/mon-exemple-ns/'>
    <nom>Le nom complet</nom>
    <titre>Titre de la personne</titre>
    <tel>Le numéro de téléphone</tel>
</employe>
```

Format de la liste des employés

Comme chaque employé aura son propre URI avec tous les détails, notre liste ne contiendra que cet URI.

```
<annuaire xmlns='http://example.org/mon-exemple-ns/'>
    <employe-ref href="URI du premier employé">
         Nom complet du premier employé</employe-ref>
    <employe-ref href="URI de l'employé #2"/>Nom complet</employe-ref>
    .
    .
    <employe-ref href="URI de l'employé #N"/>Nom complet</employe-ref>
</annuaire>
```

Figure 20 Représentation XML des ressources [JOE]

- ✓ Quelles sont les méthodes supportées pour chaque ressource ?
- ✓ Quels sont les codes de statut qui doivent être retournés?

Ressource	Méthode	Représentation	Codes de statut
Employé	GET	Format d'employé	200, 301, 410
Employé	PUT	Format d'employé	200, 301, 400, 410
Employé	DELETE	N/A	200, 204
Tous les employés	GET	Format d'annuaire	200, 301
Tous les employés	POST	Format d'employé	201, 400

Figure 21 Tableau récapitulatif des étapes[JOE]

A chaque méthode choisie est attribué un format de représentation puis le code retour attendus selon la requête envoyée au serveur.

Pour aller encore plus loin, l'auteur propose dans [JOE2] et [JOE3] de créer un service web de signets.

2.6.2 Comparaison avec d'autres styles d'architecture

Dans la littérature, nous retrouvons fréquemment les services REST comparés à ceux de SOAP.

2.6.2.1 SOAP

La Figure 22 nous illustre un exemple de fonctionnement de services web basés sur le protocole SOAP utilisé dans une architecture RPC. Il est écrit en XML et standardisé par le W3C.
Il permet l'échange de messages entre le client et le fournisseur par le biais des descriptions WSDL. Celles ci décrivent statiquement les services, leurs points d'entrée et les types de données.
Les fournisseurs publient l'existence de leurs services au moyen de ces descriptions WSDL dans des annuaires (UDDI, système d'annuaire basé sur un principe analogue à celui des pages jaunes).
Ainsi, les clients peuvent les découvrir puis les utiliser en les invoquant auprès du fournisseur.

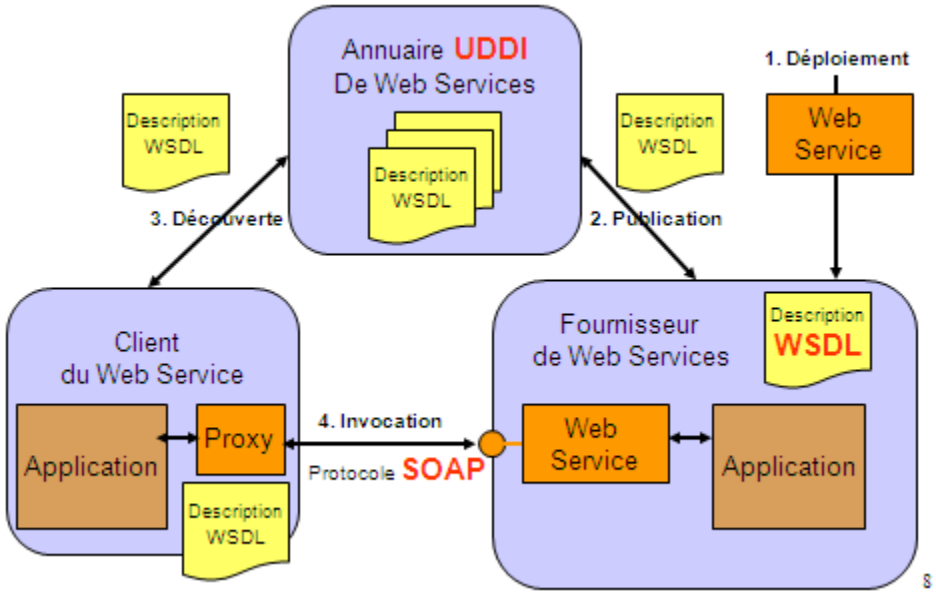


Figure 22 Fonctionnement des services web [RBLO]

2.6.2.2 REST vs SOAP

Dans la figure ci-dessous, nous pouvons constater que la requête de demande d'accès à une ressource est plus complexe dans le cadre d'une demande SOAP. REST permet l'envoi de message sans enveloppe SOAP et dans un encodage libre (XML, JSON, binaire…).

Le client qui développe une application et utilise ne architecture REST connait d'avance les méthodes qui pourront lui être proposées contrairement à celui qui utilise SOAP. « Ensuite, l'utilisation d'un nombre réduit de méthodes permet aux clients de savoir, avant même de contacter le service, quelles méthodes sont disponibles. De plus, la sémantique de ces méthodes étant définie clairement. Si le client veut récupérer une ressource, il y a de forte chance que la méthode GET fonctionne. Un client SOAP ne pourra jamais deviner le nom des méthodes avant d'avoir consulté le contrat WSDL » [CLE].

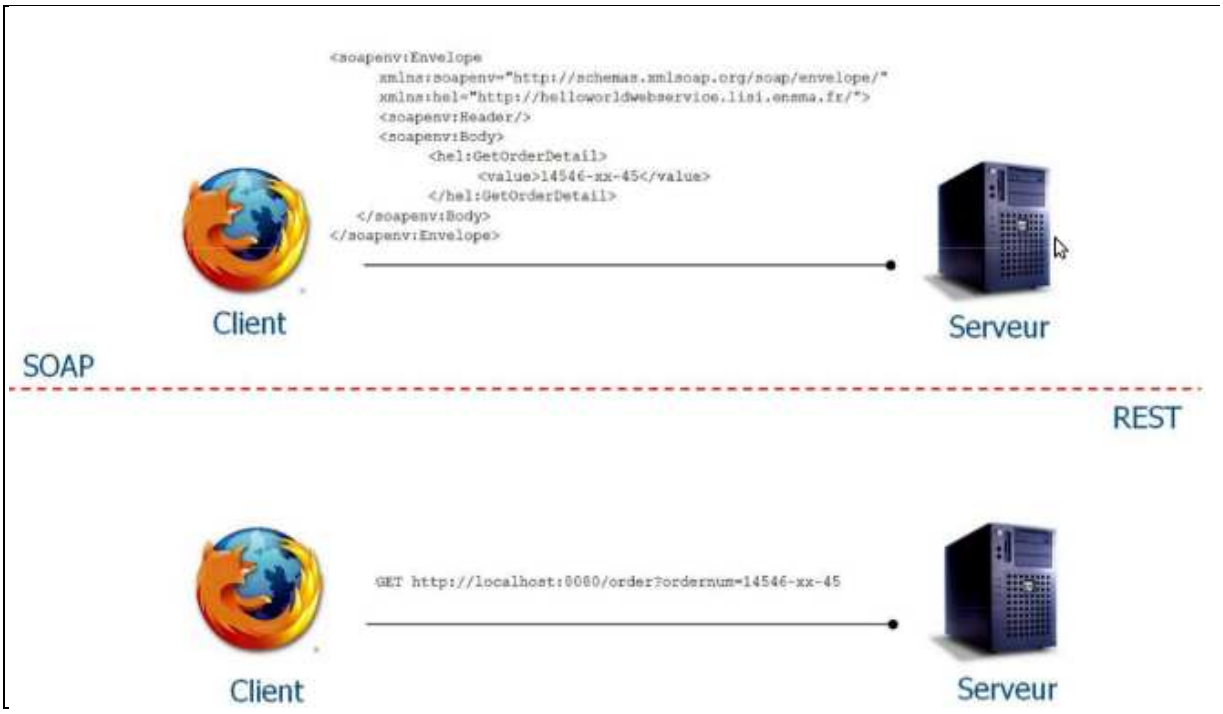


Figure 23 Message SOAP vs REST [ERI]

Les avantages accordés à l'utilisation de SOAP sont sa standardisation, l'interopérabilité et la sécurité [ERI]. En contrepartie, SOAP est complexe à mettre en place

D'un autre côté, nous avons vu précédemment la simplicité de mise en œuvre de REST. Il n'est pas standardisé mais utilise les méthodes http. L'aspect sécuritaire est en revanche limité par http.

3 Etude de l'existant

3.1 Watt, outil de workflow pour la gestion des demandes des cotisants

3.1.1 Présentation

L'application WATT (Worflow d'Assistance aux Tâches Techniques) est destinée à la dématérialisation des courriers cotisants afin d'apporter une meilleure qualité de service, meilleur temps de traitement et un suivi des demandes à traiter.

C'est un outil de workflow documentaire et procédural adaptable à chaque organisme pour la gestion des comptes cotisants. Il permet d'assister les techniciens du recouvrement dans le traitement des demandes cotisants à travers des procédures locales, régionales et nationales. La version actuelle est basée sur Eclipse RCP[8] et est accessible depuis le portail Harmonie[9]. Cette version est en production depuis 2010. Auparavant, Watt s'appuyait sur un moteur de Workflow développé sous Lotus Notes.

3.1.2 Activité sous Watt

WATT est découpé en plusieurs modules accessibles selon le profil de l'utilisateur : Watt Injection, Watt Pilot, Watt Traitement et Watt Administration.

- **Watt Injection**

Le module **Watt Injection** gère les demandes de création d'affaires[10]. La Figure 24 nous donne un aperçu du processus déroulé lors de l'injection d'une affaire. A la réception d'un document[11] (fax, lettre, e-mail,…), celui-ci peut être numérisé puis stocké dans une base GED (Gestion Electronique Documentaire) et injecté dans Watt où une nouvelle affaire est créée. Selon la nature de la demande, un type de demande (prédéterminé dans Watt) est affecté à l'affaire. A chaque type de demande correspond un circuit[12] et un modèle de distribution. La distribution peut être faite de manière automatique par le système, c'est-à-dire envoyée directement à un ou plusieurs gestionnaires, ou bien envoyée dans une corbeille dédiée à la distribution. Dans ce dernier cas, l'affaire va être traitée par une ou plusieurs personnes avant d'être affectée aux gestionnaires.

---

[8] Eclipse RCP : plateforme de développement d'application «client riche»

[9] Portail Harmonie : portail présent sur les machines des agents et proposant un accès à l'ensembles des applications du recouvrement

[10] Affaire : une affaire peut être définie comme l'ensemble des informations correspondant à un traitement à effectuer par un agent de l'organisme pour répondre à un cas soulevé par un cotisant. Une affaire équivaut donc au dossier physique constitué puis traité par le gestionnaire de comptes.

[11] Document : on désigne par document tout type de support physique (courriers,…) ou dématérialisé (fax, e-mail, appel téléphonique) qui est alors traité lors de la vie d'une affaire. On distingue leur nature en parlant de documents générateurs d'affaire à l'origine de la création d'une affaire et de documents à rattacher qui alimentent l'affaire. Chaque document possède un type permettant de l'identifier dans les processus métiers.

[12] Circuit : un circuit permet de standardiser le traitement des affaires ayant des caractéristiques communes (type de document générateur, tâches, ordonnancement des tâches). Il représente tout le cheminement possible d'un processus en fonction des tâches et de leurs états.

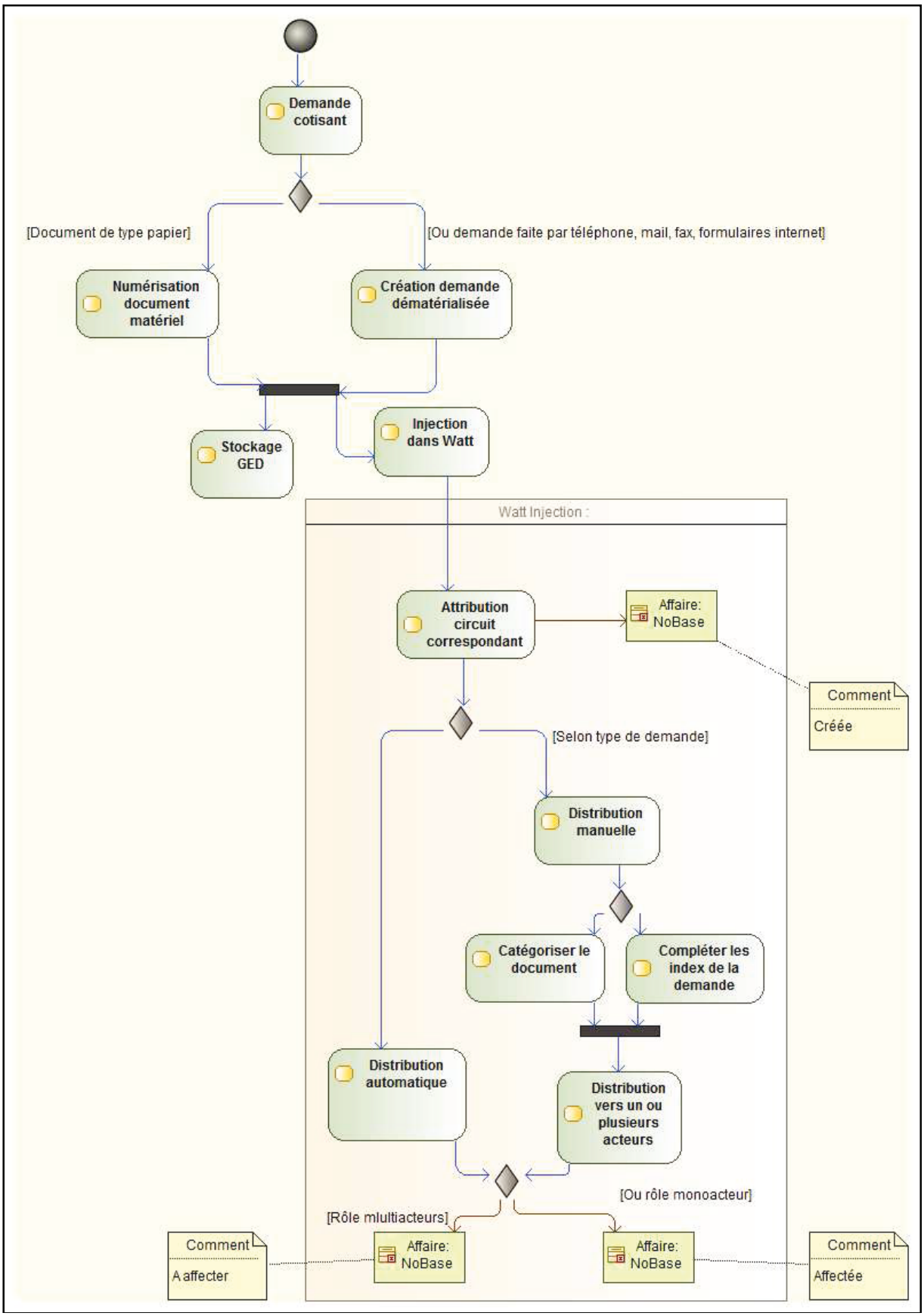


Figure 24 Injection d'une affaire

- **Watt Pilot**

Le module **Watt Pilot** permet le pilotage et le traitement des affaires. Il est utilisé par environ 6000 utilisateurs par jour. Les managers et gestionnaires utilisent ce module pour suivre les traitements d'affaires. Les managers peuvent aussi l'employer pour affecter des affaires aux gestionnaires ou même modifier le gestionnaire en charge de l'affaire. Le module met à disposition une interface de visualisation des listes d'affaires et des fonctionnalités de recherche dans les dossiers. Les managers ont aussi la possibilité de prioriser les affaires.

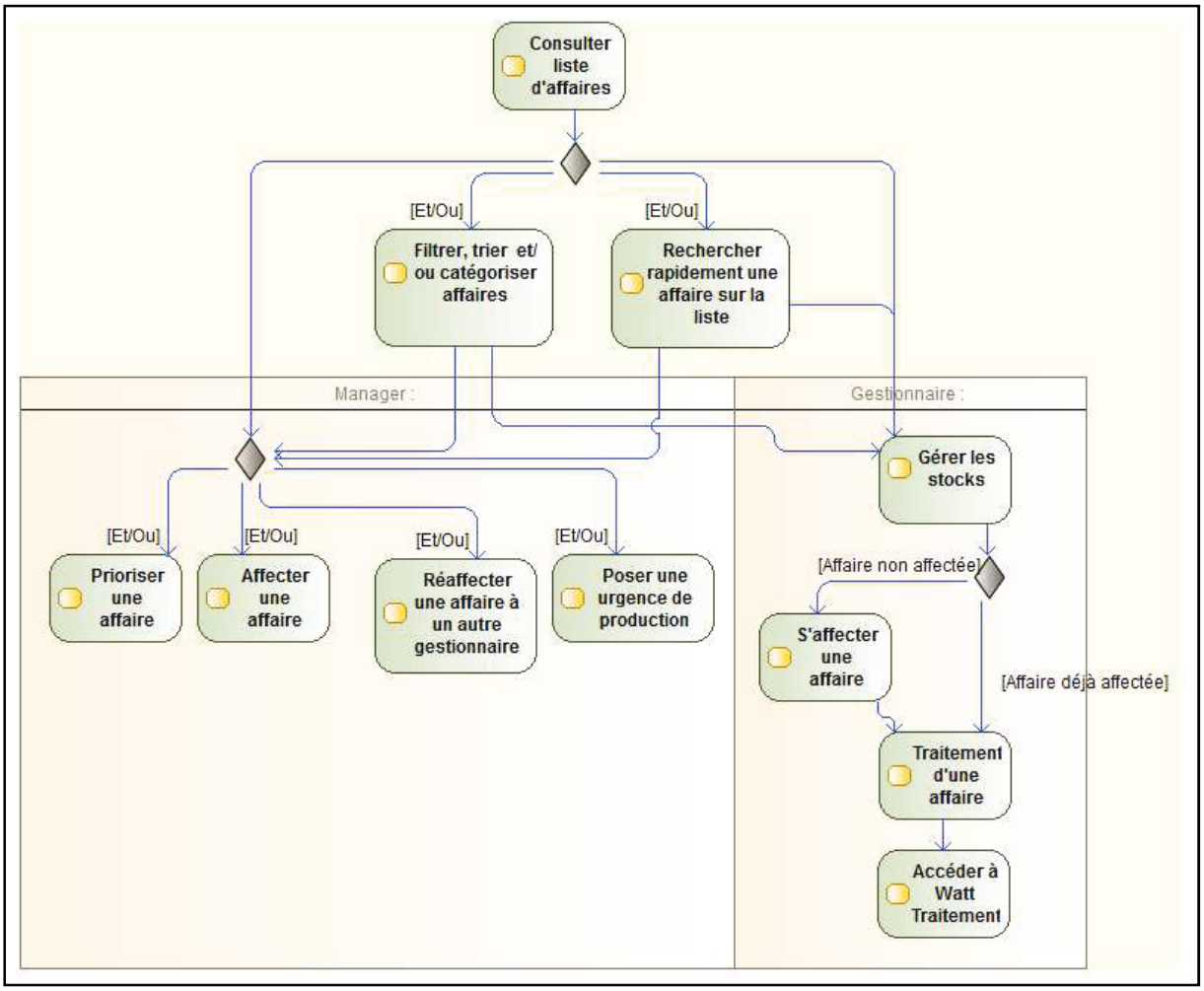


Figure 25 Fonctionnalités de Watt Pilot

- **Watt Traitement**

Les affaires à traiter par les agents sont stockées dans une corbeille (voir illustration Annexe I) qu'ils peuvent consulter depuis Watt Pilot. Les affaires peuvent être consultées et traitées au travers d'une interface, **Watt Traitement** (voir Figure 26), où sont exposées toutes les informations liées à la gestion du workflow (étapes du processus à faire ou déjà traitées) et à la gestion du compte cotisant (numéro de compte, dénomination, SIRET). Il est aussi possible de consulter l'historique de l'affaire ainsi que les éventuels commentaires laissés pour celle-ci (autres illustrations des onglets historique et commentaires en Annexe II).

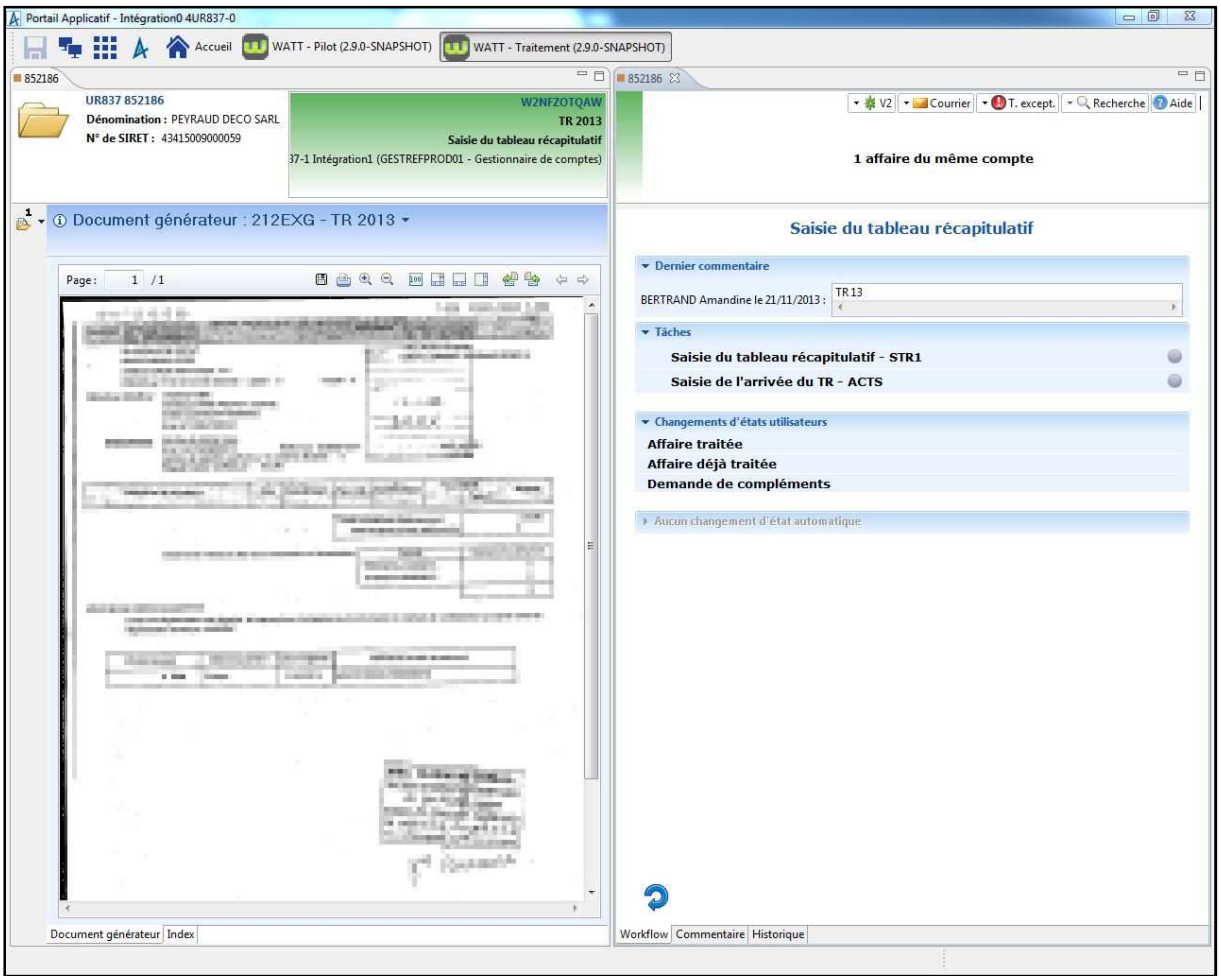


Figure 26 Interface de visualisation des informations d'une affaire

La Figure 27 expose les activités possibles depuis Watt Traitement. Ce module est accessible par les gestionnaires et les managers. Ils peuvent, selon leur besoin, effectuer des activités de consultation ou bien de traitement d'affaire au travers d'un processus. Ces actions se composent de tout un ensemble d'activités qui peuvent être effectuées distinctement ou bien les unes à la suite des autres. La consultation peut aussi être un sous ensemble dans le traitement d'une affaire pour permettre au gestionnaire la prise de connaissance de l'affaire. Cependant, si la personne qui consulte juge que l'information n'est pas suffisante pour sa compréhension, elle va alors contacter les différents acteurs qui pourraient lui apporter des éléments supplémentaires (créateur de l'affaire, autres gestionnaires intervenus en amont…). Les moyens de communication sont alors le téléphone, le mail ou la messagerie instantanée. Dans tous les cas, il faut sortir de Watt et c'est pénalisant. Beaucoup de gestionnaires n'ont pas la messagerie ouverte en parallèle. L'utilisation du téléphone peut aussi être un inconvénient si le numéro à joindre n'est pas connu, il faudra alors le chercher au moyen d'un annuaire.

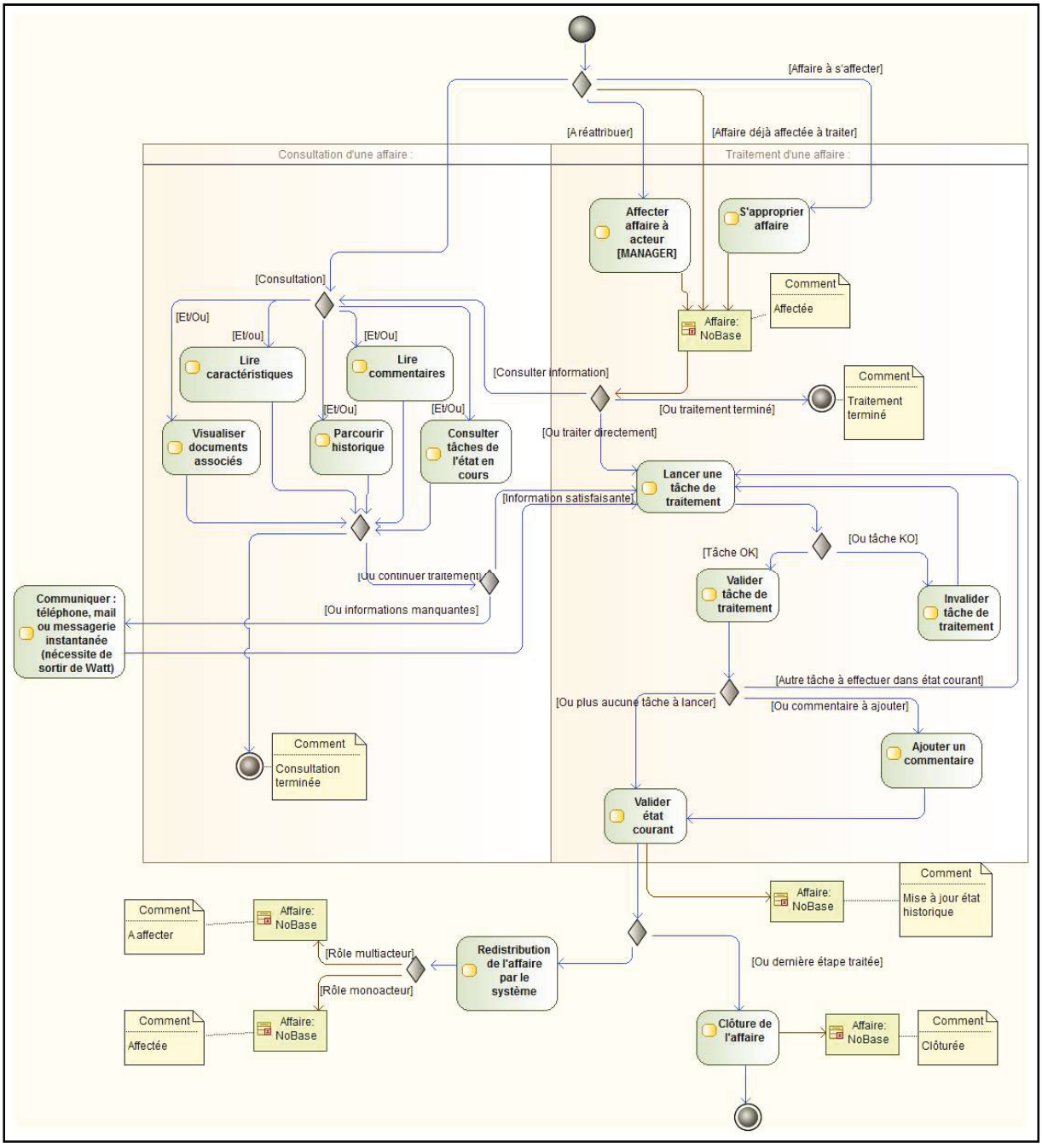


Figure 27 Activités dans Watt traitement

Les gestionnaires peuvent consommer des services du système d'information à travers les activités du processus comme par exemple interroger le SNV2[13]. Les circuits du workflow sont composés d'une ou plusieurs étapes de traitement. Chacune d'entre elles définit l'action à effectuer sur l'affaire. Sur la Figure 27, il est possible de retracer le cheminement des activités à réaliser au sein d'un état du processus de traitement d'une affaire : affectation de celle-ci, puis consultation d'informations, recherche si besoin de renseignements supplémentaires, ensuite lancement d'une ou plusieurs tâches et enfin réaffectation ou clôture de l'affaire.

- **Watt administration**

[13] SNV2 : système central d'information des URSSAF

Toutes les URSSAF utilisent le même logiciel mais chaque organisme dispose de sa propre organisation, le pilotage est géré au niveau régional (voir chapitre 4.1). **Watt administration** va permettre aux administrateurs de procéder au paramétrage fonctionnel et technique de l'application. Par exemple, dès qu'un processus de traitement d'un type de demande doit être créé ou modifié, l'administrateur va recevoir une demande de création ou de modification de circuit. Il va alors utiliser le module pour le modéliser et paramétrer.

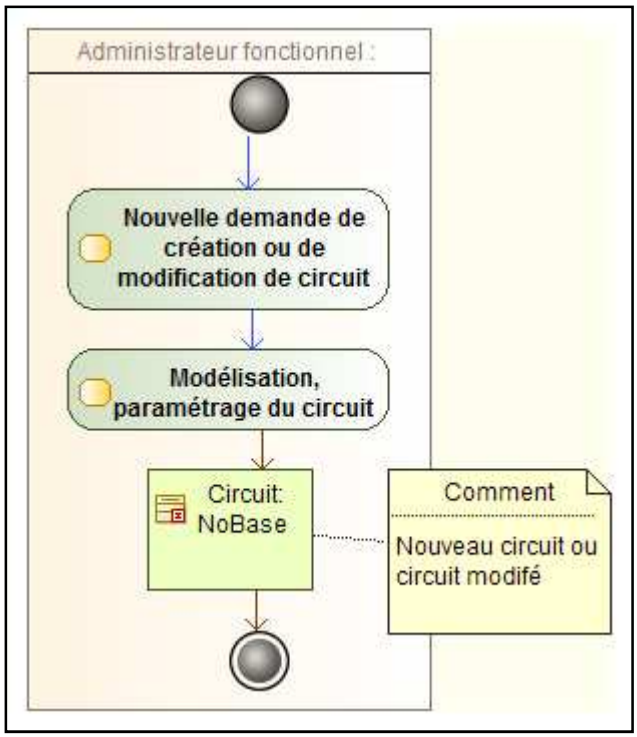


Figure 28 Paramétrage des circuits

3.1.3 Evolution de l'activité

Comme nous le verrons ensuite avec l'explicitation de la problématique, le CIRSO souhaite favoriser la collaboration entre les différents utilisateurs du workflow. Pour se faire, il souhaite coupler l'outil actuel de messagerie instantanée Sametime, qui est riche en fonctionnalités de collaboration (cf. 3.2.1.2), à l'outil de workflow Watt.

L'ajout et l'utilisation des fonctionnalités collaboratives a un impact potentiel dès la distribution d'une affaire (cf. Figure 24). En effet, la distribution peut se faire sur site distant et donc nécessiter une collaboration entre plusieurs acteurs séparés physiquement.

L'évolution apportée par ce couplage se focalise principalement au niveau de l'activité de traitement d'une affaire (cf. Figure 30). Le but du couplage étant d'apporter des facilités d'interaction tout au long du traitement d'une affaire.

L'activité de communication lors de la recherche d'information concernant une affaire évolue aussi. Les moyens de communication sont intégrés directement à l'application Watt. L'utilisateur n'a donc plus à jongler entre les diverses applications pour interagir avec d'autres acteurs.

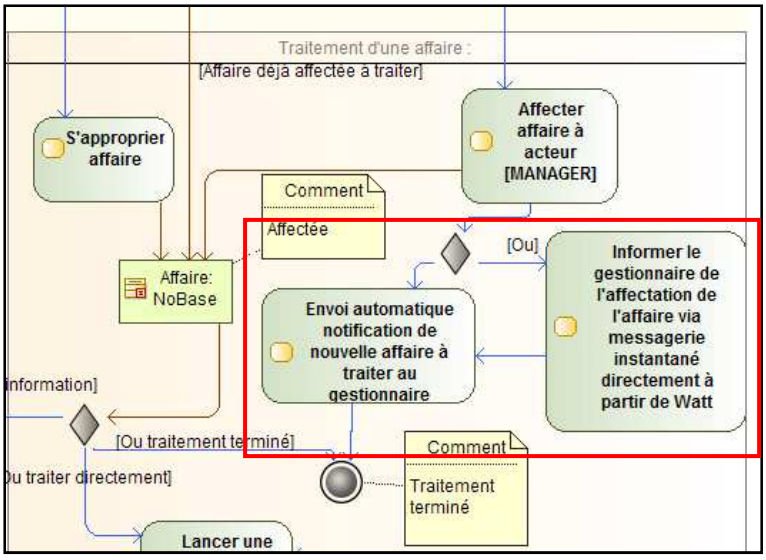


Figure 29 Nouvelles activités lors de l'affectation d'une affaire par un manager

Par exemple, si un manager affecte une affaire à un gestionnaire, il peut l'en informer immédiatement par messagerie instantanée (cf. Figure 29). L'accès à la fenêtre de communication se fait dès lors sans sortir de l'outil de workflow grâce aux nouvelles fonctionnalités implantées. D'autre part, une notification est générée automatiquement pour prévenir le gestionnaire de l'arrivée de cette nouvelle affaire dans sa corbeille.

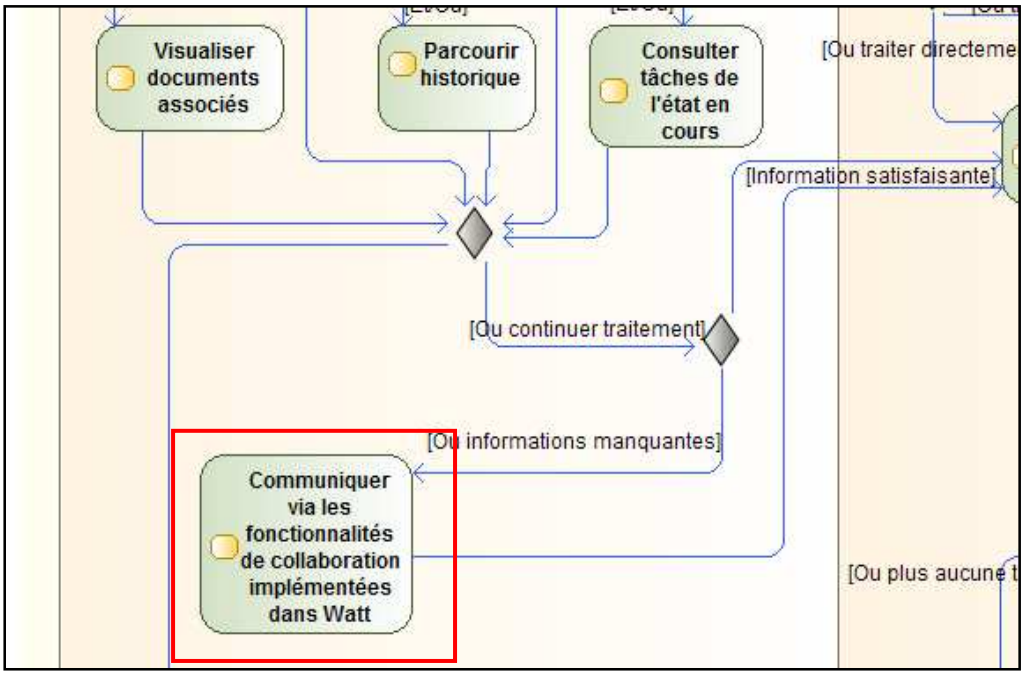


Figure 30 Evolution de l'activité de communication

Cette évolution impacte l'activité de traitement d'une affaire tout aussi bien que la consultation d'une affaire puisque les fonctionnalités sont intégrées à Watt Traitement (cf. Figure 30).

Ainsi, si un cotisant souhaite connaitre l'état d'avancement de sa demande, le conseiller en ligne qui ne trouve pas les informations adéquates sur Watt Traitement, peut contacter un des acteurs intervenus sur l'affaire par messagerie instantanée. Le nom des acteurs apparaît clairement sur l'interface (par exemple via l'onglet historique). Le conseiller n'a donc plus qu'à cliquer sur le nom de l'acteur pour obtenir des fonctionnalités de communication. Son activité se voit donc simplifiée car il n'aura plus besoin de changer d'application mais aussi de chercher le nom de l'acteur dans l'annuaire de la messagerie pour pouvoir le contacter.

- Notifications

Comme nous venons de le voir avec l'évolution de l'activité de traitement d'une affaire, les notifications constituent une nouvelle action déclenchée par un certains nombres d'actions dans Watt. Ainsi comme dans Watt traitement, une notification sera générée automatiquement depuis Watt Injection lorsqu'une affaire sera affectée suite à sa création dans Watt puis lors des étapes de distributions manuelles ou automatiques. Le ou les gestionnaires concernés recevront alors une notification les informant de l'arrivée de cette nouvelle affaire dans leur corbeille.

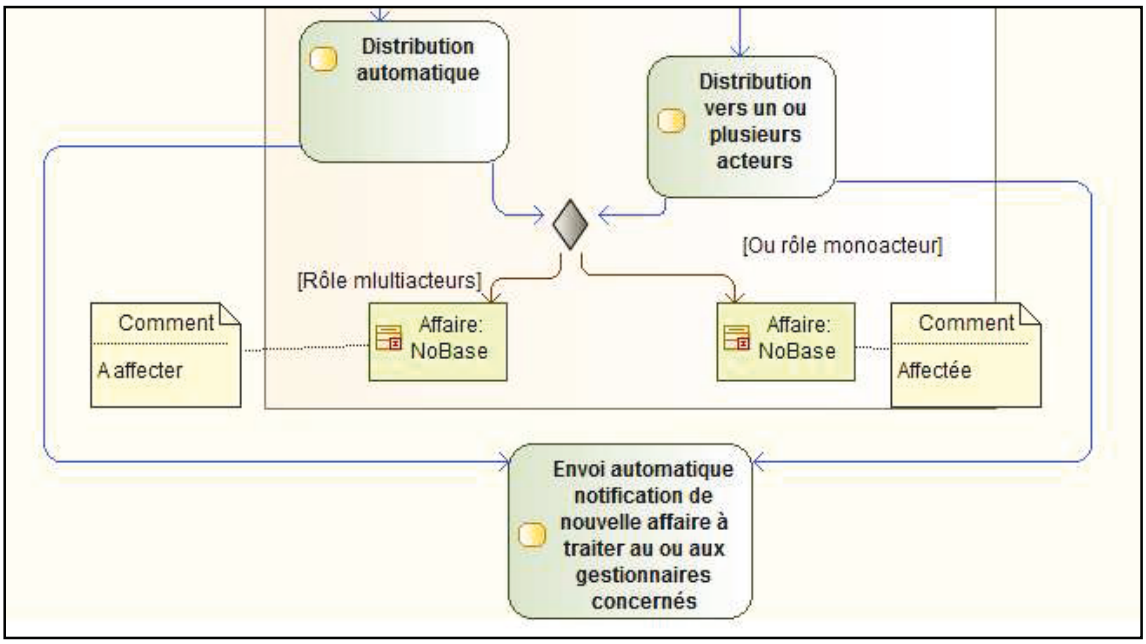


Figure 31 Evolution de l'activité dans Watt Injection

Dans Watt Pilot, différentes actions par le manager sur une affaire peuvent déclencher une notification. Ainsi, lorsque celui-ci va modifier la priorité d'une affaire ou bien lui attribuer une urgence de production toute comme lorsqu'il va l'affecter ou la réaffecter à un nouveau gestionnaire, une notification sera générée automatiquement et reçue par le gestionnaire concerné.

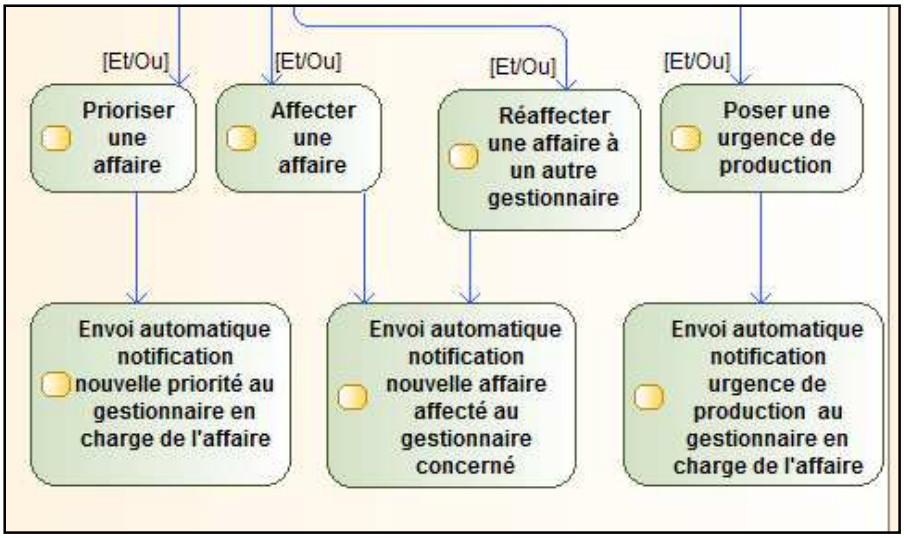


Figure 32 Evolution de l'activité dans Watt Pilot

Les notifications peuvent aussi être générées manuellement par un administrateur fonctionnel. Lorsque celui-ci intervient sur le paramétrage d'un circuit ou toute autre action sur l'application, il a la possibilité d'alerter immédiatement les utilisateurs.

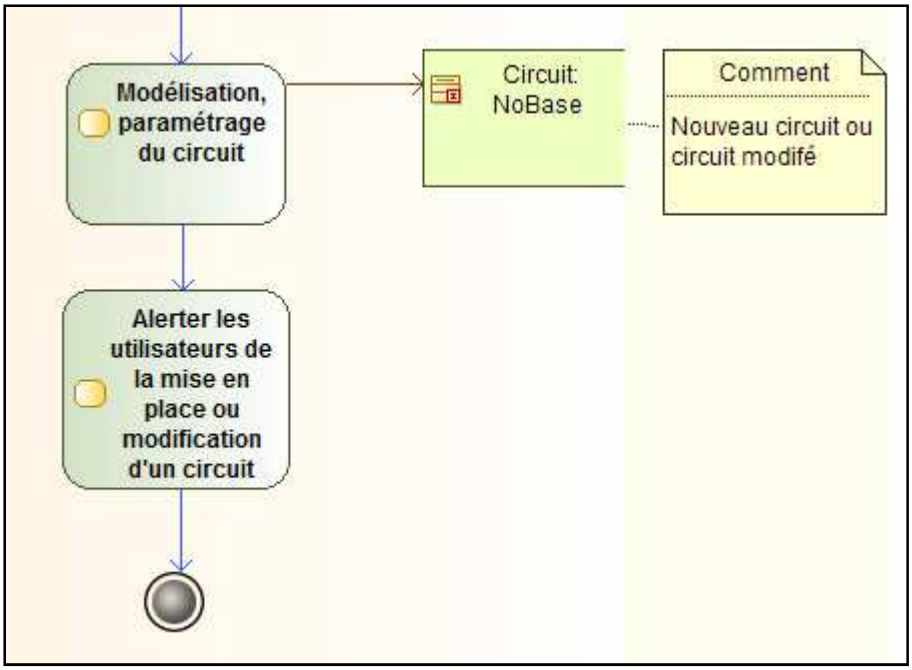


Figure 33 Evolution de l'activité d'un administrateur fonctionnel

Toutes les notifications reçues par un utilisateur sont stockées dans un historique type fil d'activité. Il pourra alors les visualiser pendant quelques jours. D'autres options ergonomiques, lui permettent aussi de savoir le nombre de notifications, ainsi que de visualiser les notifications non lues.

3.1.4 Architecture actuelle

L'infrastructure des plateformes de production des applications aux URSSAF est fournie par l'équipe Hawaï et est identique pour chaque application métier.

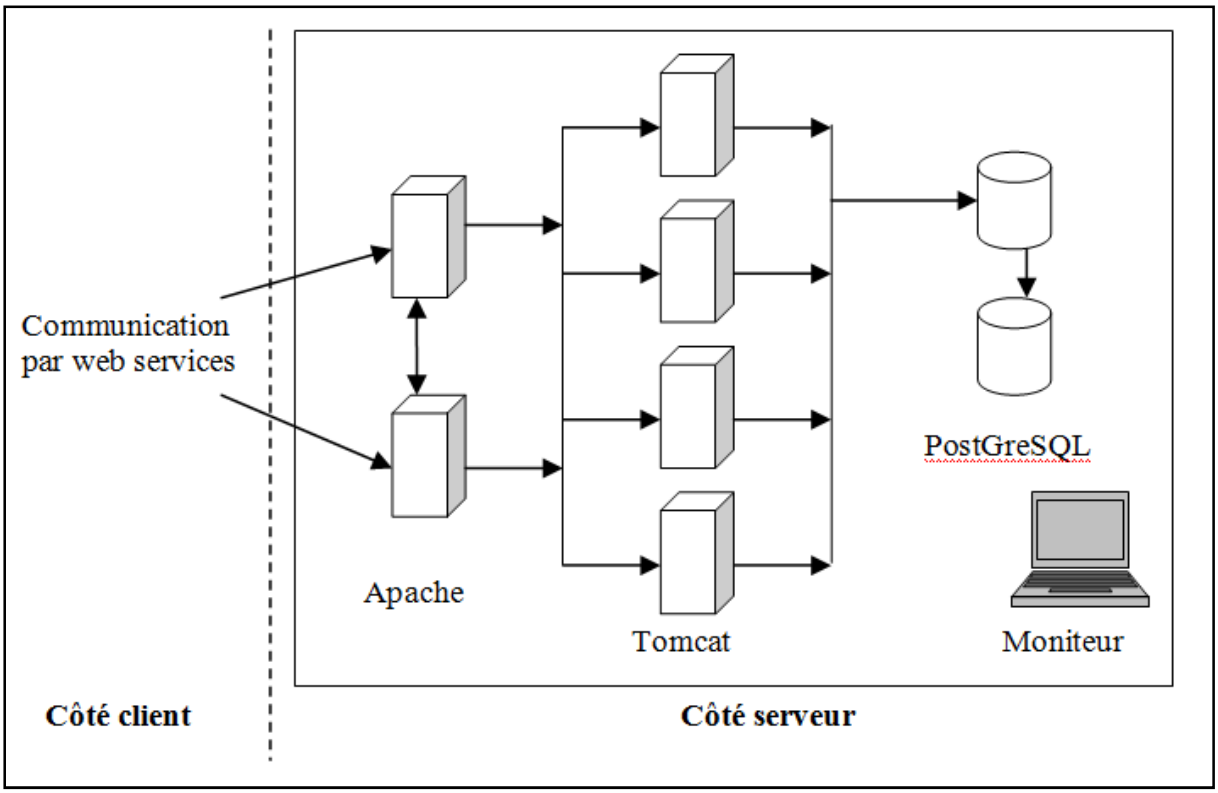


Figure 34 Architecture actuelle

Chaque plateforme fonctionne sur un modèle client/serveur. Le client envoie des requêtes côté serveur au moyen de services web.

Côté serveur on retrouve deux serveurs web Apache. Ils fournissent les pages statiques html et arbitrent en fonction de règles prédéfinies vers quel serveur applicatif (TOMCAT) ils renvoient la requête. Pour Watt la répartition se fait par ordre d'arrivée, premier vers serveur 1 puis second vers serveur2… le cinquième est envoyé au serveur 1, sixième au serveur 2 …

Pour chaque client, Apache mémorise un identifiant de session (jsessionid) et le serveur d'application vers lequel il l'a dirigé pour le rediriger vers celui-ci à chaque nouvelle requête.

Le stockage des données est géré par deux serveurs de données PostGreSQL. Un des deux serveurs sert à prendre le relais en cas de plantage du premier. Les deux serveurs sont donc identiques en termes de contenu de données.

3.2 Sametime, application de messagerie instantanée

3.2.1 Présentation

Sametime est un outil de messagerie instantanée. Cet outil permet de communiquer instantanément avec chaque agent quelque soit le lieu où il se trouve, à la condition que celui-

ci soit connecté. Chaque agent peut accéder à Sametime sur son poste de travail via le client Lotus Notes.

Sametime offre une panoplie de fonctionnalités comme le choix du statut, la possibilité de gérer ses listes de contacts…

3.2.1.1 Lotus notes

Chaque agent possède sur son poste un client Lotus Notes. C'est un outil collaboratif intégrant des fonctions de messagerie, agenda partagé et gestion des tâches mais permettant aussi d'accéder à une plate-forme d'applications.

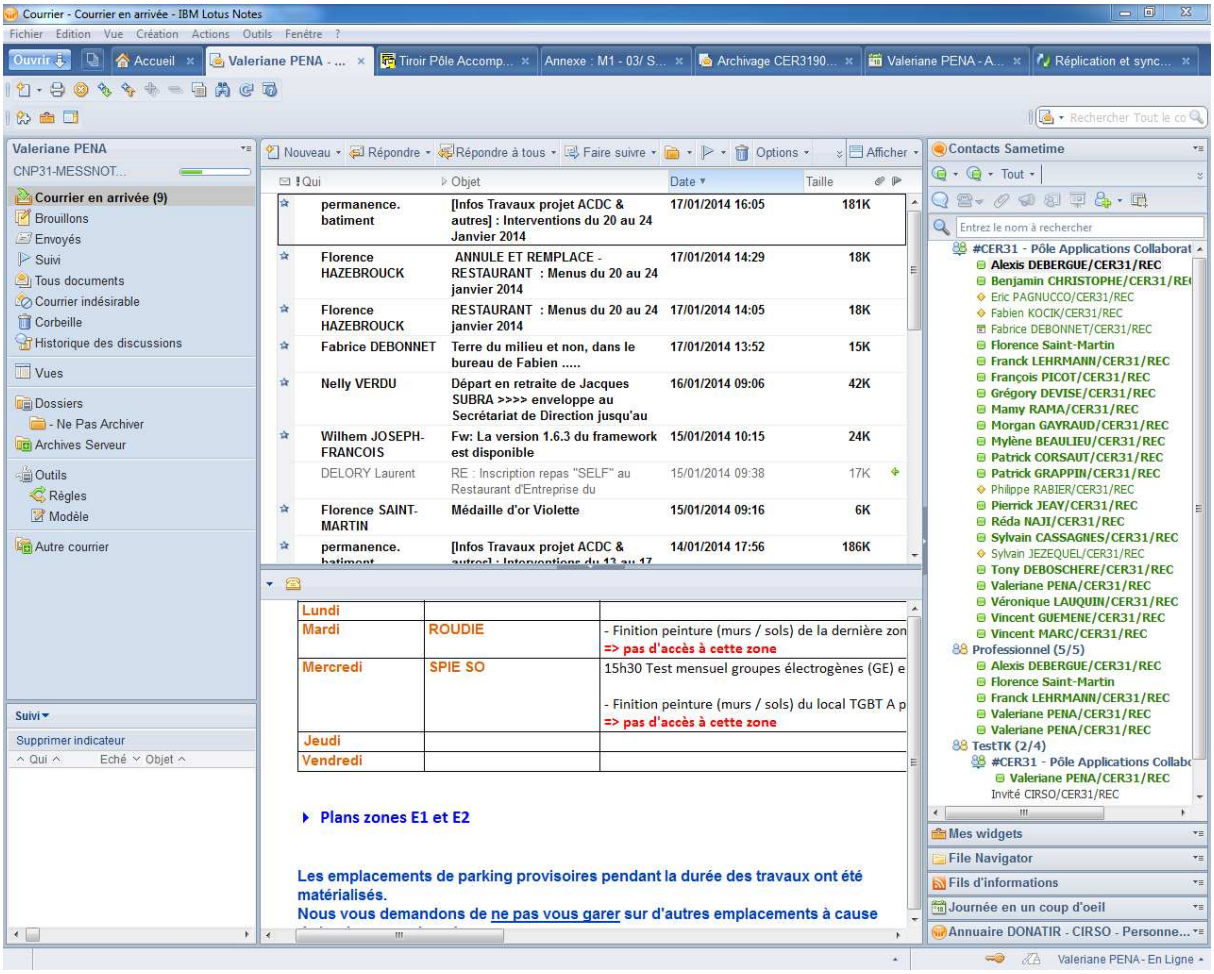


Figure 35 Client Lotus Notes

3.2.1.2 Fonctionnalités

Sametime permet une communication instantanée entre un ou plusieurs agents et propose d'autres fonctionnalités :

- possibilité de visualiser la disponibilité des autres utilisateurs en ligne (appelé « awareness » en anglais, ou présentiel au CIRSO)

- appel vidéo, réunion instantanée avec partage d'écrans entre agents

- envoi d'alerte

- gestion des listes de contact (création de groupe, ajout, suppression de contacts, recherche de contact)

- affichage de la carte de visite d'un contact

- visualisation de l'historique de discussion avec un contact

3.2.2 Architecture

Un serveur Sametime standard inclut un cluster de serveurs pour la messagerie instantanée sur une plate-forme Domino[14] et d'autres serveurs mis en cluster exécutés sous WebSphere Application Server[15] prenant en charge les réunions, les services audiovisuels et les connexions à divers clients.

L'illustration ci-dessous présente les différents types de serveurs pouvant constituer un déploiement Sametime.

[14] Domino est un serveur d'applications pour les clients Lotus Notes mais peut aussi être accessible *via* un client web. Domino est un serveur de base de documents.(wikipédia)

[15] Websphere Aplication Server : est une plate-forme applicative générique couvrant un ensemble de solutions développées par IBM qui permettent de développer, de déployer et d'utiliser des applications d'entreprise, même dans des cas complexes faisant appel à des applications et des matériels hétérogènes (wikipédia)

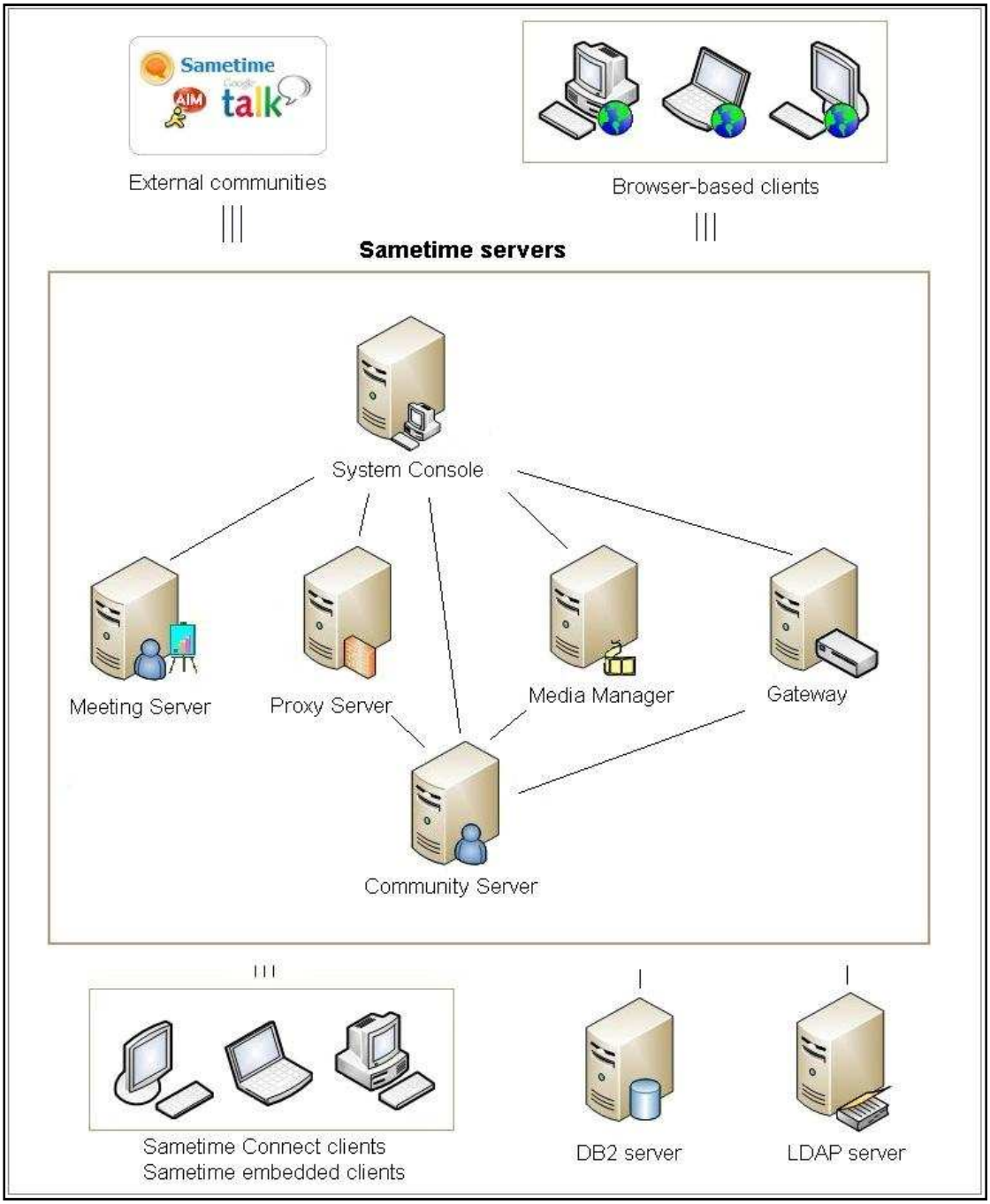


Figure 36 Architecture Sametime[16]

Dans le cadre de l'implémentation des fonctionnalités de collaboration, nous parlerons surtout du serveur de communauté[17] et du serveur proxy (community server et proxy server sur le schéma).

[16] Source : http://infolib.lotus.com/resources/sametime/8.5.2/doc/stifracd001/fr_fr/overview/images/ST_Std_852_deploy_ov.jpg

[17] Communauté (source : site IBM) La communauté Sametime fait référence à tous les utilisateurs disposant d'un accès via un navigateur Web à un serveur Sametime (ou plusieurs) et à tous les serveurs Sametime prenant en charge ces utilisateurs. La communauté Sametime peut être gérée dans l'annuaire Domino Directory sur le serveur Sametime ou dans un annuaire LDAP sur un serveur compatible LDAP tiers. Plus précisément, la communauté Sametime peut être décrite comme un annuaire partagé, ou un ensemble d'annuaires, qui répertorie

- Serveur de communauté

Il prend en charge toutes les activités de présence (présentiel ou awareness) et de discussion de type texte dans une communauté Sametime. N'importe quel client Sametime utilisant la fonctionnalité de présentiel doit s'y connecter.

Il prend en charge la gestion des demandes de connexion client, l'accès à l'annuaire, la diffusion des données de présence, la maintenance et le stockage des informations de confidentialité, des paramètres de préférence utilisateur et des listes de présence des utilisateurs en ligne.

Ce serveur est configuré en production au recouvrement pour une version Sametime 7.0. Un autre serveur est en cours d'intégration sur une version 8.5.2. Son passage en production n'est pas encore planifié car le budget n'a pas encore été accordé.

- Serveur proxy

Le serveur proxy communique avec le serveur de communauté. Il héberge le client Sametime pour les navigateurs. Ce serveur a été déployé tout récemment au CIRSO pour permettre le travail en cours sur l'intégration des fonctionnalités collaboratives dans Watt et anticiper les changements d'infrastructures imposés par la migration future de la version de Sametime. Par contre il a été déployé pour communiquer avec le serveur de communauté pour la version 8.5.2 IFR, son déploiement en production se fera en même temps que ce dernier.

4 Problématique

4.1 La régionalisation

La branche recouvrement désire maintenir un niveau accru de qualité de service, de performance et de maîtrise des risques. L'atteinte de ces objectifs est mise en difficulté par les évolutions permanentes de la réglementation et les disparités de processus de traitement constatées d'un site Urssaf à un autre.

Dans le but de contrer ces difficultés, l'Acoss (cf. paragraphe 1.1) a initié le projet de création des Urssaf régionales en 2010. Les trois premières Urssaf régionales sont nées le 1er janvier 2012 en Auvergne, Midi-Pyrénées et Pays de la Loire. Au fil des mois, toutes les autres régions se sont engagées dans la préparation de leur réorganisation. Ainsi, 12 autres Urssaf régionales ont vu le jour en janvier 2013 et les 6 dernières ont été créées le 1er janvier 2014.

Cette nouvelle organisation cible une amélioration du service rendu aux cotisants par la garantie d'un service homogène du réseau et le renforcement des expertises dans chaque région.

---

les personnes et groupes de la communauté en tant qu'un ou plusieurs serveurs Sametime disposant chacun d'un accès à l'annuaire ou à l'ensemble d'annuaires partagés.

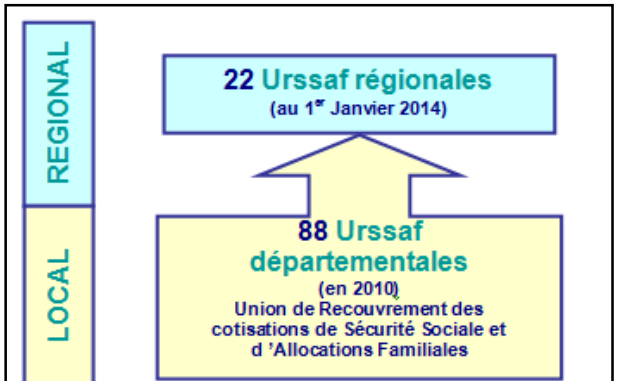


Figure 37 Création des URSSAF régionales

La création des Urssaf régionales ne modifie en rien l'ancrage territorial du réseau. La structuration de la nouvelle organisation s'appuie sur un modèle original, qui associe deux niveaux : régional et départemental. Le siège régional est en charge du pilotage stratégique. Les sites départementaux s'occupent du management opérationnel. Ils continuent d'assurer leurs activités de production. [RAACO2012]

Les Urssaf fonctionnent ainsi désormais via un siège régional et n sites distants, ce qui induit de nouveaux besoins de communication entre sites et siège-sites, dans une démarche d'harmonisation des pratiques et des procédures. A ces fins, le CIRSO souhaite mettre en œuvre des solutions favorisant la collaboration entre les différents utilisateurs du workflow.

Ici se dessine une problématique d'ordre organisationnel. La communication informelle qui se créée lorsque les personnes travaillent dans un même espace apportent beaucoup dans la réussite d'un projet sans que l'on ne s'en rende forcément compte. Celle-ci se fait dans les couloirs, à la pause café… La collaboration à distance entraine alors une perte des interactions que les équipes utilisent lorsqu'elles travaillent ensembles.

Pour solutionner ce problème, la mise en place de fonctionnalités collaboratives dans le workflow permettrait de créer une nouvelle forme de communication « informelle ».

Les URSSAF ont déjà un outil de messagerie instantanée (Sametime) déployé. Il a donc été souhaité de l'utiliser pour intégrer des possibilités de messagerie instantanée directement dans le workflow. Le but de l'intégration est de rendre transparentes les transitions entre les différents logiciels qui nous intéressent : Watt et Sametime.

La démarche méthodologique du projet est celle de rigueur au CIRSO, la méthodologie CAIRN. Elle est explicitée au chapitre 5.1.

4.2 Implémentation des fonctionnalités collaboratives de Sametime

4.2.1 Outils IBM

IBM met à disposition un kit SDK (software development toolkit) proposant un ensemble d'outils (ou toolkits) pour permettre de réutiliser les fonctionnalités temps réel offertes par Sametime, comme la messagerie instantanée, dans les applications métiers.

Une partie des outils est destinée au client Sametime ou aux applications web, l'autre partie s'adresse à des applications travaillant avec le serveur Sametime.

Le SDK se décline selon le schéma de la Figure 38. Les toolkits sont classés selon l'environnement auquel ils s'adressent dans les répertoires client, server et telephony.

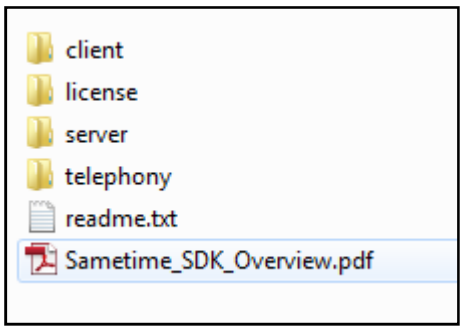


Figure 38 Arborescence du SDK

Chaque toolkit contient la documentation qui permet de s'en servir, ainsi que les ressources proposées et des exemples. Ils fournissent chacun différentes fonctionnalités et s'adressent à des environnements différents (bureau, navigateur web, serveur). Le choix d'un toolkit se base donc sur les besoins pour notre application et l'endroit où elle doit tourner. Il est possible d'en utiliser plusieurs à la fois. Le document Sametime_SDK_Overview.pdf propose un tableau comparatif des différents toolkits (cf. figure ci-dessous).

Dans le cadre de l'intégration des fonctionnalités collaboratives dans Watt, nous n'utiliserons que des toolkits clients.

Toolkit	Use this toolkit to:	Target environments
Sametime Connect Toolkit	Build Eclipse plug-ins to integrate with or extend the Sametime Connect client.	desktop
Sametime Browser IM Toolkit	Add Sametime features to Web pages using Javascript and HTML.	browser
Sametime Java Toolkit	Add Sametime features to Java applications.	desktop, server
Sametime Helper Toolkit	Invoke features in the Sametime Connect client from custom Microsoft Windows applications.	desktop (Microsoft Windows only)
Sametime Connect Web API Toolkit	Invoke Sametime features (livename presence, chat, etc.) in the Sametime Connect client from web pages.	desktop
Sametime Gateway Toolkit	Build plug-ins and event consumers to extend policy compliance and logging requirements between a local Sametime community and one or many external communities.	server
Community Server Toolkit	Build Java components that add or extend services on the Sametime server.	server
Directory and Database Access Toolkit	Build C++ or Java components for the Sametime server that provide directory integration, chat logging, or virus scanning services.	server
Sametime Monitoring and Statistics Toolkit	Access Sametime server statistics in XML format via HTTP.	desktop, server
Meeting Toolkit	Schedule and manage online meetings via HTTP. Integrate third party tools with MRC	desktop, server
TCSPI Toolkit	Provide click-to-call telephony services for Sametime Connect, Sametime Web conferencing, and Lotus Notes.	server

Figure 39 Détermination des toolkits à utiliser

4.2.2 Interactions entre toolkits, clients et serveurs

Les toolkits pouvant répondre aux exigences du projet que nous avons identifiés sont : toolkit java, Connect Web API toolkit et Sametime Browser IM toolkit. Nous allons voir comment nous pouvons les utiliser selon le client utilisé et le serveur à solliciter (serveur de communauté ou serveur proxy).

- Serveur de communauté et utilisation du toolkit java

Le serveur de communauté Sametime prend en charge les activités de présence et de chat. Il gère aussi les demandes de connexion. (cf. paragraphe 3.2.2)

L'appel aux services du serveur de communauté depuis les autres serveurs ou clients ne peut se faire que par le biais de la couche offerte par le toolkit Java. Il sera donc possible d'utiliser ces services depuis Watt en utilisant ce toolkit.

- Appel au client Sametime intégré sur les postes depuis un navigateur web avec le toolkit Connect WebAPI

Le toolkit Connect WebAPI Toolkit permet de piloter le client Sametime à partir d'une page web. Il propose plusieurs fonctionnalités comme lancer les fenêtres de chat, récupérer le statut d'une personne, …

Cependant toutes les API du serveur de communauté ne sont pas disponibles (comme la récupération de la liste des contacts).

Le client démarre un serveur web local qui publie des services REST. La communication depuis le navigateur web se fait donc via le toolkit Connect Web API toolkit vers les API REST du client.

- Un peu d'histoire : l'applet STLink

Jusqu'à Sametime 7, le toolkit STLink était le moyen privilégié d'accéder à Sametime depuis un navigateur web. Il s'agissait d'une applet utilisée par le toolkit java pour échanger des messages avec le serveur de communauté.

Ce toolkit présentait néanmoins beaucoup d'inconvénients. Il requérait une machine virtuelle java sur le poste et proposait une interface swing démodée.

L'applet est maintenant obsolète mais a été remplacé par Sametime proxy server et browser IM Toolkit (voir paragraphe suivant)

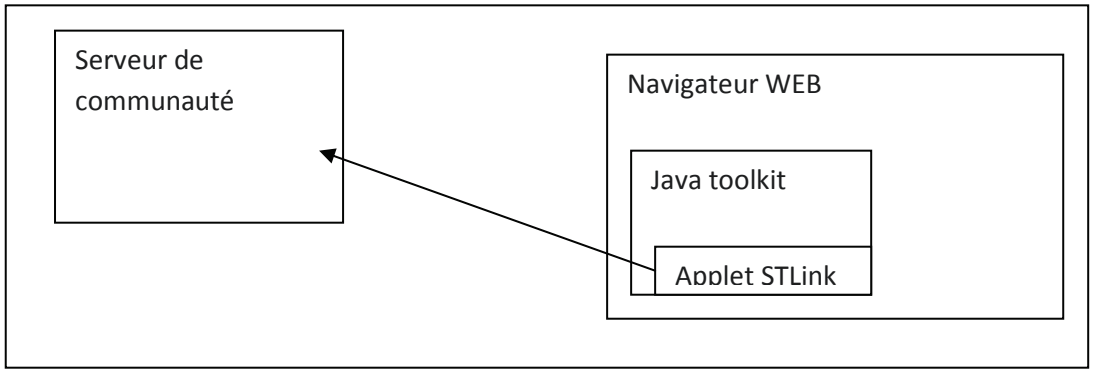


Figure 40 Architecture du STLink

- Sametime proxy server et browser IM Toolkit

Le serveur proxy a été créé pour permettre une utilisation optimale des fonctionnalités Sametime depuis un navigateur web. Il remplace donc le STLink. Son seul moyen pour accéder au serveur de communauté est d'utiliser le toolkit java. Le serveur proxy publie ses services au moyen d'API REST. La communication entre le serveur proxy et un navigateur web se fera par protocole http.

Les API REST publient 100% des API du toolkit Java.

Le browser IM Toolkit est en lien avec le Connect WebAPI toolkit : s'il détecte que le client Sametime est connecté et qu'il est autorisé à l'utiliser (paramètre à définir lors de l'utilisation du toolkit), il utilisera le client sinon il passera par les API REST du proxy. Par exemple, pour lancer une nouvelle fenêtre de chat, s'il détecte le client Sametime, il va lui demander d'ouvrir

sa propre fenêtre, sinon il ouvrira un nouveau navigateur dans le quel il chargera l'interface web.

- Synthèse des interactions

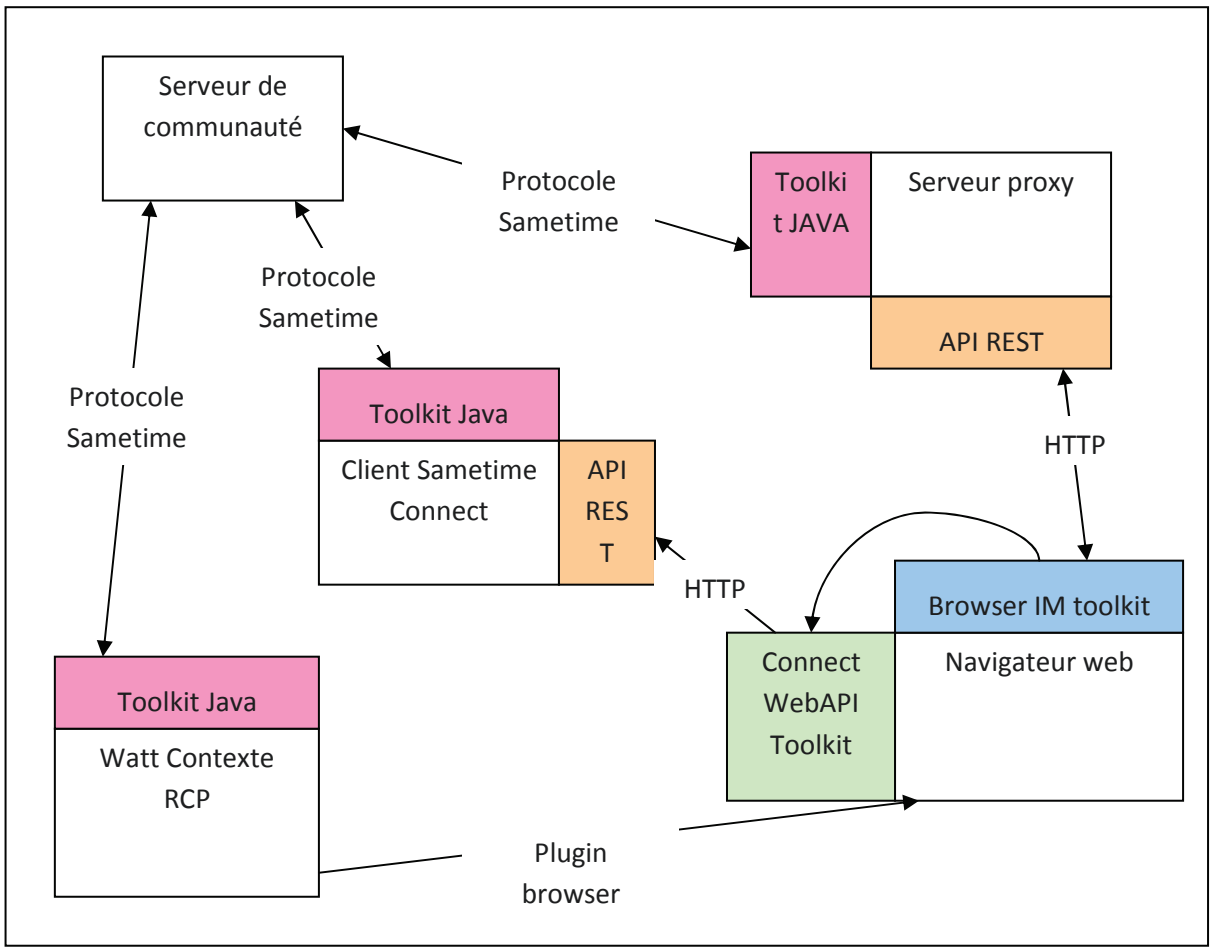


Figure 41 Schéma des interactions entre serveurs, environnements et toolkits

5 Démarche de projet

5.1 Méthodologie CAIRN

5.1.1 Présentation

La méthodologie CAIRN[18] est la méthodologie mise en place par ALTAIR[19] (voir annexe IV) pour détailler les activités à mener dans chaque phase du cycle de vie d'un projet.

Cette méthode est basée sur les méthodes de type "Processus Unifié" (RUP, TTUP,…), sur quelques pratiques Agiles, et incluant les principes d'une architecture SOA. Cette

[18] CAIRN : Conception, Analyse, Implémentation pour la RéNovation
[19] ALTAIR : Atelier Logiciel Transverse Associé à l'Informatique du Recouvrement

méthode a été adaptée aux besoins des projets du recouvrement, et continue d'être adaptée grâce aux remontées des utilisateurs.

Les principes fondateurs :
- Elle est adaptée à un processus de développement itératif et incrémental ou classique (cycle en V)
- Elle structure la réalisation par les cas d'utilisation
- Elle prône le pilotage des projets et des programmes par les risques
- Elle cadence la réalisation des projets par des ateliers collaboratifs
- Elle est basée sur une démarche d'architecture d'entreprise, intégrant SOA, structurant et alignant le SI au métier
[CAIRN][20]

La méthodologie CAIRN a été détaillée et mise à disposition des agents du recouvrement sur un wiki dédié.

Plusieurs outils de gestion de projet appuient cette démarche : Reqtify pour la gestion des exigences, Redmine pour la gestion des anomalies, PSNext pour la planification, Enterprise Architect pour la modélisation applicative, ALM pour l'intégration et la validation (ALM)…Un tableau récapitule les outils en fonction des processus dans l'annexe IV.

5.1.2 Cycle de vie projet

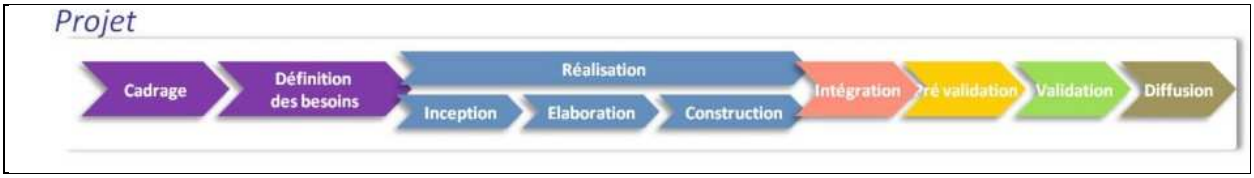


Figure 42 Phases du projet[21]

Le projet est découpé en plusieurs phases (cf. Figure 42). Chacune de ces phases est décrite brièvement dans le tableau qui suit.

Cadrage		Analyse de l'opportunité du projet et lancement
Définition des besoins		Description des besoins métiers et de l'impact sur les processus métiers
Réalisation	Inception	Lancement de la réalisation : les besoins et le périmètre sont figés et les exigences identifiées et modélisées
	Elaboration	Consolidation exigences fonctionnelles, développements spécifications techniques
	Construction	Développement du projet

[20] [CAIRN] source : http://wiki.altair.recouv/doku.php?id=altair:domaines:methodologie_projet:start
[21] Source : http://wiki.altair.recouv/doku.php?id=altair:domaines:methodologie_projet:cycle_de_vie

Intégration	Qualification du logiciel produit et intégration dans le SI
Validation	Validation de l'adéquation aux besoins métiers
Diffusion	Mise en production, maintenance, formation au logiciel

Je n'ai pas pu aborder chaque phase du cycle de vie du projet. Comme nous le verrons plus loin, l'infrastructure en place ne m'a pas permis d'intégrer le logiciel dans le SI. D'autre part l'ampleur du projet m'a contrainte à le découper en plusieurs lots, dont un n'a pu être étudié. J'ai néanmoins pu tester la faisabilité des lots à ma charge, et ainsi pu réaliser les phases de cadrage, de définitions des besoins et de réalisation (sans toutefois achever la sous phase de construction). Ma démarche et mes résultats de travail sont exposés dans la suite du document.

5.1.3 Liste de livrables

La méthodologie CAIRN met à disposition un référentiel de documents livrables. Chaque phase du projet est jalonnée par l'initialisation de ces livrables ainsi que leur progression et validation. L'illustration en annexe (voir annexe V) offre une synthèse de tous ces documents ainsi que leur cycle de vie tout au long du projet.

Chaque projet s'approprie ce référentiel de livrables en fonction de ses habitudes de travail.

D'autres livrables sont aussi proposés pour le pilotage du projet comme le compte rendu de réunion et le RIDA[22].

Dans le cadre de mon projet, j'ai réalisé un cahier des charges (cf. 5.2.2), un dossier des cas d'utilisations pour recenser les spécifications fonctionnelles (cf. 5.3), une matrice de couverture des exigences (cf. 5.3.10) pour vérifier la couverture des exigences du cahier des charges par les exigences fonctionnelles, des spécifications techniques (cf. 6.3.2) pour guider le développement des lots. L'infrastructure actuelle ne permettant pas de mettre en œuvre les besoins, j'ai réalisé un POC (cf. 6.3.1) pour lequel j'ai produit un document de cadrage et un compte rendu. J'ai aussi utilisé des comptes rendu de réunion et le RIDA pour consigner les informations, décisions et actions abordées lors des différentes réunions qui m'ont permis de piloter mon projet.

Le projet n'ayant pas dépassé l'étape de construction, je n'ai pas produit les documents permettant la validation des phases de développement, intégration et validation. Ces documents seront rédigés respectivement à partir des spécifications techniques, dossier des cas d'utilisations et du cahier des charges en reprenant les exigences définies dans chacun d'entre eux. L'outil utilisé au Cirso pour gérer les tests est ALM (anciennement Quality Center édité par Hewlett-Packard).

[22] RIDA Relevé d'Informations, de Décision et d'Actions

5.1.4 Autres activités du cycle de vie du projet

La méthodologie regroupe aussi les activités de modélisation, de livraison et de chiffrage. Elle apporte des normes de modélisation pour l'analyse et la conception en UML du projet, organise comment et à qui doit-il être livré et met à disposition des règles et outils permettant de le chiffrer tout au long de son déroulement.

5.2 Expression des besoins

5.2.1 Recueil des besoins

Après une phase de découverte de l'environnement de travail et des outils utilisés, j'ai démarré le projet par une série de réunions. Ces réunions avaient pour but de m'aider à situer les besoins du client et les formaliser.

La première rencontre a donc consisté en une réunion de lancement de projet. A l'issu de chaque rencontre, je consignais les informations et décisions dans un compte rendu et reportais aussi les questions soulevées. Ces comptes rendu me servaient de fil conducteur d'une réunion à une autre.

Le profil des participants aux réunions étaient assez varié : développeur Watt, chef de projet développement Watt, chef de projet MOE Watt, consultant MOE Watt, responsable du pôle applications collaboratives. Un des participants se trouvaient même être un ancien gestionnaire, il avait donc une certaine idée des besoins réels d'un utilisateur du workflow. Sa participation était intéressante car nous apportait une perspective supplémentaire. Cette mixité était intéressante car chaque acteur avait un point de vue propre et soulevait des problématiques différentes. Mais elle présentait aussi des limites, plus il y avait de participants, plus il était fréquent de trouver des désaccords sur les exigences évoquées.

Une autre limite vite rencontrée a été de faire une liste pertinente des exigences. Tout au long du projet nous avons dû composer sans la MOA ou les utilisateurs. La MOA n'étant pas très présente, les équipes de développement et de MOE de Watt doivent être force de propositions.

Je me suis aussi retrouvée confrontée à un problème de disponibilité des participants. La phase de recueil des besoins est intervenue à un moment où les équipes de Watt étaient très mobilisées par la dernière vague de régionalisation (cf. chapitre 4.1). Par ailleurs, ils ont longtemps maintenu des doutes sur les besoins du projet ayant pour conséquence l'ajout ou le retrait répété d'exigences.

5.2.2 Itérations réalisées pour le cahier des charges

5.2.2.1 Vocabulaire et concept métier

Afin d'avoir une compréhension mutuelle et une visions globale des besoins à mettre en œuvre, nous nous sommes d'abord mis d'accord sur les termes utilisés pour décrire les besoins et les exigences. Cette mise au point a par ailleurs permis de structurer ensuite l'enchainement des exigences.

Nous avons retenus deux modes de collaboration nécessaires. Le premier concernant tout ce qui touche à la communication instantanée et le second aux notifications.

- Modes de collaboration

Les besoins font ressortir deux modes de communications nécessaires :

- Pouvoir communiquer instantanément avec un ou plusieurs utilisateurs : **communication instantanée**
- Pouvoir porter une information à une ou plusieurs personnes selon le contexte. Nous parlerons de **notifications**

Les fonctionnalités mises en œuvre pour répondre à ces besoins seront regroupées dans un espace dédié.

- Communication instantanée

Deux modes de communications instantanées vont être distingués.

- La notion de **communication point à point** sera utilisée pour parler de communication instantanée entre deux utilisateurs
- Lorsqu'il sera question de communication entre plusieurs membres, nous parlerons de **communication de groupe**

Les échanges par communication instantanée se matérialiseront par **une fenêtre de conversation**.

- Notifications

Certaines actions de modifications d'affaire vont générer des notifications celles-ci seront de type **notification sans alerte**.

D'autres notifications, par exemple un administrateur fonctionnel qui souhaite informer les utilisateurs d'une coupure applicative, généreront **une fenêtre d'alerte**.

Afin de garder une trace de ces notifications, elles seront stockées dans un **historique** quelque soit le type de la notification. Cet historique prendrait la forme d'un journal de bord.

5.2.2.2 Rédaction des exigences

Au terme des réunions, une cinquantaine d'exigences a été formalisée (voir tableau récapitulatif extrait du document de la matrice des exigences en annexe VI). Dans le cahier des charges issu de la méthodologie CAIRN, elles sont exprimées dans le cahier des charges au moyen d'un cartouche. Celui-ci respecte la structure suivante.

ID	CDC_WATT2_IFC_F_000103Espace dédié	
VS	1.0	*L'espace dédié est accessible quelque soit le contexte Watt lancé dans le*

CR	[Critique]	*portail Harmonie.* *Il affiche l'interface de messagerie instantanée et l'interface d'historique des notifications*
ST	En cours	

Figure 43 Cartouche de définition des exigences

Nous retrouvons dans le cartouche : l'identifiant de l'exigence et son libellé, une description, sa version, sa criticité et son statut.

L'identifiant est soumis à des règles de nommage. Il prend la forme :<Niveau>_<Projet>_<Type>_<N°>. Où <Niveau> représente le trigramme du document (ici CDC pour cahier des charges), <Projet> une abréviation du projet concerné, <Type> une lettre catégorisant l'exigence (F pour fonctionnelle, R pour Règle de gestion, T pour Technique ou non fonctionnelle et O pour organisationnelle).

Les exigences ont donc été réparties entre trois catégories, elles même redécoupées en sous catégories. Ces catégories sont les généralités, les types de collaborations et les fonctionnalités.

- Généralités

Cette petite partie traite des points généraux non abordées dans les autres sections : la connexion, l'utilisation de la communauté actuelle et la définition de l'espace dédié.

- Types de collaboration

  ✓ Communication instantanée

Dans cette partie, nous avons détaillé toutes les exigences se rapportant à la communication instantanée.

Tout d'abord, nous avons décrit les moyens d'accès à cette communication qui se ferait par la matérialisation d'un client de messagerie dans l'application Watt mais aussi par l'utilisation de fonctionnalités disponibles dès qu'un couple nom_prénom d'un acteur apparaît dans l'application (au niveau de la corbeille d'affaire et de l'interface d'information de l'affaire).

Puis les exigences concernant la communication point à point et la communication de groupes ont été rédigées. Celles-ci décrivent comment doit se faire l'émission et la réception des messages, ainsi que les modalités concernant une fenêtre de communication.

  ✓ Notifications

Dans cette catégorie, nous avons distingué les notifications faisant l'objet d'une alerte et celles apportant simplement une information consignée dans l'historique. Ce dernier a aussi été spécifié dans les exigences.

Une liste d'action pouvant déclencher automatiquement des notifications sans alerte a été dressée. Par exemple, l'attribution d'une affaire à un gestionnaire fait apparaitre l'information dans l'historique de ce dernier.

Les exigences portant sur les notifications avec alertes vont spécifier la fenêtre pop up d'alerte, le stockage de l'information dans l'historique, les actions à l'origine de ce type de notification et la mise en place d'un espace de création de ces alertes.

La partie sur l'historique mentionne les règles de classement et de purge des notifications, la mise en place d'indicateurs de notifications non lues ainsi que leur nombre et la possibilité de les distinguer facilement des plus anciennes.

- Fonctionnalités

Cette section a aussi été divisée en sous catégories : les fonctionnalités propre à la communication instantanée, les options et la gestion du statut.

Les besoins en terme de fonctionnalité vont être de pouvoir gérer une liste de contact sous Watt et d'avoir accès à un outil de recherche dans le cas où l'acteur à contacter n'apparait ni sur la liste de contact ni sur les informations d'une affaire dans Watt.

Les options spécifiées sont l'affichage de la carte de visite des contacts et la possibilité de communiquer avec le manager d'un acteur depuis le couple nom_prénom de celui-ci dans Watt.

5.2.2.3 Validation des exigences

Comme expliqué dans le chapitre sur le recueil des besoins (cf. p.58), il a été difficile de satisfaire toutes les parties et les besoins n'étaient pas clairs dès le départ.

J'ai dû planifier plusieurs réunions pour avancer dans la rédaction du cahier des charges.

Lors de chaque réunion, nous passions en revue les exigences écrites et évoquions celles qui manquaient. Certaines exigences faisaient l'objet d'une suppression car n'apparaissaient finalement plus nécessaires aux yeux des participants.

Je profitais du temps entre chaque rencontre pour compléter le cahier des charges avec les nouveaux besoins et mettre à jour les autres en fonction des remarques faites.

Ces modifications faisaient l'objet d'une validation à la réunion suivante.

Il a fallu mener plusieurs réunions, réparties sur deux mois pour parvenir à la validation du cahier des charges. Il aura aussi fallu que je réalise que je devais m'imposer en responsable du projet d'implémentation et assurer l'animation des réunions et savoir trancher par moment pour pouvoir avancer ou bien éviter que le sujet initialement prévu ne dérive sur un tout autre sujet.

5.2.2.4 Définition du niveau de criticité des exigences

Au vu du nombre important d'exigences, nous avons fait une dernière réunion pour arbitrer sur le niveau de criticité des exigences. Ceci pour décider des thèmes à traiter en priorité.

Trois niveaux de criticité sont formalisés : critique, moyenne et faible.

- Niveau critique

La priorité s'est donc portée sur les généralités, la communication point à point, l'historique des notifications (avec notifications déclenchées automatiquement) et la gestion des statuts.

- Niveau moyen

La mise en place des notifications avec alerte et la fonctionnalité « liste de contact » sont secondaires.

- Niveau faible

L'importance accordée aux fonctionnalités autres que « Liste de contacts » et aux options est moindre.

Les principaux efforts de conception et de réalisation seront focalisés sur la mise en place de la possibilité pour un utilisateur de Watt de communiquer aisément depuis l'application avec un autre acteur. Le développement des fonctionnalités sera intégré si celles-ci sont présentes de base dans les solutions qui seront choisies puis déployées sur Watt. Ce lot intégrera aussi les exigences de généralités et de gestion des statuts.

La réalisation de l'historique constituera un deuxième lot.

5.3 Cas d'utilisation (ou CU)

Les cas d'utilisation ont été rédigés à partir des exigences exprimées dans le cahier des charges. Pour faciliter leur utilisation et leur lecture, ils ont été regroupés sous sept thématiques différentes :

- Les impacts dans Watt
- L'espace dédié (hors notifications)
- Les notifications
- La fenêtre de communication
- Les fonctionnalités offertes par le nom/prénom
- La liste des contacts
- L'espace de création d'alertes

Le choix de ces parties reflète en partie la répartition faite au niveau du cahier des charges. Afin de vérifier qu'aucune exigence n'était oubliée des diagrammes de cas d'utilisations créés, j'ai utilisé une matrice de vérification de couverture des exigences.

Pour ne pas reproduire les difficultés rencontrées par les nombreuses itérations de rédaction du cahier des charges, j'ai choisi de rédiger le document des cas d'utilisation dans sa totalité avant de le soumettre à validation à l'équipe du projet.

5.3.1 Description des cas d'utilisation

Le document des cas d'utilisation (ou DCU) fourni par la méthodologie CAIRN propose une description détaillée pour chaque cas d'utilisation. Cette description suit un plan précis dans lequel nous retrouvons l'objectif du CU qui reprend les exigences du cahier des charges accompagné d'une description du service attendu et d'indicateurs de version, criticité et statut. Les autres items de la description du CU porteront sur les acteurs, les déclencheurs et évènements, les scénarios nominaux, alternatifs et d'exception, les exigences non fonctionnelles, les dépendances avec d'autres CU et l'IHM proposée. Voici une illustration de ce formalisme avec la description d'un cas d'utilisation du paragraphe suivant.

Objectif

ID	**DSF_ WATT2_IFC _A02 Se connecter à la communauté du recouvrement**	
VS	1.0	Ce cas d'utilisation permet de se connecter à la communauté du recouvrement pour pouvoir utiliser les fonctionnalités de collaboration offertes par la messagerie. *Connexion automatique à l'ouverture de WATT ou manuelle* *L'utilisateur est connecté à la communauté recouvrement* *L'utilisateur peut discuter avec n'importe quelle personne de l'annuaire DONATIR*
CV	CDC_WATT2_IFC_F_000101 CDC_WATT2_IFC_F_000102	
CR	Critique	
ST	Validé	

Lien avec le(s) processus métiers de niveau 4

NA

Acteurs

Utilisateur

Pré-conditions

Avoir une fenêtre watt ouverte

Déclencheurs et evenements

Le cas d'utilisation est déclenché par l'accès à Watt ou par le déclenchement de l'utilisateur depuis l'espace dédié dans le cas où la messagerie a été déconnectée après l'ouverture de Watt.

Scenario nominal

1. Le système se connecte au serveur de messagerie
2. La connexion réussit ou échoue

Scenario alternatif 1 – Accès par Espace Dédié

1. L'utilisateur sélectionne l'option Se connecter depuis le menu de l'espace dédié
2. Le système se connecte au serveur de messagerie
3. La connexion réussit ou échoue

Scénarios d'exceptions

NA

Tableau de correspondance

NA

Exigences non fonctionnelles

NA

Post-conditions

Propagation de la mise à jour du statut vers les autres utilisateurs (voir UC F01)

IHM

N/A

Figure 44Illustration du formalisme de description des CU avec le cas « Se connecter à la communauté recouvrement »

5.3.2 Les profils

L'ensemble des acteurs concernés par les besoins de communication/notifications sont :

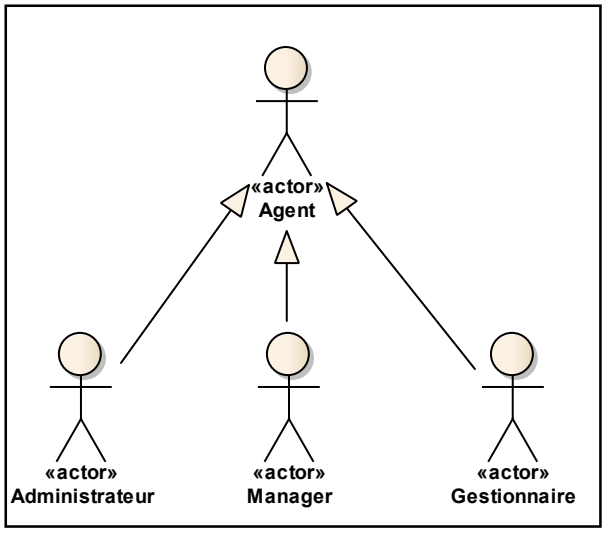


Figure 45 Les différents profils

- Gestionnaire : il gère un ensemble d'affaires selon son rôle dans l'organigramme. Il accède à sa corbeille d'affaires via la perspective WATT PILOT de WATT2 et aux demandes en attente d'injection via la perspective WATT INJECTION.
- Manager : c'est un « gestionnaire » de gestionnaire ayant des droits particuliers lui donnant accès à des corbeilles de gestion particulières dites « manager » via la perspective WATT PILOT de WATT2 et aux demandes en attente d'injection via la perspective WATT INJECTION.
- Administrateur : personne ayant des droits particuliers lui donnant accès à la partie administration de WATT 2 via la perspective WATT ADMINISTRATION.

Ces trois profils appartiennent à un profil commun « agent » ou « utilisateur ». Ce profil sera le plus utilisé dans les diagrammes car beaucoup de cas d'utilisation sont commun aux profils d'utilisateur de Watt concernés par l'implémentation des fonctionnalités de collaboration.

5.3.3 Impacts dans Watt

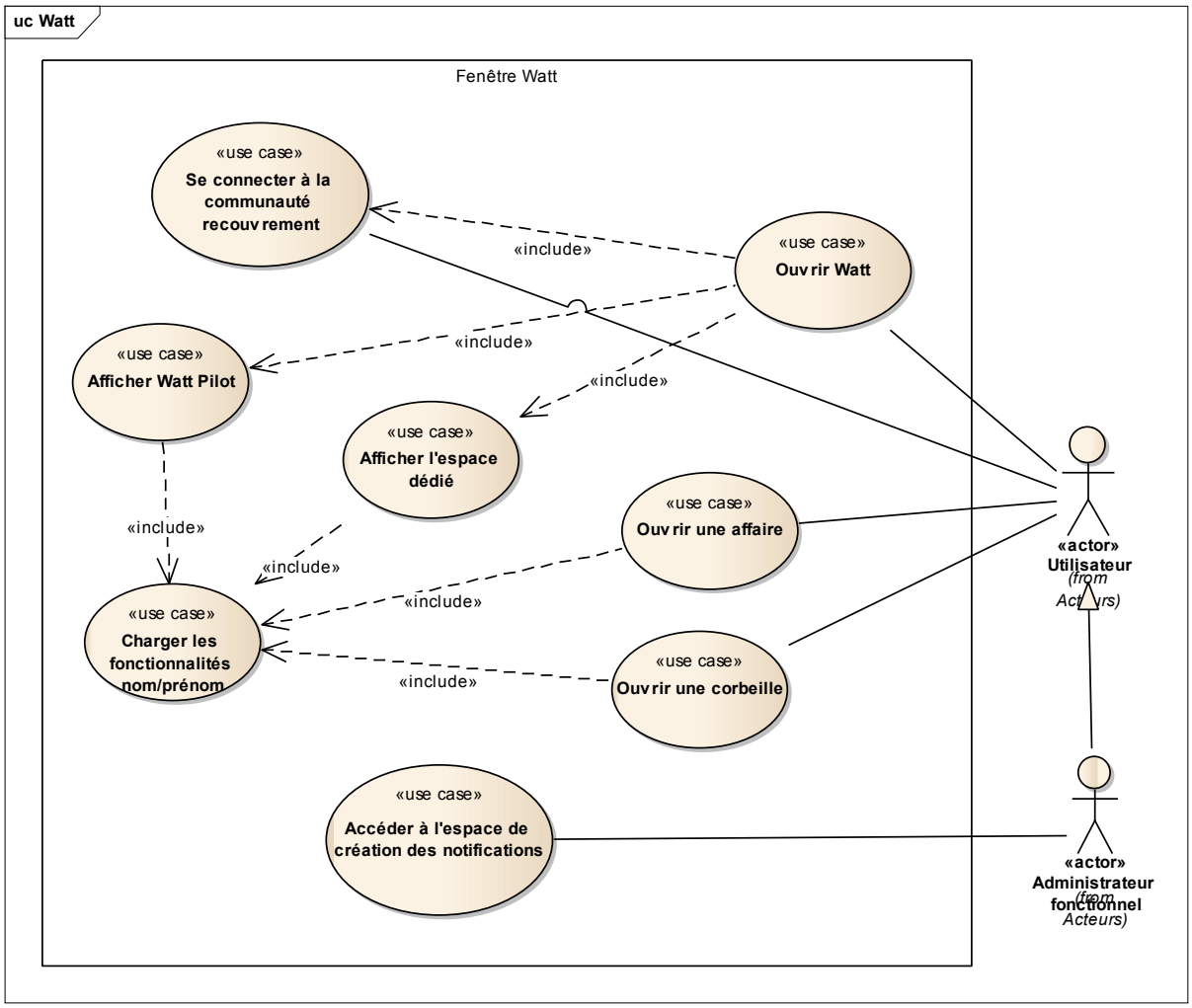


Figure 46 Diagramme des cas d'utilisations des impacts sur Watt

Le premier diagramme porte sur les fonctionnalités à mettre en œuvre dès l'ouverture de Watt.

Lorsqu'un utilisateur se connecte à Watt, l'application s'ouvre sur Watt Pilot. Avec la mise en place des fonctionnalités collaboratives, l'ouverture de Watt va générer de nouveaux cas d'utilisation comme la connexion à la communauté du recouvrement (communauté utilisée par le client Sametime actuel embarqué dans Lotus Notes). Mais il va aussi engendrer l'ouverture de l'espace dédié de communication. Tous les cas d'utilisations de l'espace dédié sont traités dans un diagramme à part (cf. paragraphe 5.3.4).

L'affichage de Watt Pilot et de l'espace dédié vont inclure le cas d'utilisation du chargement des fonctionnalités nom_prénom. En effet, ces deux espaces affichés comportent des couples nom_prénom, ils doivent donc proposer les nouvelles fonctionnalités exigées dans le cadre de l'expression des besoins. Cette action sera aussi induite par l'ouverture d'une corbeille ou d'une affaire par un utilisateur. Les fonctionnalités offertes par le couple nom_prénom font aussi l'objet d'un diagramme distinct (cf. paragraphe 5.3.7).

L’accès à l’espace de création de notification (cf. paragraphe 5.3.9) par l’administrateur va aussi impacter sur l’application de manière générale. L’administrateur doit pouvoir le faire dès lors qu’il se trouve sur l’application.

5.3.4 Espace dédié (hors notifications)

Ce diagramme ne prend pas en compte les cas d’utilisations qui concernent l’historique de notification, celui-ci étant inclus dans le diagramme des notifications (cf. paragraphe 5.3.5).

Dans ce diagramme, nous retrouvons deux sous ensembles. Celui se rapportant aux exigences sur les fonctionnalités non traitées dans les autres diagrammes et celui sur la gestion des contacts.

En effet, l’espace dédié va permettre de gérer des contacts et les aspects de connexion. Si l’utilisateur ne souhaite pas être connecté à la messagerie, il aura la possibilité de désactiver la connexion depuis cet endroit. Il pourra aussi gérer son statut et effectuer des recherches d’agents en fonction de leur nom et/ou prénom.

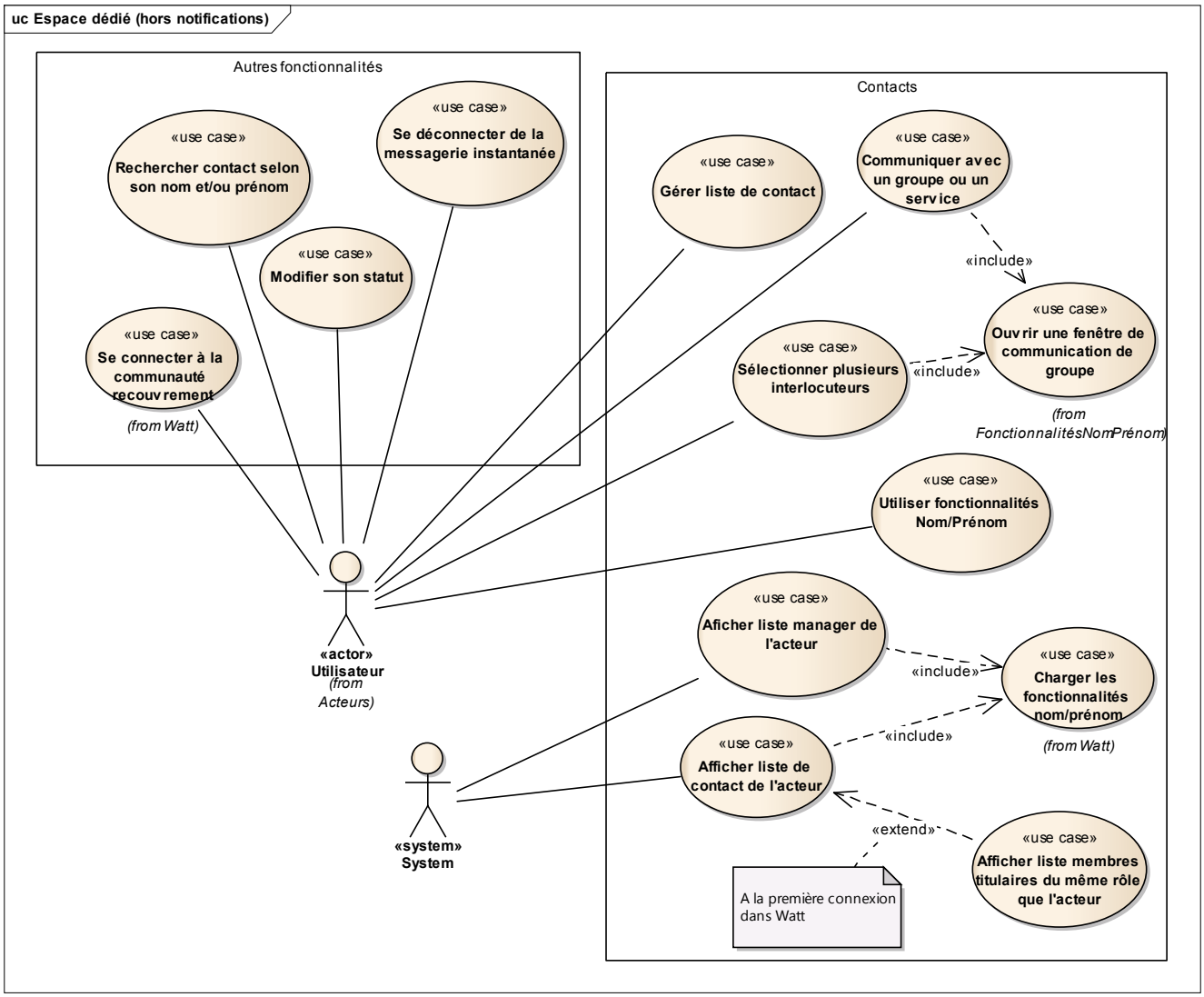


Figure 47 Diagramme des cas d'utilisation de l'espace dédié

La partie « Contacts » permet à l'utilisateur de gérer ses contacts (cf. paragraphe 5.3.8), de communiquer avec plusieurs interlocuteurs en même temps et d'utiliser les fonctionnalités des couples nom_prénom constituant la liste de contacts.

La communication avec plusieurs personnes à la fois peut se faire de plusieurs façons différentes : soit en sélectionnant tous les interlocuteurs depuis la liste de contacts, soit en lançant une discussion avec un groupe (tous les membres du groupe recevront alors une invitation de discussion). Ces actions génèrent l'ouverture d'une fenêtre de communication de groupe. Ce cas est explicité dans la partie « Fonctionnalités nom_prénom » (cf. paragraphe 5.3.7).

L'utilisateur doit aussi pourvoir visualiser la liste de ses managers et sa liste de contacts. Ces cas incluent le chargement des fonctionnalités nom_prénom. L'équipe de projet a émis le souhait d'avoir un groupe prédéfini à la première connexion dans Watt. Ce groupe afficherait les membres de même rôle que l'utilisateur. Libre ensuite à l'utilisateur de le conserver ou non grâce aux fonctionnalités de gestion des contacts.

5.3.5 Notifications

Les notifications vont trouver leur origine dans deux environnements différents.

Elles peuvent être générées automatiquement depuis une action faite par le manager depuis la corbeille. S'il modifie la priorité d'une affaire, réaffecte l'affaire à un autre gestionnaire ou bien qualifie une affaire « d'urgence de production », alors une notification sera directement générée. Par ailleurs, un indicateur informant qu'une notification a été émise sur l'affaire apparaîtra dans la corbeille du gestionnaire de l'affaire impactée.

La notification peut aussi être émise par l'administrateur depuis l'espace de création de notification (cf. paragraphe 5.3.9). L'administrateur a aussi la possibilité de choisir une notification de type alerte. Dans ce cas les utilisateurs concernés par son envoi, recevront s'ils sont connectés (ou se connectent durant le temps de validité de l'alerte), le message sous forme de fenêtre pop-up.

La notification apparaîtra dans le centre de notification et l'utilisateur pourra donc la visualiser. Il pourra aussi recevoir une alerte et valider la fenêtre pop up.

Ce centre de notification, prendra la forme d'un historique où seront affichées les notifications classées dans l'ordre chronologique. S'il s'agit d'une nouvelle notification, l'utilisateur le visualisera facilement à l'aide d'indicateurs du nombre de notifications non lues ainsi que de la présence de nouvelles notifications.

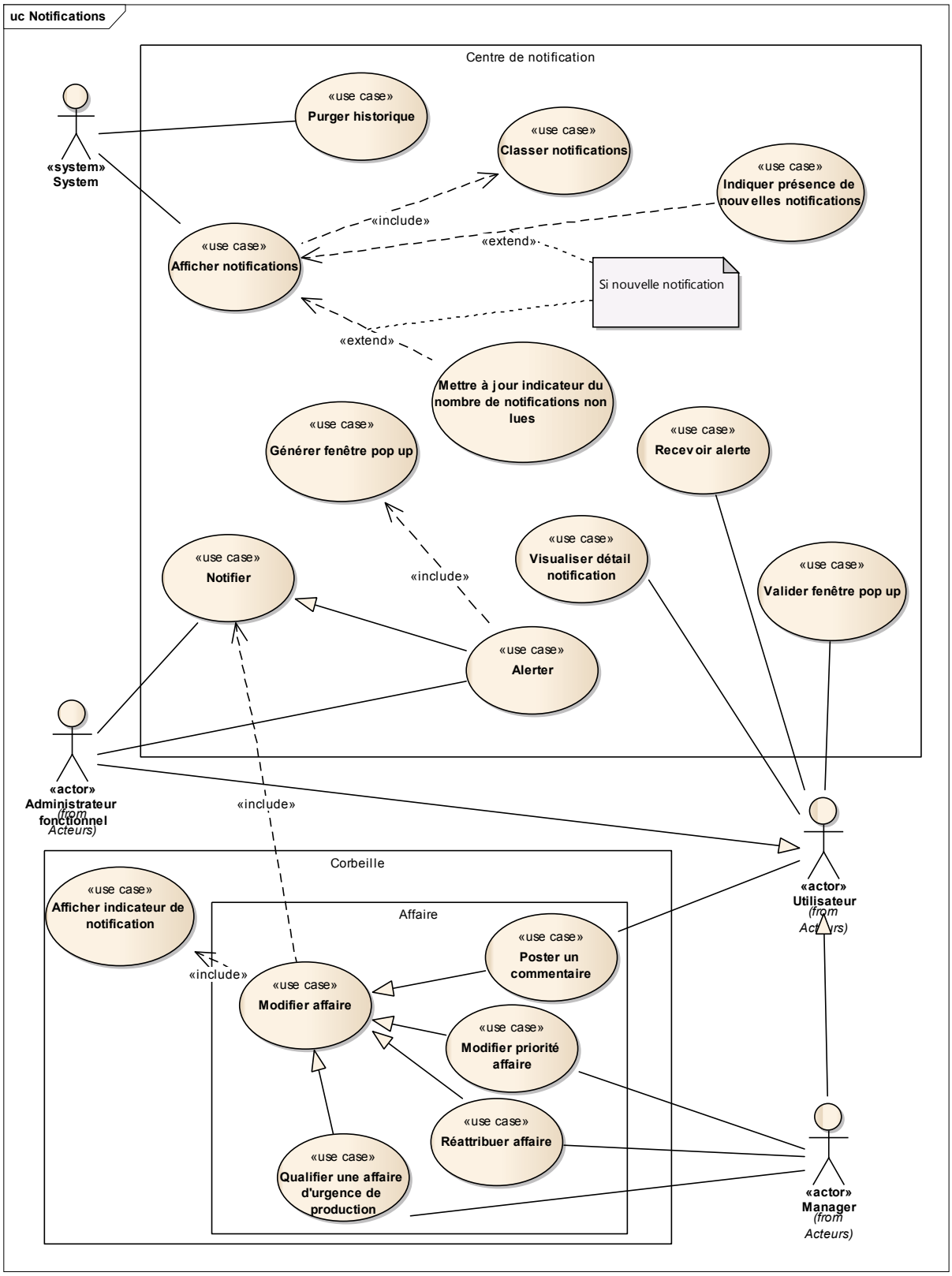


Figure 48 Diagramme des cas d'utilisation des notifications

5.3.6 Fenêtre de communication

La partie fenêtre de communication se compose en deux sections. La première concerne la fenêtre de communication et la deuxième celle utilisée dans le cadre d'une communication à plusieurs.

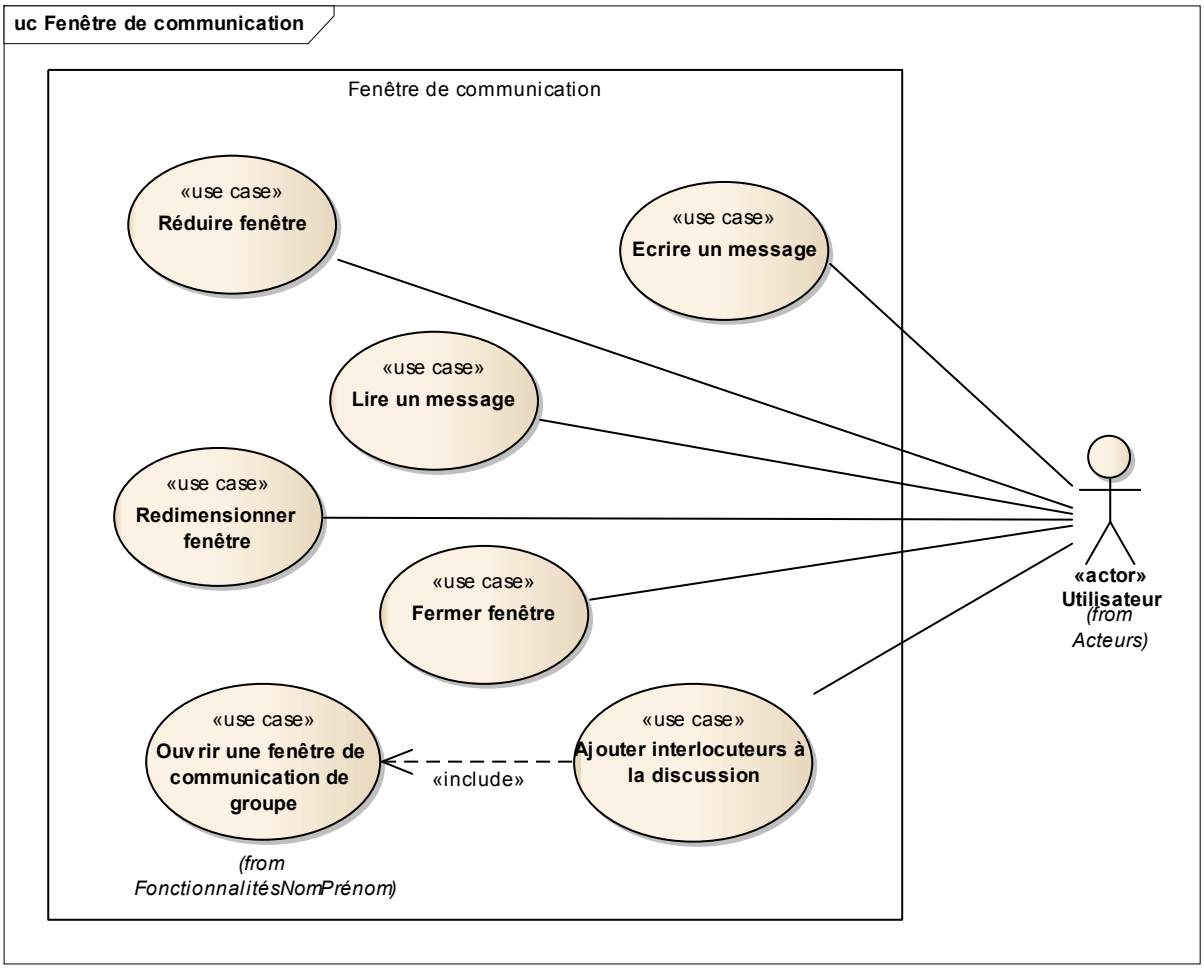


Figure 49 Diagramme des cas d'utilisation de la fenêtre de communication

La fenêtre de communication doit permettre un ensemble d'actions de gestion par l'utilisateur. Il peut la fermer, la réduire ou bien la redimensionner. Cette fenêtre est aussi l'interface qui lui permet d'écrire le message à son interlocuteur et de lire le contenu de la discussion.

Il peut aussi ajouter de nouveaux interlocuteurs depuis cette fenêtre. Nous retrouvons alors une configuration de fenêtre de communication de groupe.

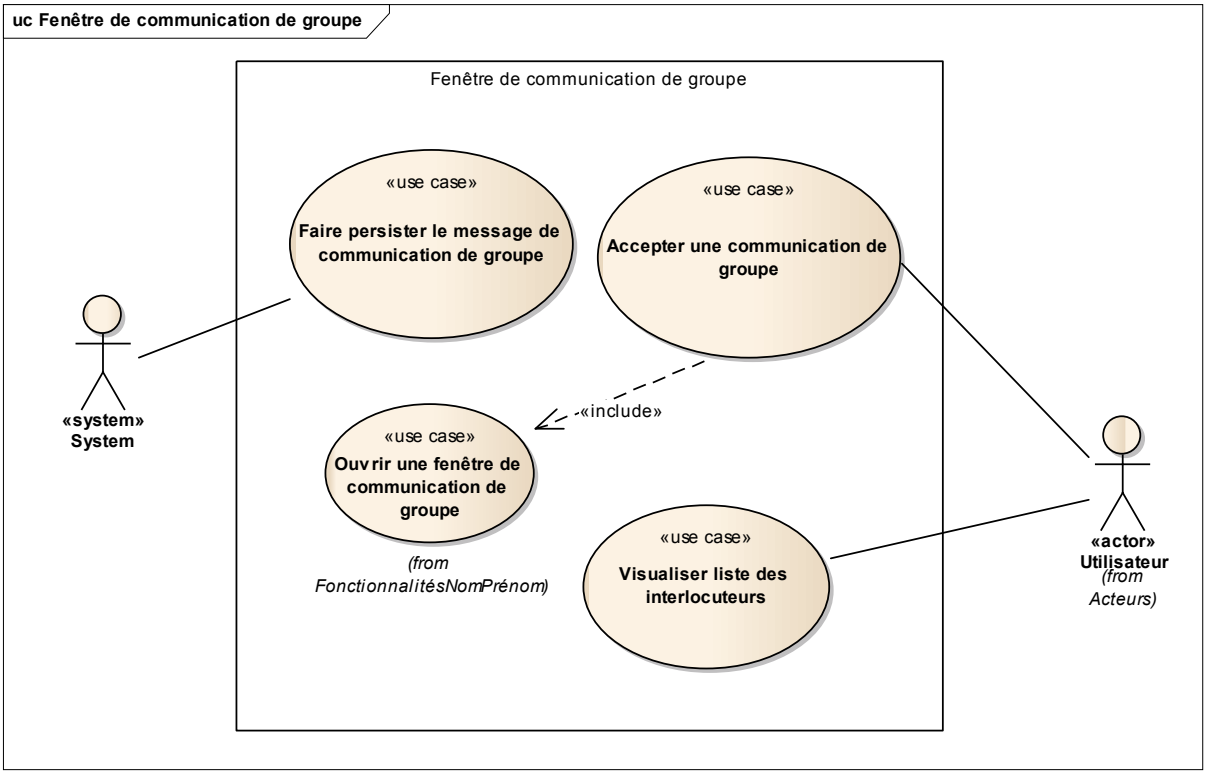


Figure 50 Diagramme des cas d'utilisation de la fenêtre de communication de groupe

La fenêtre de communication de groupe est une extension de la fenêtre de communication. Nous y retrouvons donc tous les cas d'utilisation vus dans le diagramme précédent auxquels viennent s'ajouter des cas spécifiques.

Tout d'abord un utilisateur contacté par un autre reçoit une invitation à rejoindre la discussion de groupe. Celui-ci peut accepter ou décliner. En cas d'approbation, la fenêtre de discussion de groupe s'ouvre. L'utilisateur a alors la possibilité de visualiser la liste des participants à la discussion et de lire la discussion depuis sa création par l'auteur.

5.3.7 Fonctionnalités offertes par le nom/prénom

J'ai réuni dans ce diagramme toutes les fonctionnalités que l'équipe projet souhaiterait utiliser depuis un couple nom_prénom dans l'application.

Les premières fonctionnalités concernent les actions possibles depuis un clic droit sur le couple nom_prénom. Un menu doit proposer la possibilité de communiquer avec la personne, de communiquer avec le manager de cette personne ou d'afficher sa carte de visite.

Un gestionnaire peut avoir plusieurs managers. Dans l'organigramme, chaque rôle est précédé d'un manager qui lui-même peut avoir un manager. Il n'est pas rare qu'un gestionnaire ait plusieurs rôles attribués, il se retrouve donc affecté à plusieurs managers. Le cas d'utilisation « Communiquer avec le (ou les) N+1 du contact » déclenchera donc l'ouverture d'une fenêtre de communication de groupe avec invitation de tous les managers si le contact en a plusieurs, dans le cas contraire, ce sera une fenêtre de communication simple qui s'ouvrira.

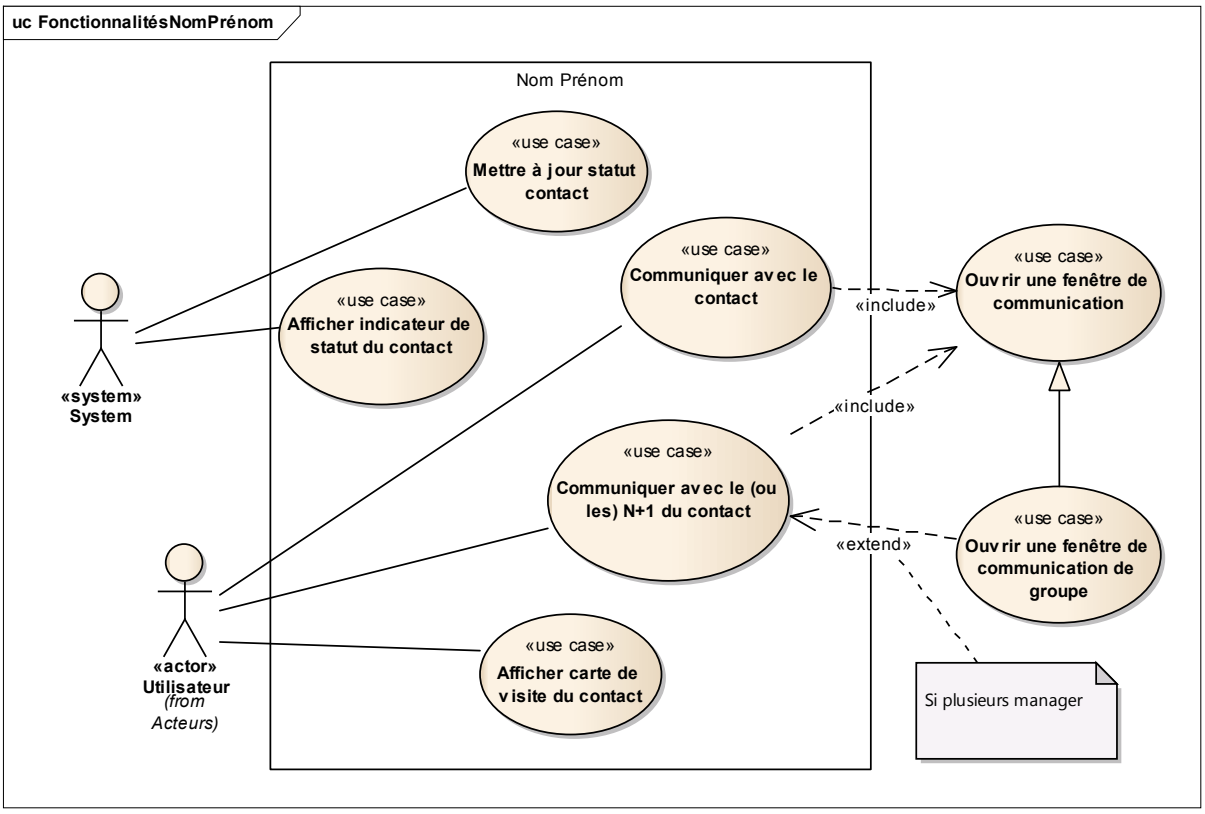


Figure 51 Diagramme des cas d'utilisation des fonctionnalités nom_prénom

Le couple nom_prénom doit aussi permettre de visualiser le statut du contact. Cette notion de conscience de la présence souvent trouvée sous l'appellation de « awareness » est appelée « présentiel » par les équipes du CIRSO. L'utilisateur doit donc avoir la possibilité, au moyen d'un indicateur graphique, de connaître la disponibilité du contact. Les différents statuts sont : « en ligne », « absent », « occupé », « hors ligne » et « ne pas déranger ». Le statut pouvant changer régulièrement, il est important que le système procède à sa mise à jour régulièrement.

5.3.8 Liste des contacts

Du fait du nombre important de cas d'utilisation découlant de la gestion de la liste de contacts, j'ai créé un diagramme de cas d'utilisation à part entière.

L'utilisateur peut effectuer un nombre d'actions assez important sur sa liste de contact. Il doit pouvoir en ajouter ou supprimer. Les contacts sont classés dans des groupes. L'ajout/suppression d'un contact va donc inclure une modification du groupe.

L'utilisateur peut aussi modifier directement le groupe en le renommant. Il peut organiser sa liste en créant et supprimant des groupes.

L'annuaire des agents Urssaf permet aussi de récupérer les listes de services entiers. L'utilisateur a donc la possibilité d'ajouter ou de supprimer un service dans sa liste. Le service apparaîtra sous forme de groupe dans lequel seront renseignés les nom_prénom de chaque membre du service.

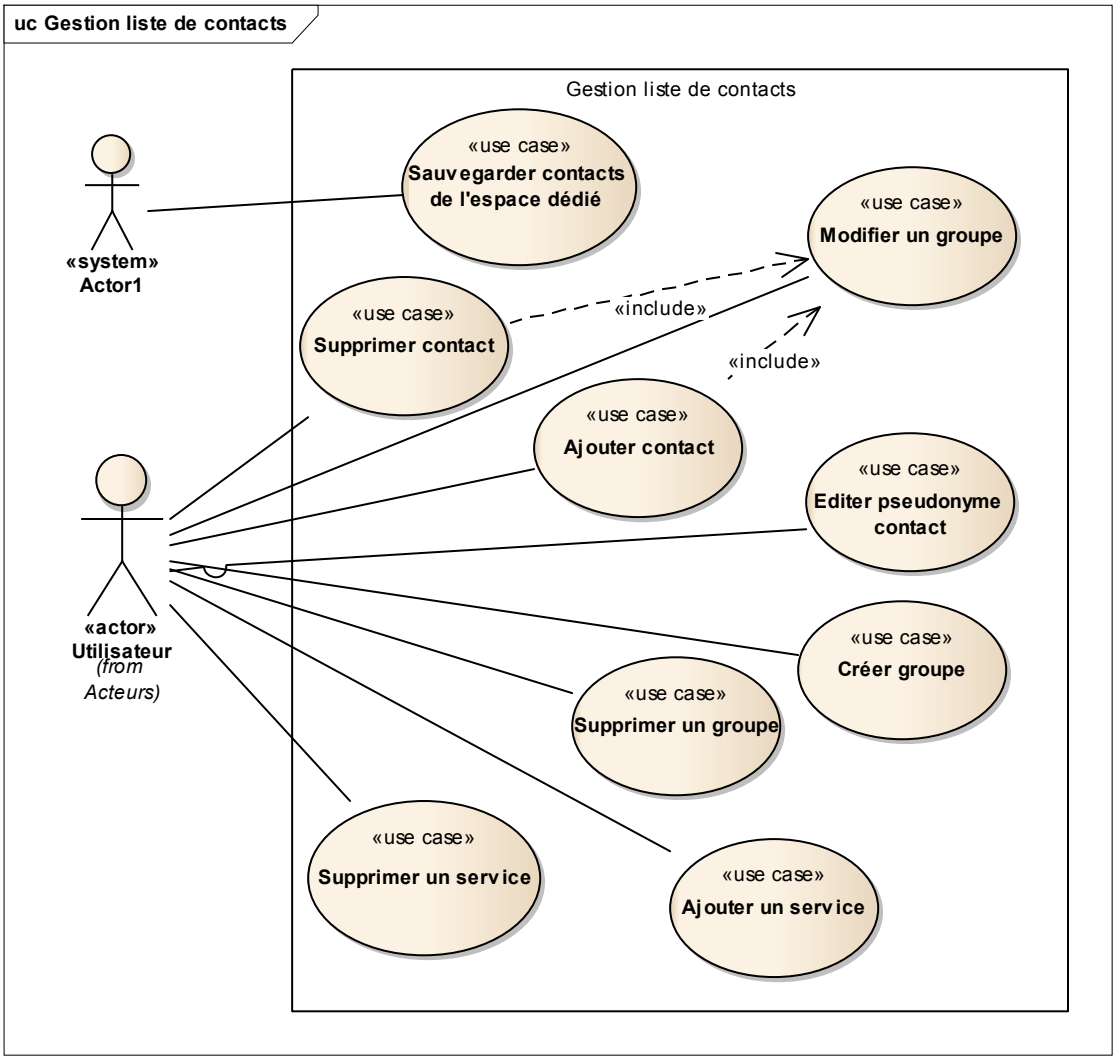


Figure 52 Diagramme des cas d'utilisation de gestion de la liste de contacts

Afin de permettre une personnalisation plus exhaustive de la liste, l'utilisateur peut même éditer le nom des contacts.

Le système devra sauvegarder la liste telle qu'elle sera après personnalisation par l'utilisateur.

5.3.9 Espace de création d'alertes

Afin de pouvoir notifier les utilisateurs de l'application, en cas de modification d'un circuit par exemple, l'administrateur fonctionnel aura a sa disposition un espace de création d'alertes. Celui-ci doit lui permettre d'écrire le message à faire passer aux utilisateurs et de définir les destinataires. L'administrateur choisira s'il souhaite émettre la notification sous forme d'alerte.

Enfin, il pourra fermer l'espace lorsque la notification sera créée.

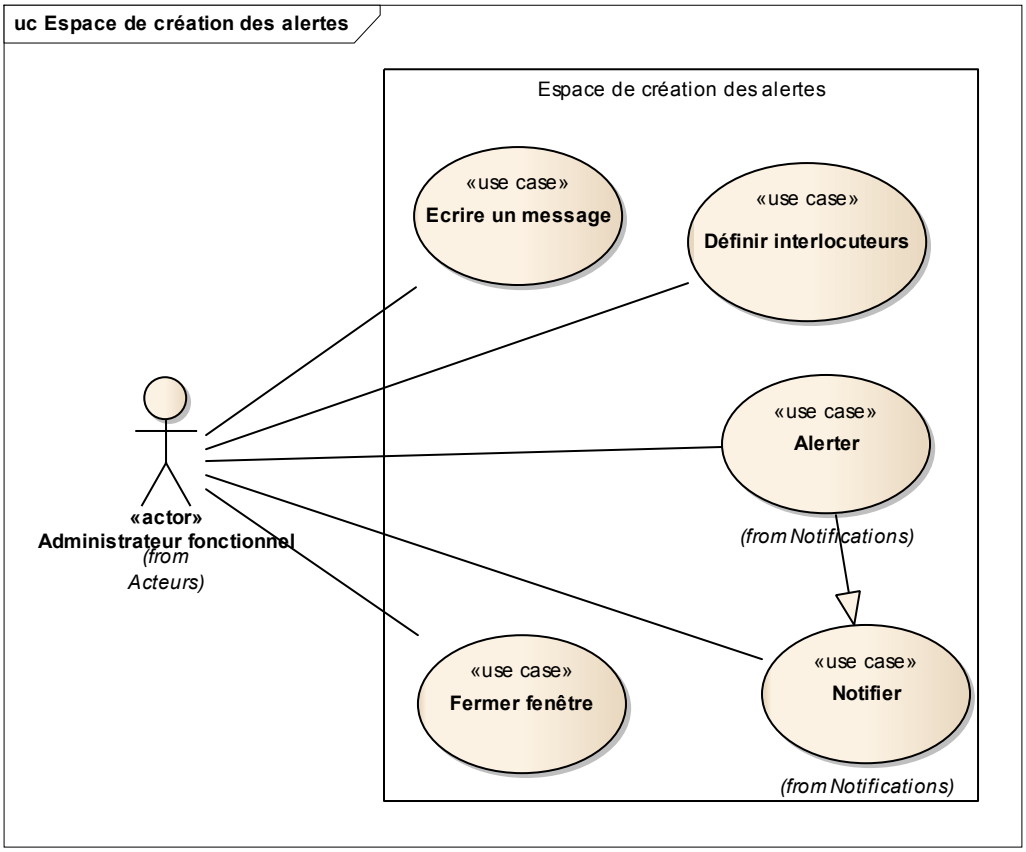


Figure 53 Diagramme des cas d'utilisation de l'espace de création des alertes

5.3.10 Validation du cahier des cas d'utilisations

Après avoir vérifié que toutes les exigences du cahier des charges étaient couvertes par les cas d'utilisation, j'ai proposé à l'équipe du projet de le valider.

La vérification de la couverture des exigences a été possible grâce à un outil proposé par la méthodologie CAIRN sous la forme d'une matrice. Cette matrice m'a d'ailleurs permis de réaliser que certaines exigences étaient redondantes. Une nouvelle mise à jour du cahier des charges a donc été faite après la validation du DCU.

La matrice de traçabilité des exigences est un document qui recense tous les liens pouvant exister au sein d'un projet. Elle permet de confectionner différentes vues du système à partir de l'ensemble des exigences. Ces liens permettent de gérer des exigences interconnectées situées dans différentes hiérarchies. Par exemple, nous pouvons commencer à partir d'une ligne d'exigences du cahier des charges, puis passer à une liste d'exigences du dossier des spécifications fonctionnelles et finir par la liste des tests réalisés.

Pour plus d'efficacité, j'ai organisé plusieurs réunions de trente à soixante minutes réparties sur la semaine. Chaque réunion était focalisée sur un ou deux diagrammes ainsi que sur les descriptions de tous les cas d'utilisations présents dans le diagramme.

J'ai demandé aux participants de prendre connaissance des parties qui allaient être traitées à la réunion suivante. Ainsi nous pouvions discuter directement des points qui ne leur convenaient pas ou bien sur lesquels ils avaient des interrogations.

La validation du dossier des cas d'utilisation s'est faite beaucoup plus rapidement que celle du cahier des charges.

6 Conception

6.1 Etude de faisabilité

L'information et les retours d'expérience sur l'utilisation des toolkits du SDK sont quasiment inexistants sur internet. Nous avons reçu la visite d'un partenaire d'IBM qui est venu nous faire une présentation succincte du SDK et nous a conseillé sur les toolkits les plus pertinents à utiliser selon les actions à mettre en œuvre. A l'issue de cette intervention, nous avons donc arbitré sur le choix des toolkits.

Nous avons donc étudié les différents scénarios de mise en œuvre des toolkits selon le type de fonctionnalité.

6.1.1 Rappel des besoins

Le projet se découpe en 4 grandes parties :

- ✓ Mettre à disposition de l'utilisateur un espace dédié de communication qui proposerait un client web de messagerie instantanée
- ✓ Permettre à l'utilisateur de connaitre l'information de présence des acteurs dont le nom/prénom apparaît dans les fenêtres de travail Watt (par exemple dans la corbeille, dans les informations d'une affaire…) par le biais d'un indicateur graphique.
- ✓ Ce couple nom/prénom doit aussi permettre quelques fonctionnalités de collaborations, comme démarrer une discussion avec l'acteur, accessibles au moyen d'un clic droit.
- ✓ Mettre en place un système d'alertes et notifications avec création un historique des messages dans l'espace dédié

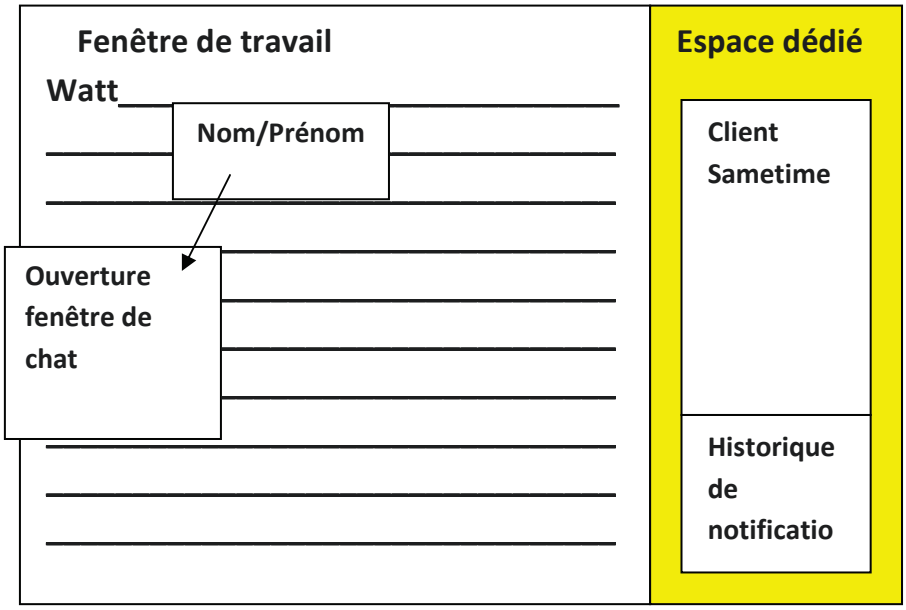


Figure 54 Découpage des besoins dans Watt

La conception de la partie notification n'a pas été traitée dans le cadre de mon stage. Elle fera l'objet d'un nouveau projet, la solution technique identifiée par le CIRSO pour cette partie porterait sur l'utilisation d'Activity Stream[23] qui est fourni par IBM dans une des solutions déployées aux URSSAF. Une étude doit être faite ultérieurement sur la faisabilité de cette solution.

6.1.2 Fonctionnement technique des toolkits

6.1.2.1 Toolkit Java

- Présentation

Le toolkit Java est modulaire, il propose un ensemble de composants qui permet de réutiliser les services et fonctionnalités de Sametime dans des applications métiers. Ces composants peuvent être utilisés dans n'importe quel environnement de développement qui supporte JDK 1.4.2 ou Java 5.0 et autres versions plus récentes. Cette modularité permet de ne charger que les composants utilisés dans le package de l'application. Cela permet de concevoir des livrables plus légers et aussi un temps de chargement réduit pour l'utilisateur.

- Services proposés

Le toolkit propose les services suivants (voir tableau récapitulatif en annexe VII) : alertes, présentiel, liste d'utilisateurs, gestion des sessions (statut, connexion…), annuaire, transfert de fichiers, messagerie instantanée, lookup (pour retrouver un groupe ou un utilisateur sur le serveur de communauté), multicast[24], gestion des pseudonymes en local, réunions virtuelles,

[23] Activity stream : solution proposée par IBM permettant la mise en place d'un fil d'activité dans les applications

[24] Définition wikipédia : Le multicast (qu'on pourrait traduire par « multidiffusion ») est une forme de diffusion d'un émetteur (source unique) vers un groupe de récepteurs. Les termes « diffusion multipoint » ou « diffusion

envoi d'un message à plusieurs utilisateurs (par exemple une invitation ou un rappel pour une réunion) , accès aux attributs de l'utilisateur et token LTPA.

Pour chacun des services le toolkit propose un package. Ils sont détaillés dans le document stjavatkdevguide.pdf qui est inclus dans le répertoire \doc du toolkit. Le toolkit propose aussi une documentation technique des packages sous forme de javadoc accessible depuis le même répertoire.

- Conditions d'utilisation

Pour utiliser ce toolkit il faut intégrer les 3 fichiers CommRes.jar, STComm.jar et stjavatk.jar, présents sous \client\stjava\bin dans le dossier du SDK.

Pour intégrer le toolkit dans Eclipse il faut prévoir des points d'extension (points d'accès au plug-in), sinon il est à intégrer dans une servlet[25] sur un autre serveur. Les services proposés par le toolkit sont basés sur des listeners[26].

- Initialisation

Quelques soient les services à implémenter, il faut réaliser une initialisation pour pouvoir utiliser les composants associés. Les éléments de bases sont l'instanciation d'un objet STSession, le chargement des composants, le démarrage et l'arrêt d'une session Sametime, la connexion et la déconnexion ainsi que la mise en place des listeners de connexion et déconnexion. Il est possible de trouver des informations plus détaillées dans l'annexe B du guide du développeur (cité précédemment) ou dans les pages 27 à 30 du tutorial du toolkit (stjavatktutorial.pdf accessible au même endroit que les autres documents).

  - STSession

L'objet STSession va permettre de gérer les composants et d'y accéder. Sa création se fait avec le code suivant :

```
m_session = new STSession("Nom de la session " + this);
```

Une fois STSession créé, nous pouvons charger les composants du projet puis démarrer la session et les composants.

```
m_session.loadAllComponents();
m_session.start();
```

Il faut aussi prévoir l'arrêt et le déchargement de la session lorsque l'application est arrêtée.

```
m_session.stop();
```

---

de groupe » sont également employés. Les récepteurs intéressés par les messages adressés à ce groupe doivent s'inscrire à ce groupe.

[25] Définition wikipédia : Un servlet est une classe Java qui permet de créer dynamiquement des données au sein d'un serveur HTTP.

[26] Définition wikipédia : Le listener, en français écouteur, est un terme anglais utilisé de façon générale en informatique pour qualifier un élément logiciel qui est à l'écoute d'évènements afin d'effectuer des traitements.

```
m_session.unloadSession();
```

- Connexion/Déconnexion

Afin de pouvoir utiliser les services, il faut se connecter au serveur de communauté. L'interface des services du serveur est implémentée par le composant STBase, il est responsable de la connexion et déconnexion au serveur. La méthode STSession.getCompApi() permet de récupérer la référence d'un composant. Afin de pouvoir se connecter, il faut récupérer la référence de ce STBase, son nom est CommunityService.COMP_NAME.

```
m_comm =(CommunityService)m_session.getCompApi(CommunityService.COMP_NAME);
```

Il est alors possible de se connecter. Plusieurs méthodes de connexion sont disponibles (voir p19 du guide du développeur). L'instruction getCodeBase().getHost().toString() permet de récupérer le nom du serveur.

```
m_comm.loginByPassword(getCodeBase().getHost().toString(),
getParameter("loginName"),
getParameter("password"));
```

Tout comme nous avons prévu d'arrêter l'objet STSession, il faut au préalable se déconnecter du serveur.

```
m_comm.logout();
```

- Gestion des évènements

Lorsque la méthode de connexion au serveur se termine, il est possible que la connexion ne soit pas encore effective. Le processus de connexion peut prendre du temps. Afin d'être notifié de la connexion effective au serveur, il faut ajouter un listener de connexion. Cet ajout est possible en implémentant l'interface LoginListener.

```
m_comm.addLoginListener(this);
```

Pour traiter les évènements de connexion et déconnexion, il faut alors créer deux méthodes de gestion d'évènement (ou event-handler) : loggedIn() et loggedOut(). C'est dans ces méthodes que nous détaillerons les actions à effectuer lorsque la connexion ou la déconnexion sera effective. Elles seront activées dès qu'un processus de connexion ou déconnexion sera terminé.

L'utilisation des autres services du serveur de communauté est basée sur le même principe. Il suffit d'implémenter les interfaces associées aux services à utiliser, puis d'ajouter les listeners et de décrire les actions à réaliser dans les event-handlers.

6.1.2.2 Toolkit ConnectWebAPI

- Présentation

Le toolkit Connect Web API permet de réutiliser des services de base du client installé sur le poste de travail. Ainsi depuis une page web, il est possible par exemple d'ouvrir une fenêtre de messagerie instantanée du client Sametime de l'utilisateur. Cela impose donc que le client soit installé et démarré sur le poste local.

Le toolkit est stocké sur le répertoire \client\connectWebApi du SDK.

Le client Sametime repose sur Lotus Expeditor (tout comme Notes) qui lui-même est monté sur Eclipse RCP (voir Figure 55).

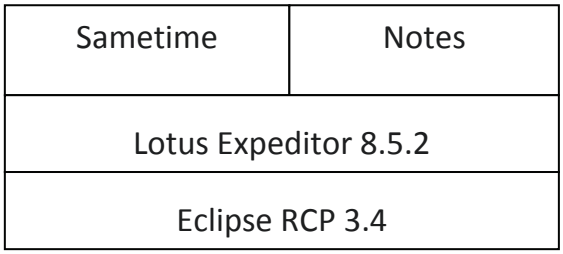


Figure 55 Architecture du client Sametime

Les services du toolkit sont gérés parle conteneur web[27] de la plateforme Lotus Expeditor sur le port 59449.

- Services proposés

Le toolkit propose les services suivants (voir tableau récapitulatif en annexe VIII) : gestion des contacts (récupération liste locale, ajout/suppression de groupe ou contact, renommer un contact, recherche d'un utilisateur de la communauté), envoi d'une alerte, ouverture d'une fenêtre de messagerie instantanée, gestion statut (récupération statut des contacts, mise à jour statut de l'utilisateur).

- Activation du toolkit sur le poste
    - Fichier plugin_customization.ini

Avant d'utiliser les fonctions du toolkit, il faut activer le conteneur web dans le fichier plugin_customization.ini qui se trouve dans le répertoire <Repertoire Notes>\framework\rcp\. La valeur de la ligne suivante doit être initialisée à 'true'.

```
com.ibm.collaboration.realtime.webapi/startWebContainer=true
```

Dans ce même fichier, il faut autoriser l'utilisation des fonctions que nous souhaitons utiliser. Nous pouvons le faire de deux manières :

- soit nous activons toutes les fonctionnalités en ajoutant la ligne suivante :

[27] Définition Jean-Marie Doudoux : Un conteneur web est un moteur de servlets qui prend en charge et gère les servlets : chargement de la servlet, gestion de son cycle de vie, passage des requêtes et des réponses ... (http://www.jmdoudoux.fr/java/dej/chap-servlets.htm)

com.ibm.collaboration.realtime.webapi.**enableAllWebApisOverride**=true

- soit une par une en précisant le nom de la fonction dans la ligne ajoutée, dans l'exemple ci-dessous, il s'agit la fonction addgroup.

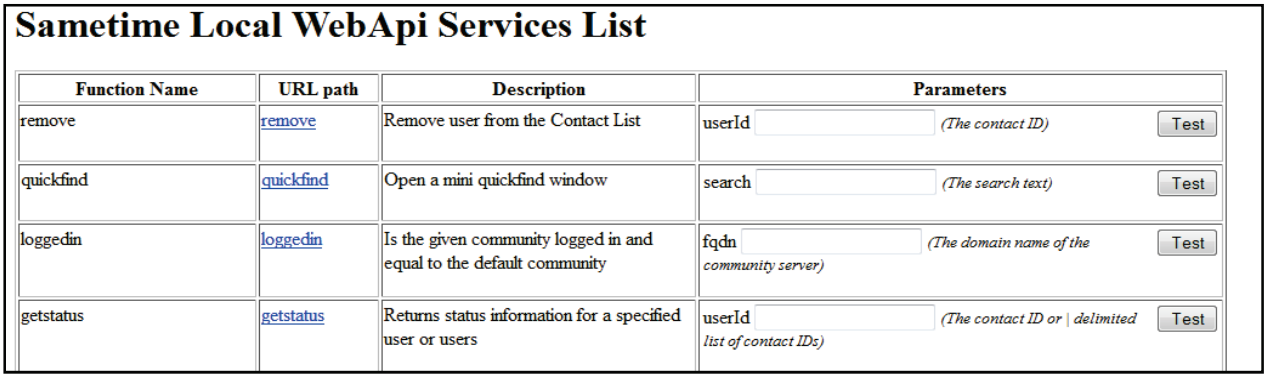
com.ibm.collaboration.realtime.webapi.**addgroup**Enabled=true

Si le fichier a été modifié, il faut redémarrer le client Sametime (ou Notes si le client est intégré) afin que les modifications soient bien prises en compte.

- Vérification du fonctionnement des services

Il est possible de vérifier le bon fonctionnement des services que nous souhaitons utiliser en consultant la page située à l'adresse suivante :

http://localhost:59449/stwebapi/listservices



Sametime Local WebApi Services List

Function Name	URL path	Description	Parameters
remove	remove	Remove user from the Contact List	userId *(The contact ID)* Test
quickfind	quickfind	Open a mini quickfind window	search *(The search text)* Test
loggedin	loggedin	Is the given community logged in and equal to the default community	fqdn *(The domain name of the community server)* Test
getstatus	getstatus	Returns status information for a specified user or users	userId *(The contact ID or | delimited list of contact IDs)* Test

Figure 56 Affichage de la page localhost:59449/stwebapi/listservices

- En cas de dysfonctionnement

Les consignes de mise en œuvre du toolkit sont allégées dans le cadre de notre projet car le poste est normalement déjà configuré pour fonctionner avec le toolkit. Si un dysfonctionnement apparaît après les modifications précédentes, il faut se référer à l'annexe du guide du développeur fourni dans le toolkit (ConnectWebApiDevguide.pdf présent dans le répertoire \doc du toolkit).

- Utilisation

Pour intégrer des fonctionnalités du toolkit dans une page web, il faut tout d'abord initialiser l'en-tête de la page:

```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Sametime Connect Web API: Contact Action Sample</title>

    <!-- Step #1: Import the ST Connect Web API Stylesheet -->
    <link rel="stylesheet" href="http://localhost:59449/stwebapi/main.css" type="text/css" />

    <!-- Step #2: Import the ST Connect Web API getStatus JavaScript include file -->
    <script type="text/javascript" src="http://localhost:59449/stwebapi/getStatus.js"></script>
</head>
```

Figure 57 En-tête HTML de la page web

Les fonctions JavaScript sont codées dans le fichier getStatus.js qui est stocké sur le conteneur web de la plateforme. Il faut inclure la feuille de style (http://localhost:59449/stwebapi/main.css) et le fichier JavaScript entre les balises <HEAD> et </HEAD> de la page web avec le code suivant :

```
<link rel="stylesheet" href="http://localhost:59449/stwebapi/main.css" type="text/css" />
<script type="text/javascript" src="http://localhost:59449/stwebapi/getStatus.js"></script>
```

L'appel aux fonctions se fera ensuite soit dans les balises html du corps de la page web, soit dans l'en-tête, codé en JavaScript à la suite d'une balise <script language="javascript">. Dans l'exemple suivant, le nom d'utilisateur apparaîtra avec une icône indiquant son statut à la place du texte « Resolving contact, please wait... » dès que la page web sera chargée et que la connexion avec le serveur de communauté sera effective. Ce code doit être inséré entre les balises <BODY> et <BODY> de la page et la balise <USER_ID> remplacée par l'identifiant du contact que nous souhaitons afficher.

```
<a class="awareness" userId="<USER_ID>" Resolving contact, please wait....</a>
```

Le toolkit fournit 5 fichiers html à titre d'exemple (dans le répertoire \samples du toolkit). Nous pouvons y retrouver l'exemple ci-dessus ainsi que l'utilisation de cette fonction codée en JavaScript dans l'en-tête.

6.1.2.3 IM Browser Toolkit

- Présentation

Le toolkit permet de donner un nouveau degré d'interaction aux applications web. Les utilisateurs pourront communiquer directement depuis ces applications avec d'autres utilisateurs connectés à Sametime. Il est présenté dans le document STBrowserSDK.pdf que nous pouvons trouver dans le répertoire client/proxy/doc du SDK.

Ce nouveau mode de communication entre les applications web et le serveur de communauté Sametime est possible grâce au serveur proxy (voir paragraphes 3.2.2 et 4.2.1).

- Architecture

Le toolkit se décline en trois niveaux.

  - API REST

La première couche propose des services REST pour communiquer avec le serveur proxy. Les API REST publient 100% des API du toolkit Java (voir paragraphe 6.1.2.1).

Les applications web peuvent communiquer avec un serveur web par le biais d'un protocole http. Celui-ci est synchrone alors qu'une messagerie instantanée fonctionne sur un mode asynchrone. Pour permettre l'utilisation de services asynchrones, le toolkit utilise un mécanisme de « long poll connexion » ou protocole Comet (voir Figure 58). Le principe est le suivant :

- le navigateur émet une requête AJAX(la « long poll connexion ») qui n'aboutit que lorsqu'un évènement se produit.
- lorsque le navigateur a besoin d'une information (par exemple, suivre le statut d'une personne),il l'envoie au serveur au travers d'une seconde requête et recevra les mises à jour de statut par la « long poll connexion ».

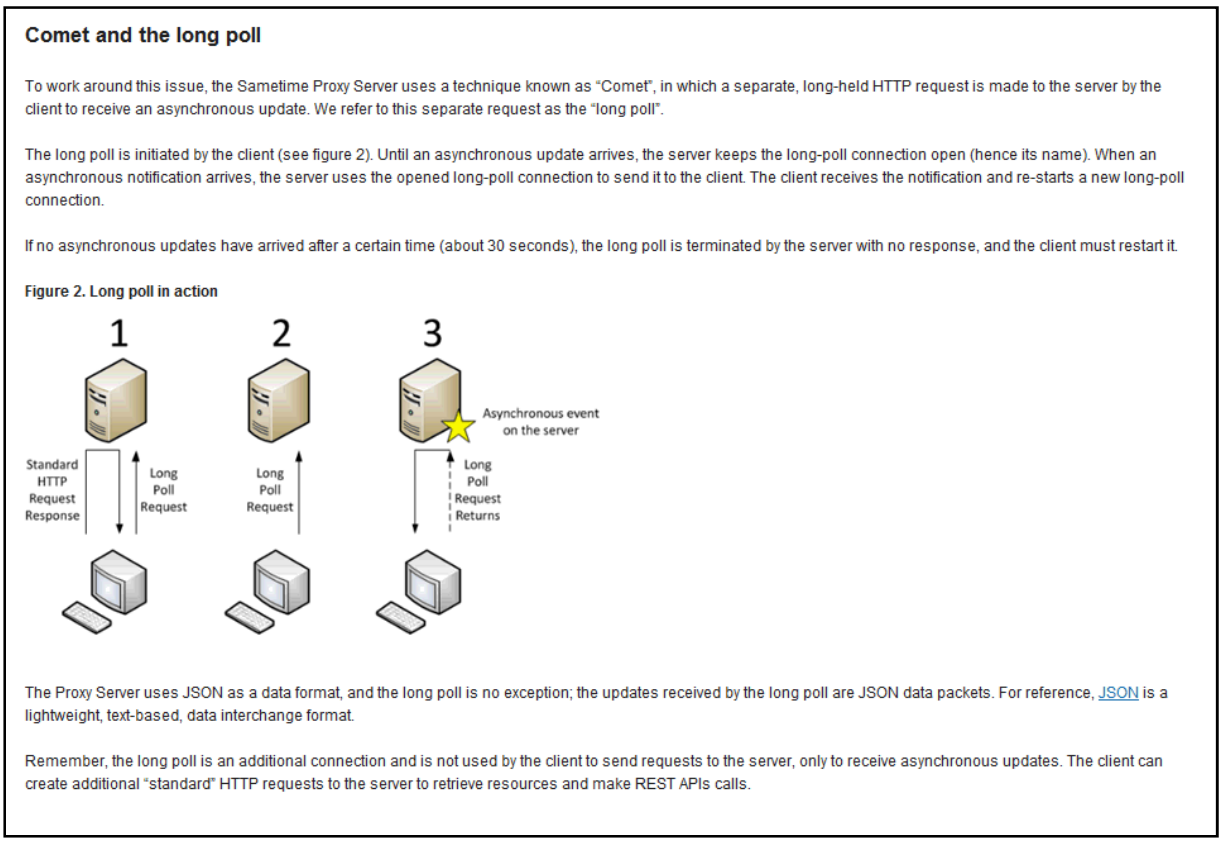


Figure 58 Mécanisme de long poll connexion[28]

Pour éviter les connexions/reconnexions intempestives au serveur de communauté (par exemple, au rechargement ou changement d'une page web), le serveur proxy attend un délai de 90 secondes après la fermeture de la « long poll connexion » avant de fermer la session qu'il a ouverte.

[28]Source : http://www-10.lotus.com/ldd/stwiki.nsf/dx/Integrating_Java_desktop_applications_with_IBM_Sametime_using_the_Sametime_Proxy_Server?OpenDocument&sa=true

Du fait de la complexité de la « long poll connexion », il n'est pas recommandé de la gérer directement avec les API REST. Cependant, le développeur hardi trouvera toutes les informations nécessaires à l'utilisation de ces API dans le chapitre 6 du guide du toolkit et les références de chaque service dans l'annexe A.

- API Base Component

Dans la seconde couche, le toolkit offre un ensemble d'API JavaScript (API Base Component) qui gère et couvre toutes les API REST. Elles permettent de masquer la difficulté de gestion de la « long poll connexion ».

Ces API utilisent OpenAjax Hub[29] pour faciliter l'usage de requêtes AJAX. OpenAjax Hub assure la « Javascript's same origin policy »[30]. Sans cela, l'appel d'une requête vers un autre serveur depuis une fenêtre html ouverte sur TOMCAT (serveur applicatif déployé dans le cadre de Watt) par exemple, ne fonctionne pas.

L'appel à OpenAjax Hub se fait via le fichier tunnel.html (fichier fourni avec le toolkit dans le répertoire \client\stproxy\samples du SDK) qui est dans le serveur de fichier statique avec les autres fichiers web développés.

En revanche, les API Base Component ne proposent aucune interface utilisateur.

- API User Interface

Cette couche fournit un ensemble de widget[31] à incorporer dans une application web. Elle va permettre de proposer des interfaces graphiques aux applications web comme l'intégration d'un client de messagerie instantané complet. Les widgets sont basés sur la technologie dojo[32], chacun encapsule une fonctionnalité Sametime.

- Socle de base

Les étapes de base à réaliser pour utiliser les fonctionnalités Sametime sont :

-Inclure les fichiers JavaScript et paramétrage de l'objet stproxyConfig
- Se connecter à Sametime
- Inclure l'appel aux fonctionnalités
- Se déconnecter de Sametime en sortant de l'application

[29] Implémentation de référence du modèle de programmation Ajax
[30] Spécifications du W3C (World Wide Web Consortium) pour restreindre la manière dont un document ou un script chargé depuis une origine peut interagir avec une autre ressource chargée depuis une autre origine. L'origine est définie par un même port, même protocole, même hôte. Pour de plus amples renseignements : https://developer.mozilla.org/fr/docs/Web/JavaScript/Same_origin_policy_for_JavaScript
[31] Définition wikipédia : En informatique, le mot widget recouvre deux notions distinctes, chacune en relation avec les interfaces graphiques. Il peut alors être considéré comme étant la contraction des termes window (fenêtre) et gadget. Il peut désigner : un composant d'interface graphique, un élément visuel d'une interface graphique (bouton, ascenseur, liste déroulante, etc.) ou bien un widget interactif, un petit outil qui permet d'obtenir des informations (météo, actualité, dictionnaire, carte routière, pense-bête – en anglais post-it –, traducteur, etc.).
[32]Dojo: Le toolkit dojo est un framework JavaScript open source qui facilite la création d'applications web riches (voir http://www.dojotoolkit.org).

```
<script type="text/javascript">
  var stproxyConfig = {
      server: "http://proxy.company.com:9080",
      tunnelURI: "http://www.company.com/tunnel.html",
      tokenlogin: true,        // if it's SSO, don't display login page
      isConnectClient: false,  // Prevent use of the rich client
      chat:   {
                  bringWindowToFront: true, // Force chat windows to front
                  partnerLeftMessage: true // Display when user leaves chat
              }
  }
</script>

<script type="text/javascript" src="/stbaseapi/baseComps.js"></script>
```

Figure 59 Configuration typique dans une page web

La Figure 59 présente la première étape de base que nous retrouverons lors de l'utilisation des API base components dans une page web. Le fichier JavaScript inclus est baseComps.js. L'objet stproxyConfig permet de définir les éléments de base permettant la relation entre l'application web et le serveur proxy (adresse serveur proxy, emplacement fichier tunnel.html…). Une présentation plus exhaustive de ces attributs est faite dans les pages 11 et 12 du guide du toolkit.

- Ajout d'interfaces graphiques

Pour utiliser les « API user interface », il faut inclure le fichier dojo.js en supplément puis selon les widgets souhaités :
-livename.js : si livename (visualisation du statut d'une personne) est la seule fonctionnalité requise
- widget.js : pour charger tous les widgets du toolkit

L'objet stproxyConfig va nécessiter un attribut supplémentaire : plugins. Celui-ci contient une liste de plugins à activer ou désactiver selon les besoins de l'application. Par défaut, tous sont activés. La liste des plugins est présentée dans les pages 55 et 56 du guide du toolkit.

Pour assurer l'affichage correct des interfaces du toolkit dojo, il faut ajouter la classe « stbody tundra » dans la balise <Body> (voir Figure 60).

```
<body class="stbody tundra" >

  <div class="stmain">
     <div id="myChat" style="width: 100%; height: 400px; display: none;" ></div>
        <input style="display:none;" type="button" id="closeButton" value="Close Chat" onclick="closeChat();" />
  </div>

</body>
```

Figure 60 Exemple d'utilisation de widget dojo

L'insertion de widget se fera ensuite soit par l'utilisation de balise html <div dojotype= « …»> ou par la programmation de l'objet en JavaScript comme l'objet « myChat » dans l'exemple de la Figure 60, qui est défini dans le corps html de la page comme

un objet Div ayant pour identifiant « myChat » mais qui sera créé dans l'entête du fichier avec le code suivant :

```
dojo.byId("myChat").appendChild(newComponent.domNode);
```

Les noms et les paramètres des widgets dojo sont détaillés dans le chapitre 4 du guide du toolkit.

Cette couche du toolkit permet aussi de modifier l'apparence de chaque widget ou composant du widget. Les démarches à suivre sont développées dans le chapitre 5 du guide.

6.1.2.4 Multisession : quelques limites…

Il est possible pour un utilisateur d'être connecté au serveur de communauté plusieurs fois depuis différents endroits, par exemple : un client ouvert dans Notes, un autre ouvert dans une application web et une session initiée avec le toolkit java sur une application quelconque. Cependant, si le serveur de communauté reçoit une demande de communication avec cet utilisateur, le choix de l'application depuis laquelle sera ouverte la fenêtre de communication est aléatoire. Il n'existe pas de possibilité de privilégier une application plutôt qu'une autre.

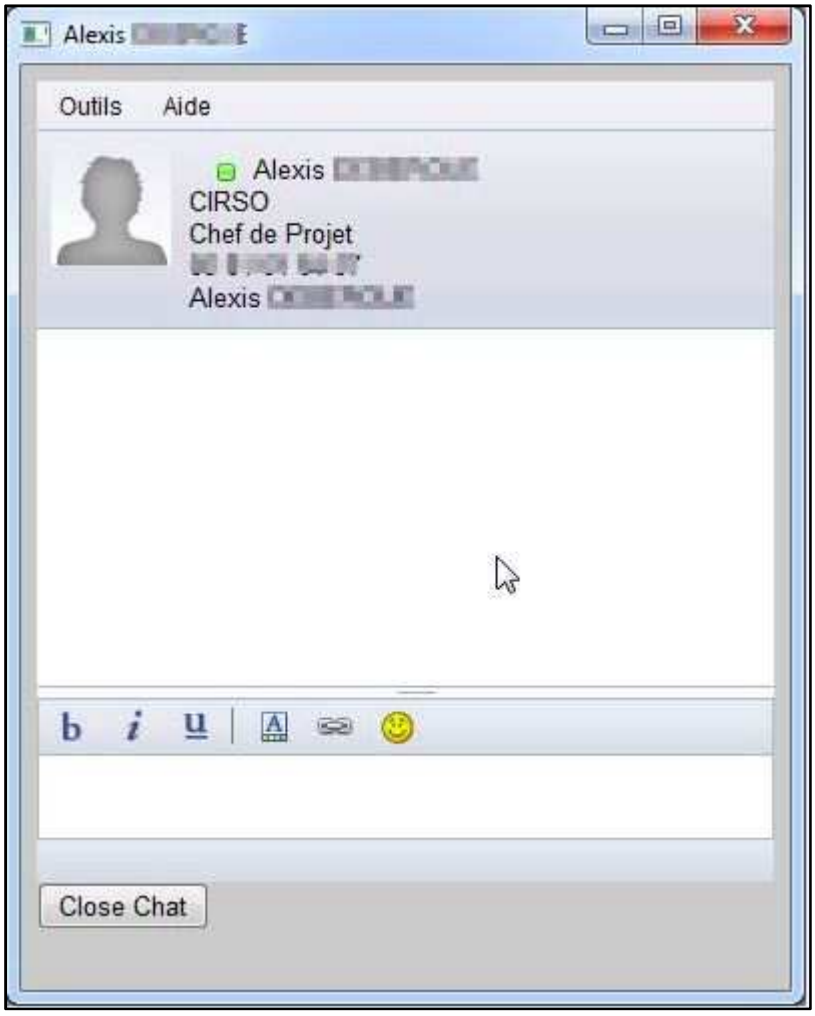


Figure 61 Apparence fenêtre de communication toolkit Browser IM

Si la fenêtre est ouverte depuis Notes ou une application web, cela ne pose pas trop de problème car l'apparence de la fenêtre de communication de chacun est similaire. Par contre, si le serveur de communauté choisit la connexion ouverte depuis le toolkit java, la fenêtre de communication aura une apparence inhabituelle et désuète.

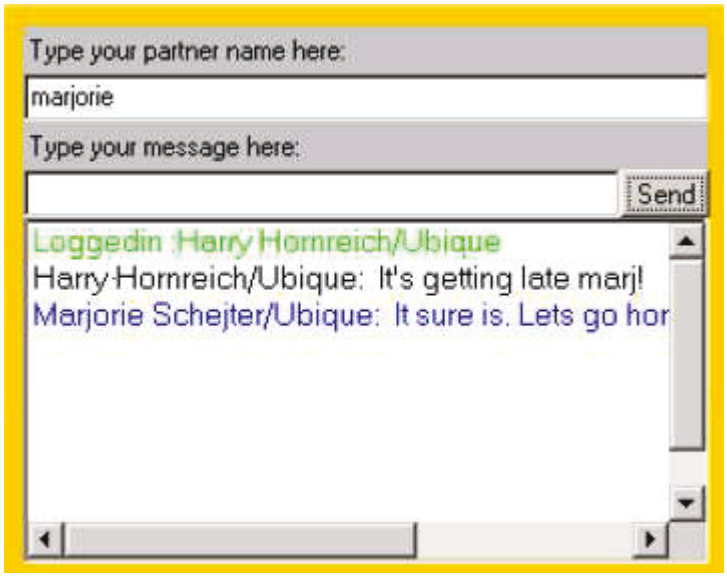


Figure 62 Apparence fenêtre de communication toolkit Java

6.1.3 Etudes des toolkits

A partir de nos connaissances sur les toolkits, nous avons étudiés les avantages et inconvénients de chaque partie à réaliser selon le toolkit mis en œuvre.

6.1.3.1 Afficher client Sametime dans espace dédié

Le toolkit ConnectWebApi n'a pas été étudié dans le cadre de l'affichage du client Sametime dans l'espace dédié car il n'offre pas cette fonctionnalité.

- Utilisation toolkit Java

Les interfaces utilisateurs développées dans le cadre de Watt tout comme pour toutes les applications métiers accessibles depuis le portail Harmonie sont en SWT[33].Le toolkit java propose des interfaces SWING[33].

Deux solutions :

- soit nous utilisons les interfaces du toolkit mais nous obtenons une apparence des fenêtres très désuète (voir Figure 62).

[33] Bibliothèques graphiques pour le langage Java

- soit nous développons nos propres interfaces, ce qui impose un travail conséquent dans le cadre d'un client de messagerie instantanée

- Utilisation toolkit IM Browser

Dans Eclipse, il est possible d'intégrer un navigateur dans une vue. L'utilisation du toolkit IM Browser peut donc se faire en chargeant dans le navigateur une page html qui lance le toolkit. Cette page se chargerait ensuite d'afficher un client Sametime web complet. L'apparence du client web proposé par le toolkit est beaucoup plus en adéquation avec l'apparence actuelle du client intégré dans Notes.

6.1.3.2 Intégration présentiel dans un objet SWT

Le deuxième besoin est de permettre à l'utilisateur de connaitre l'information de présence des acteurs dont le nom/prénom apparaît dans les fenêtres de travail Watt. Dans Watt (et dans les autres applications du portail Harmonie), les couples nom/prénom apparaissent dans des objets SWT.

- Utilisation toolkit Connect Web API

L'utilisation du toolkit ConnectWebAPI depuis Eclipse peut se faire dans un navigateur. Pour connaitre le statut d'un utilisateur, il suffit de faire un appel REST depuis le navigateur vers le client Sametime intégré. Cet appel se fait au moyen d'une requête http, il n'utilise pas le mécanisme de « long poll connexion » comme le toolkit IM browser. De ce fait, le client renvoie l'état de l'utilisateur au moment de la requête. L'évolution de son statut n'est pas connue sauf si d'autres requêtes http sont faites régulièrement. La connaissance de l'information n'est donc pas simultanée au changement de statut. Cependant, la requête se fait vers un serveur local, elle a donc l'avantage d'être rapide.

Dans le cadre de l'affichage du présentiel au niveau de la corbeille d'affaires, ce scénario est envisageable car la corbeille est rafraîchie toutes les 30 secondes. Si elle contient 1000 lignes, 1000 requêtes http sont envoyées au serveur local. Cette hypothèse ne pose pas de problème de rapidité dans sa mise en œuvre et est sans contrainte sur le réseau ni sur le serveur puisque les requêtes sont faites en local. Elle peut être gérée dans un thread en arrière-plan. Par contre, pour l'utilisation de cette fonctionnalité à d'autres endroits de Watt, où il n'est prévu aucun rafraîchissement, il faudra prévoir un système de relance régulier des requêtes au serveur local.

Par ailleurs, cette solution implique que le client soit installé sur le poste et connecté au serveur de communauté.

- Utilisation toolkit Java

Comme nous l'avons évoqué précédemment, la réception d'une communication instantanée est conflictuelle lorsque plusieurs sessions sont simultanément en cours. Dans le cas où nous utiliserions le toolkit Java pour afficher le présentiel nous pourrions avoir jusqu'à 3 sessions ouvertes simultanément sur le serveur de communauté:

- Une au travers du serveur proxy pour l'espace dédié
- Une au travers du client intégré dans notes
- Et celle-ci au travers du portail harmonie ou de Watt via les API Java

Pour répondre aux demandes de communication instantanée, le serveur de communauté utilisera aléatoirement une des 3 sessions. Si c'est la connexion Proxy ou celle du client embarqué dans Notes, alors la cohérence d'IHM est assurée. Par contre si c'est celle du toolkit java, une fenêtre SWING va être ouverte, or celle-ci est complètement démodée et incomplète.

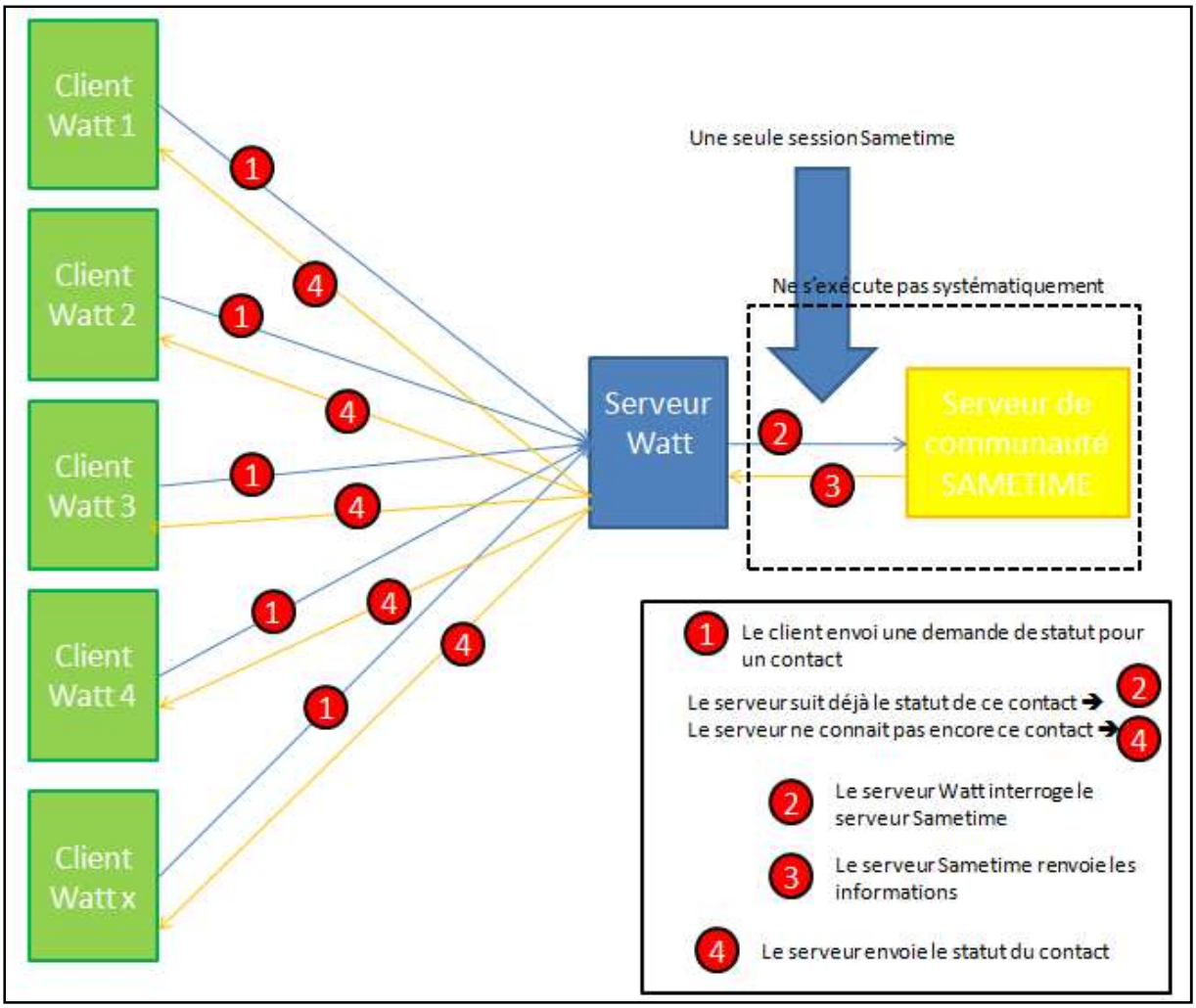


Figure 63 Utilisation toolkit Java côté serveur

Pour contourner cette problématique, si notre choix se porte sur l'utilisation du toolkit Java pour implémenter le présentiel dans les objets SWT de Watt, une solution serait de créer une session dans Eclipse côté serveur avec un utilisateur système (voir Figure 63). C'est-à-dire que

les services seraient proposés par le serveur, dans ce cas chaque client pourrait invoquer ces services en s'adressant au serveur. Le serveur, lui, nécessiterait donc d'une seule connexion au serveur de communauté. Ainsi pas de risque d'ouverture d'une fenêtre SWING sur un client et une seule connexion qui traite le présentiel.

- Utilisation toolkit IM Browser

La fonctionnalité de présentiel est à mettre en œuvre sur un contrôle SWT. Ce qui consiste à remplacer le contrôle qui affiche le nom par un objet de type navigateur chargé d'afficher une page HTML permettant de visualiser le statut de l'utilisateur. Dans le cadre de l'affichage du statut des utilisateurs dans la corbeille d'affaire, il faut instancier un nouveau navigateur pour chaque ligne, donc s'il y a 1000 lignes il faudra instancier 1000 navigateurs, soit 1000 requêtes actives vers le serveur proxy. Cette solution n'est pas convenable et demande une sollicitation trop importante du serveur proxy, donc peut entrainer des perturbations au niveau du réseau et de la rapidité des réponses.

6.1.3.3 Lancement communication instantanée depuis couple nom/prénom

- Utilisation toolkit Java

Comme nous l'avons vu précédemment, l'interface proposée par ce toolkit pour les fenêtres de communication instantanée est obsolète et démodée. Nous ne retiendrons donc pas cette solution.

- Utilisation toolkit IM Browser

La mise en œuvre du toolkit IM Browser consiste à ouvrir une nouvelle fenêtre de type navigateur. Celui-ci pointe vers une page html qui charge le toolkit et affiche le widget de chat.

- Utilisation toolkit ConnectWebAPI

Comme pour le toolkit IM Browser, l'utilisation de ce toolkit consiste à ouvrir une nouvelle fenêtre de type navigateur. La page html contenue dans celui-ci se charge d'envoyer une requête locale vers le client embarqué dans Notes. Cette solution est cependant limitée, car elle ne peut fonctionner que si l'utilisateur a ouvert son client local et s'est connecté au serveur de communauté.

6.1.4 Choix des toolkits

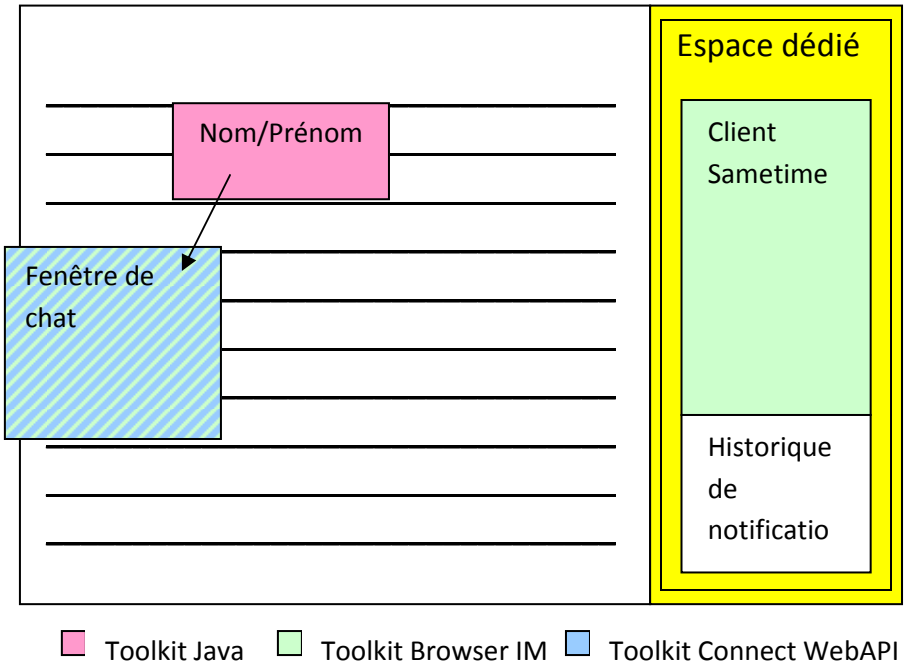


Figure 64 Choix des toolkits

Comme nous avons pu le voir au travers de l'étude des toolkits en fonction des besoins à mettre en œuvre, le **toolkit java** n'est pas intéressant à utiliser d'un point de vue graphique. Cependant pour **récupérer les informations de présentiel** dans un couple nom/prénom c'est celui-ci que nous choisirons car ne nécessite pas forcément d'interface graphique. Dans ce cas-là, nous pourrons attribuer à l'objet SWT une icône (enregistrée en local) en fonction du résultat retournée par la toolkit. Les deux autres toolkit présentent des limitations, pour le premier il y a des risques de ralentissement du réseau et pour le second le statut ne peut pas être récupéré si l'utilisateur n'a pas connecté son client embarqué dans Notes.

Pour la mise en œuvre de **l'ouverture d'une fenêtre de communication** depuis un couple nom/prénom les deux toolkit ConnectWebAPI et Browser IM sont envisageables. Cependant nous choisirons le **toolkit Browser IM** car ne nécessite pas d'avoir un client local déjà connecté.

Enfin, pour **l'implémentation du client Sametime dans l'espace dédié** de Watt nous retenons **le toolkit Browser Im**. Il propose les mêmes fonctionnalités que le toolkit java mais l'apparence du client est plus moderne.

6.2 Sécurité –Authentification

6.2.1 Possibilités de connexion

L'ouverture d'une session Sametime, se fait par le biais d'une fonction de login. Le toolkit proxy en propose quatre :

- loginByPassword : cette fonction permet de se connecter en renseignant directement en dur dans le code de la page web le nom de l'utilisateur et son mot de passe.

- loginByToken va utiliser un token LTPA par le biais d'un cookie SSO soit disponible
- loginAsAnon permet une authentification anonyme, cette option est désactivée sur le serveur de communauté des Urssaf
- loginWithToken propose une authentification en passant le token LTPA en paramètre par exemple dans l'URL de la page html.

6.2.2 Fonctionnement actuel d'authentification

L'ouverture d'une session Sametime, quel que soit le toolkit utilisé, se fait soit par le biais d'un token LTPA soit par les identifiants et mot de passe de Notes.

L'accès à Watt se fait par le portail Harmonie. Les identifiants pour se connecter au portail sont les identifiants Anaïs[34].L'ouverture de la session Sametime dans Watt ne peut donc pas se faire par les identifiants puisque ce ne sont pas les mêmes.

L'authentification depuis Watt se fera donc par token LTPA. Une solution existe depuis l'intranet informatif de la branche recouvrement (Illiad[35]) qui utilise les identifiants Anaïs : la solution d'authentification unifiée Illiad (voir Figure 65). Celle-ci consiste à permettre à permettre à des utilisateurs Illiad d'accéder à des données hébergées sur un serveur WebDomino sans avoir à se réauthentifier.

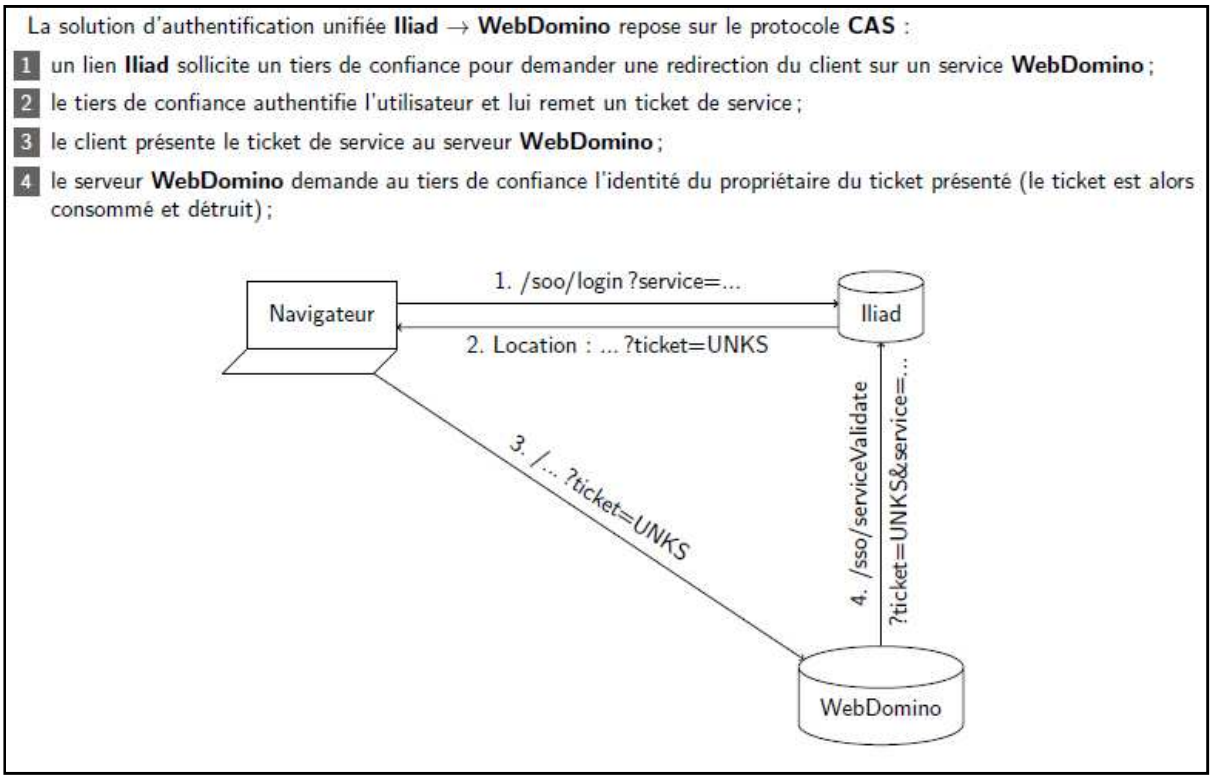


Figure 65 Solution d'authentification unifiée Illiad

[34] Annuaire Anaïs : annuaire des agents mis en place par la branche recouvrement au cœur de son système d'information pour élever le niveau de sécurité du contrôle d'accès à ses applications.
[35] Illiad : intranet informatif de la branche recouvremet

Cette solution est basée sur un protocole CAS[36]:

- on se sert des identifiants ANAIS pour demander à Illiad de nous fournir un ticket CAS.

- On se sert du ticket CAS pour demander à un serveur Domino de nous fournir un token LTPA

Et à terme, il faudrait envisager de mettre en place un protocole CAS sur le serveur Harmonie et utiliser la fonction loginWithToken vue dans le chapitre précédent pour passer le token LTPA en paramètre et permettre la connexion.

Cette solution n'est pas testable actuellement pour des raisons d'infrastructures. Le serveur proxy n'est pas sur le même domaine que le serveur de communauté. Ils ne peuvent donc pas échanger de token LTPA. Il faudra donc attendre le déploiement des serveurs proxy et de communauté en production pour tester l'utilisation du token LTPA lors de l'authentification à Sametime.

6.3 Mise en œuvre

6.3.1 POC

6.3.1.1 Principe

Comme nous l'avons vu dans les chapitres précédents, l'infrastructure adéquate aux besoins du projet n'est pas totalement en place (serveur proxy non déployé en production et sur un domaine différent de celui du serveur de communauté). De plus, arrivée à ce stade du projet, il ne me restait plus beaucoup de temps de présence dans l'entreprise. Pas suffisamment pour mener la réalisation du projet à terme.

Il a donc été décidé avec les équipes que je réaliserais un POC (ou Proof Of Concept) qui nous permettrait de valider la faisabilité du projet.

Pour se faire, j'ai développé dans l'environnement Eclipse, une mini application me permettant de simuler l'utilisation d'un plugin proposant les fonctionnalités nom_prénom, l'affichage du statut des utilisateurs ainsi que l'affichage d'un client web. Cette mini application est découpée en deux fenêtres (voir Figure 66) : une avec une liste d'utilisateurs pour tester l'affichage du statut et la communication depuis le nom_prénom et l'autre embarquant le client web.

Le périmètre du POC a été défini dans un document ainsi qu'un compte rendu[37] lorsque celui-ci a été réalisé. Le compte rendu et le code source réalisé peuvent ainsi servir de support à la rédaction des spécifications et aux développements.

[36] Protocole CAS : système d'authentification unique pour le web
[37] Document rédigé : « Note technique - Intégration Sametime dans Watt.doc »

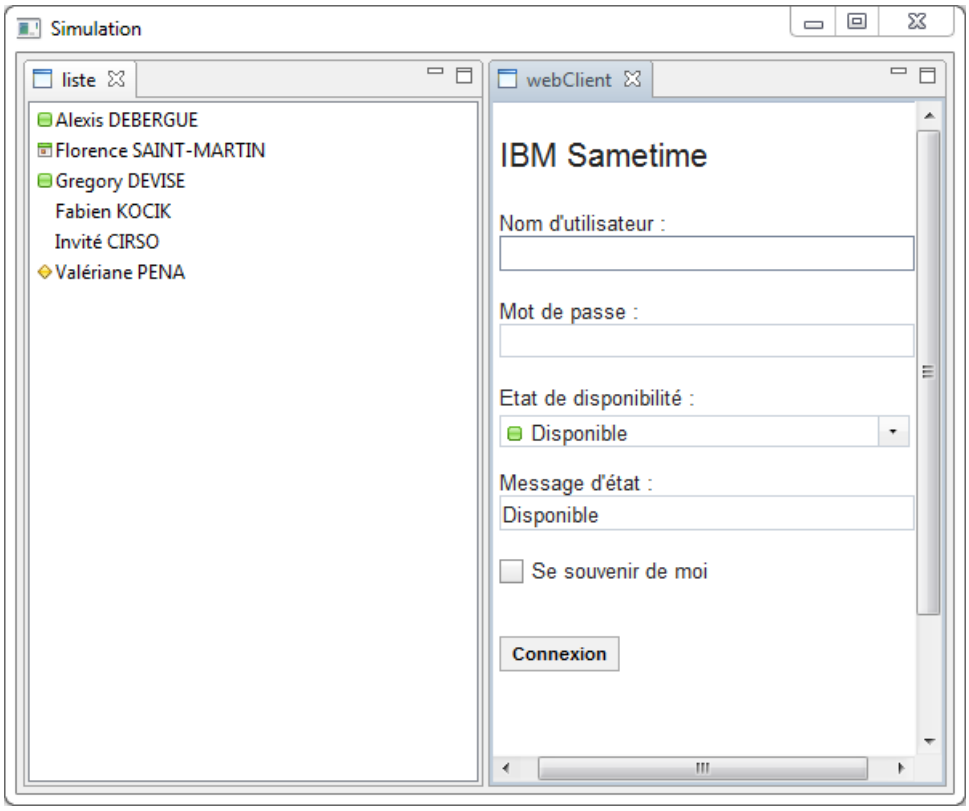


Figure 66 Application de simulation des fonctionnalités

6.3.1.2 Composition du POC

Le POC a été réalisé sous la forme d'un plugin (**fr.recouv.collaboratif**, voir Figure 67) et d'une application (**fr.recouv.miniApplicationDecouverte,** voir 6.3.1.3) permettant de simuler ses services.

Le plugin propose les services ci-dessous, leur réalisation est détaillée dans les chapitres suivants :

- affichage du client web dans une perspective (voir 6.3.1.4)
- communiquer avec une personne à partir d'un menu déroulant depuis le couple nom_prénom de la personne (voir 6.3.1.5)
- affichage du statut de la personne (voir 6.3.1.6)

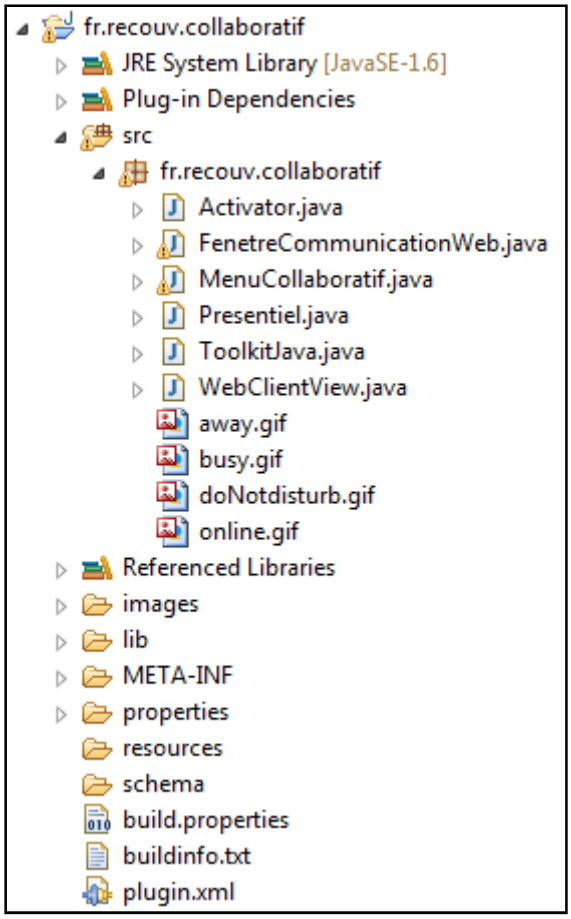


Figure 67 Composition du plugin

L'affichage d'une fenêtre de communication est réalisé avec le toolkit Browser IM. Les fichiers html développés dans ce cadre ainsi que le fichier **tunnel.html** (fichier fourni avec le toolkit, voir 6.1.2.3) sont stockés sous :

http://cer31-dvlpnots2.cer31.recouv/public/AdminNotes/connectim.nsf/

Les chapitres suivants s'attachent à détailler rapidement la mise en œuvre des fonctionnalités. Une description plus précise et technique est disponible dans le compte rendu du POC.

6.3.1.3 Application

L'application simulant est constituée d'une perspective qui affiche deux vues. Chacune va afficher la liste de couples nom_prénom et le client web. Afin d'utiliser les fonctionnalités des couples nom_prénom, ceux ci sont intégrés dans des tableitem.

Le client web est affiché par le biais de la classe WebClientView du plugin.

Un clic droit sur un couple nom_prénom de la liste fait apparaître le menu « Communiquer avec » (voir Figure 67). Celui-ci est proposé par la classe MenuCollaboratif du plugin.

Figure 68 Menu "Communiquer avec"

L'affichage de l'icône qui apparaît à droite des couples nom_prénom et permet de connaître le statut de l'utilisateur fait appel à la fonction affichePresentiel de la classe Presentiel du plugin.

6.3.1.4 Client web

Pour intégrer le client web dans une vue Eclipse il a été décidé d'utiliser le toolkit IM Toolkit Browser du SDK. Son implémentation est développée dans la classe WebClientView du plugin.

L'affichage du client dans Eclipse va se faire dans une fenêtre de type browser qui appellera le fichier html dans lequel est développé le client.

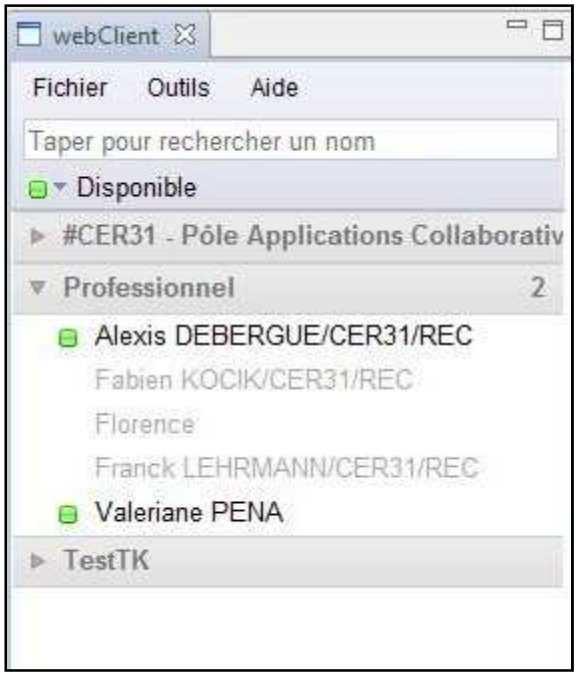


Figure 69 Apparence du client web

Dans le cadre du POC, le fichier web utilisé pour afficher le client web est celui présent sur le serveur proxy :

```
http://dev31pacint8.cer31.recouv:9080/stwebclient/popup.jsp
```

6.3.1.5 Ouverture fenêtre de communication

Le menu « Communiquer avec » fonctionne avec un listener. La sélection de ce libellé provoque l'appel à la classe FenetreCommunicationWeb. Cette classe va permettre la création d'une fenêtre de communication avec l'interlocuteur identifié dans le couple nom_prénom pour lequel l'option a été sélectionnée.

Comme pour le client web, la réalisation de la fenêtre de communication est faite avec le toolkit IM Browser. Le fichier html de création de la fenêtre de communication est example7.html.

L'appel au fichier html se fait avec passage en paramètre de la personne à contacter. Son identifiant est récupéré grâce au texte du tableitem du couple nom_prénom. Ce passage de paramètre se fait de la façon suivante pour un fichier hébergé sur un serveur Domino.

```
example7.html?OpenFileResource&userid=<id_utilisateur_à_contacter>
```

La réalisation de cette fonctionnalité a soulevé plusieurs problèmes (voir compte rendu POC). Certains ont pu être résolus après des recherches auprès de différents supports (notamment le guide du toolkit et internet). Cependant, il en persiste un assez important : après l'ouverture de la fenêtre de communication, si nous écrivons un message et que notre interlocuteur nous répond, sa réponse apparaît dans une nouvelle fenêtre. La première fenêtre créée initialement devient inactive mais la deuxième fonctionne encore si la conversation continue.

Mon stage arrivant à la fin, je n'ai pas eu suffisamment de temps pour le résoudre. La résolution de ce problème passera par une étude approfondie des fonctions chat de la bibliothèque baseComp.js du toolkit.

6.3.1.6 Présentiel

La mise en place du présentiel est faite avec le toolkit java.

Une classe **ToolkitJava.java** a été créée pour permettre de gérer une session Sametime avec l'utilisateur Automate Watt (utilisateur système retenu pour éviter l'affichage aléatoire d'une fenêtre de communication avec ce toolkit voir 6.1.3.2) et de récupérer le statut d'un utilisateur. Elle va initialiser les sessions et les listener pour le présentiel.

La classe propose une fonction `public STUserStatus recupUseStatut(String user)`.Elle va permettre de récupérer le statut d'un utilisateur. Elle utilise les services de listener AwarenessServiceListener, ResolveListener et StatusListener du toolkit (voir compte rendu du POC et guide de développement du toolkit pour plus d'informations).

Le plugin met aussi à disposition des icônes pour matérialiser le statut d'un utilisateur. Elles sont stockées dans le répertoire source du plugin. La classe **presentiel.java** va gérer ces icônes et proposer une fonction `public void` affichePresentiel(TableItem item) qui va attribuer l'icône correspond au statut de l'utilisateur passé en paramètre.

6.3.2 Spécifications techniques

6.3.2.1 Diagramme de classe

Pour pouvoir illustrer les spécifications techniques, j'ai élaboré le diagramme de classe suivant. Il a été fait à partir des diagrammes de cas d'utilisation (cf. paragraphe 5.3).

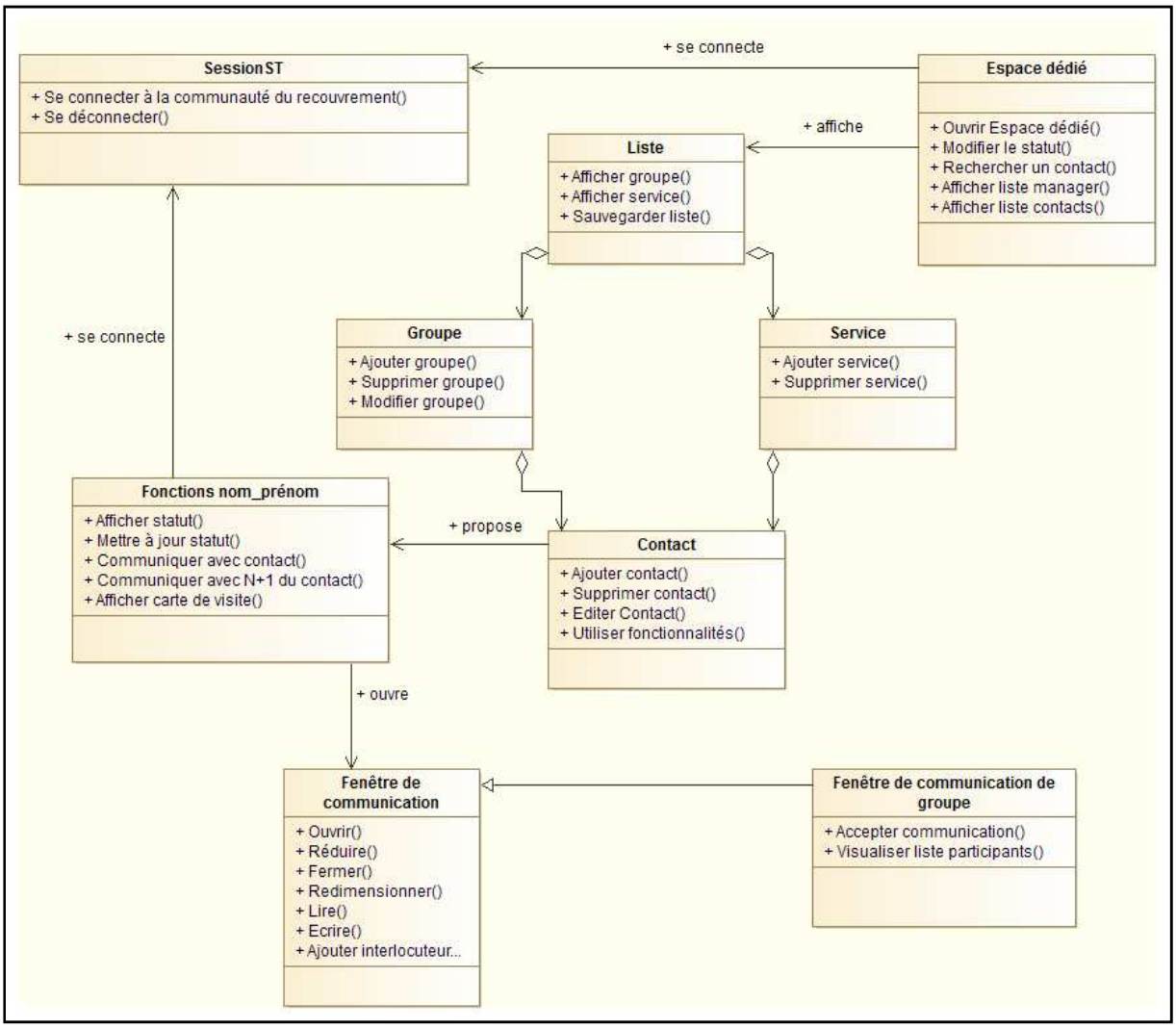


Figure 70 Diagramme de classe

De ce diagramme se dégagent 9 classes :

- **SessionST** pour gérer la connexion avec le serveur de communauté
- **Espace dédié** qui permet d'afficher les listes de contacts, de modifier son statut ou de rechercher un contact
- **Liste** qui est affichée dans l'espace dédié et peut contenir des groupes et/ou des services
- **Groupe**, il est créé par l'utilisateur pour organiser son espace dédié. Il peut donc être ajouté, modifié ou supprimé de la liste de contacts. Il est composé de contacts.

- **Service**, consiste en un ensemble de contacts appartenant à un même service. L'utilisateur peut en ajouter ou en supprimer dans sa liste. Le service est composé de contacts.

- **Contact** qui peut être ajouté, supprimé ou modifié dans un groupe. Il est aussi possible d'accéder aux fonctionnalités nom_prénom depuis le contact.

- **Fonctions nom_prénom** qui gère toutes les fonctionnalités accessibles depuis un couple nom_prénom comme afficher le statut, mettre à jour le statut, communiquer avec le contact, communiquer avec le N+1 du contact ou afficher sa carte de visite.

- **Fenêtre de communication** qui regroupe toutes les caractéristiques qui permettent d'interagir sur la forme de la fenêtre (ouvrir, réduire, fermer, redimensionner) ainsi que son contenu (lire la conversation, écrire, ajouter un interlocuteur).

- **Fenêtre de communication de groupe** qui est une extension de la fenêtre de communication. Elle hérite donc de toutes ses méthodes et en propose deux supplémentaires qui est d'accepter l'ouverture de la fenêtre lors de la réception d'une invitation à la communication et de visualiser la liste des participants.

6.3.2.2 Plugin

Pour permettre la réutilisation du projet par d'autres applications que Watt, il sera réalisé sous forme de plugin nommé : **fr.recouv.collaboratif**. L'étude fait apparaître plusieurs catégories. Le plugin sera donc divisé et développé en plusieurs parties.

Les modules sous jacents sont :

- Client web : **fr.recouv.collaboratif.clientWeb** que j'ai décrit dans le fichier de spécifications techniques WATT2-ConceptionClientWeb-V1.0.docx

- Fonctionnalités nom prénom : **fr.recouv.collaboratif.foncNomPrenom** que j'ai décrit dans le fichier de spécifications techniques WATT2-Conception foncNomPrenom - V1.0.docx

- Présentiel : **fr.recouv.collaboratif.presentiel**, ce package se découpe en deux parties une côté client et l'autre côté serveur, il est décrit dans le fichier de spécifications techniques WATT2-Conception présentiel-V1.0.docx

J'ai créé un document principal de spécifications technique (WATT2-ConceptionImplémentationFonctionnalitesCollGeneralités-V1.0.docx) dans lequel j'ai synthétisé les choix qui avaient été fait pour l'utilisation des toolkits ainsi que les actions à mettre en œuvre pour les utiliser. Ce document identifie les sous modules cités précédemment et détaille le plugin général.

Des opérations ont été identifiées comme communes à chacune des parties : la récupération du token LTPA et celle de l'identifiant unique du contact dans l'annuaire LDAP du

recouvrement. Celles-ci seront donc implémentées dans le package principal du plugin. Elles sont aussi décrites dans le document de spécifications techniques général.

6.3.2.3 Structure des documents de spécifications

Chaque document est rédigé selon le même schéma. Le schéma de base est celui préconisé par la méthodologie CAIRN. L'objectif de ces documents est de décrire la conception cible au niveau logique et logiciel et de spécifier les composants à développer. Il permet de capitaliser le travail de conception. Je remarque après coup que les exigences fonctionnelles n'ont pas été redispatchées en exigences techniques. Seul le diagramme de classe fait un lien les exigences fonctionnelles et les spécifications techniques. Ce manque entraine un risque de perte de fonctionnalités au niveau de la réalisation. Les tests d'intégration permettront de vérifier ce point (cf. paragraphe 5.1.3).

Un premier chapitre resitue le contexte en guise d'introduction. Il est donc identique pour les trois documents qui spécifient le client web, les fonctionnalités nom_prénom et le présentiel. Je résume en quelques lignes le projet global, illustre avec le diagramme de classe complet et renvoie vers un lien du document général de spécifications techniques.

Le deuxième chapitre sert de glossaire et de liste d'abréviations

Le troisième chapitre brosse un descriptif de la partie spécifiée dans le document et répertorie les différentes références associées à cette partie. Au niveau du descriptif, je rappelle en quoi consiste cette partie par rapport au projet. Puis je détaille comment j'ai prévu de le réaliser : nom des packages à rajouter au plugin, toolkit choisi, fichier généré et brève description de sa composition. La partie référence renvoie donc vers les spécifications fonctionnelles du document des cas d'utilisations, vers les différents documents utiles à l'utilisation du toolkit choisi et vers les analyses faites en amont au niveau du POC et de l'étude du toolkit. Elle permet en un coup d'œil de retrouver le document dans lequel se trouveront les informations pertinentes pour comprendre ou réaliser la suite du module.

Les derniers chapitres décrivent les étapes de réalisation.
Dans le cadre des modules réalisés avec le toolkit IM Browser, j'ai découpé en trois parties. La première où je détaille les composants à réaliser avec le toolkit, dans la seconde j'expose comment ces composants sont intégrés au niveau du plugin et enfin dans la dernière je dresse les consignes pour insérer ces fonctionnalités dans Watt.
En revanche, pour le module détaillant le présentiel, j'ai ventilé cette section sur trois parties : une qui décrit les développements côté client, une autre côté serveur. La dernière porte sur la mise en œuvre de la communication entre les deux. Les paragraphes suivants s'attachent à synthétiser cette partie pour chacun des documents rédigés.

6.3.2.4 Client web

Pour intégrer le client web à Watt, il a été décidé d'utiliser le toolkit IM Browser fourni par le SDK de Sametime.

Comme expliqué précédemment, cette réalisation est codée dans le package **fr.recouv.collaboratif.clientWeb** qui est intégré au plugin du projet fr.recouv.collaboratif.

Le client web couvre toutes les caractéristiques du diagramme de classe du projet global. Il est développé dans un fichier html **clientWeb.html** à partir du toolkit IM Browser. La spécification de ce fichier est disponible dans l'annexe IX.

Pour utiliser ce client depuis le plugin développé sous Eclipse, il faut créer la vue **WebClientView**. Celle-ci doit permettre d'ouvrir une fenêtre de type browser qui affiche le ficher html créé précédemment. Le pseudo code de cette vue est exposé dans le document de spécification.
La connexion automatique à Sametime se fait par le biais d'un token LTPA. La récupération de celui-ci est faite dans la fonction recupTokenLTPA présente dans le package principal du plugin.

Pour insérer cette vue dans Watt, il est nécessaire de créer une perspective. Les instructions pour y parvenir sont écrites dans le document de spécification

6.3.2.5 Fonctionnalités nom_prénom (hors présentiel)

Pour implémenter les fonctionnalités nom_prénom dans Watt, il a été décidé d'utiliser le toolkit IM Browser fourni par le SDK de Sametime.

Cette réalisation est codée dans le package **fr.recouv.collaboratif.foncNomPrenom** qui est intégré au plugin du projet **fr.recouv.collaboratif**.

Pour rappel, dans Watt le besoin est de permettre, à partir d'un couple nom_prénom (présents dans la colonne acteur des corbeilles d'affaires ou dans les caractéristiques d'une affaire dans Watt traitement), d'accéder à des fonctionnalités de collaboration. Ces fonctionnalités sont :

- ✓ Communiquer avec le contact
- ✓ Communiquer avec le N+1 du contact
- ✓ Afficher la carte de visite du contact

Ce module couvre une partie de la classe « Fonctions nom_prénom » du diagramme de classe global. Les fonctions « Afficher statut » et « Mettre à jour statut » seront développées dans la partie « Présentiel ».

- Utilisation du toolkit IM Browser

La mise en œuvre des fonctionnalités nécessaire requiert la création de trois fichiers html. Ces fichiers comportent une partie commune qui est détaillée dans le document de spécifications techniques. Puis chacun se spécifie en fonction de la propriété à concevoir.

Le fichier **comPointAPoint.html** va afficher une fenêtre de communication de type point à point, c'est-à-dire avec un seul interlocuteur. Cette fenêtre est contenue dans l'objet Chat du toolkit.

Le fichier **comMultiple.html** va afficher une fenêtre de communication multiple, c'est-à-dire avec plusieurs interlocuteurs. Cette fenêtre servira dans le cas où la fonctionnalité choisie est « Communiquer avec le N+1 » et que l'acteur a plusieurs supérieurs hiérarchiques (cf. Figure 51). Cette fenêtre est contenue dans l'objet GroupChat du toolkit.

Le fichier **afficheCarteVisite.html** va afficher la carte de visite de la personne pour laquelle l'item « Afficher carte de visite » a été sélectionné. La commande via une balise div dans le body du fichier est :

```
<div dojoType="sametime.BusinessCard" userId="NomUtilisateur"></div>
```

- Intégration dans le plugin : package **fr.recouv.collaboratif.foncNomPrenom**

La classe **FoncNomPrenom** regroupe les méthodes **afficheMenu**, **comPointAPoint**, **recupN1, comMultiple** et **afficheCarteVisite**.

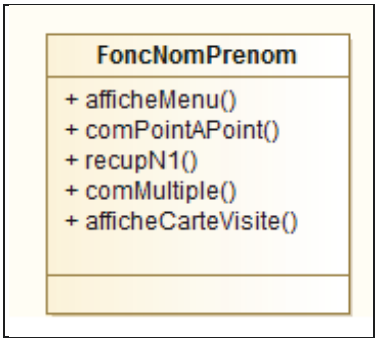


Figure 71 Classe FoncNomPrenom

La connexion automatique à Sametime se fait par le biais d'un token LTPA passé en paramètre lors de l'ouverture des fichiers html. La récupération de celui-ci est faite dans la recupTokenLTPA présente dans le package principal du plugin.

- Méthode afficheMenu

La méthode afficheMenu permet l'affichage du menu qui est composé des items suivants:

- Communiquer avec
- Communiquer avec N+1
- Afficher carte de visite

La sélection d'un item provoque l'appel d'une ou plusieurs méthodes:

- Communiquer avec : comPointAPoint(userId)
- Communiquer avec N+1 :
  - recupN1(userId) qui retourne la liste du ou des N+1 (listeN1)
  - Si listeN1contient plusieurs identifiants : comMultiple(listeN1) sinon comPointaPoint(listeN1)

- Afficher carte de visite : afficheCarteVisite(userId)

Où userID est le userId associé au couple nom_prénom depuis lequel est déclenché le menu

- Méthode recupN1

La méthode recupN1 va retourner le ou les N+1 de l'agent dont le userId aura été passé en paramètre.

Le résultat est de type String et les userid des N+1 sont séparés par un point-virgule.

Il existe une API pour récupérer les données hiérarchique d'un utilisateur dans OPUS[38]. Elle est exposée dans le service web IServiceExterneFacadeV3_6.

- Méthode comPointAPoint

Elle crée une fenêtre de type browser qui affiche le ficher comPointAPoint.html. L'appel de ce fichier se fait en passant en paramètre le nom de l'utilisateur et le token (récupérés en utilisant les méthodes du package principal). Le pseudo code de cette méthode est exposé dans le document de spécification.

- Méthode comMultiple

La méthode est similaire à celle de comPointAPoint à la différence qu'elle ouvre le fichier web : comMultiple.html et qu'au lieu d'un seul nom d'utilisateur, une liste de noms d'utilisateurs est passé en paramètre.

- Méthode afficheCarteVisite

Cette méthode est construite de la même manière que les deux précédentes. Elle ouvre le fichier afficheCarteVisite.html en passant en paramètre le nom de l'utilisateur et le token.

6.3.2.6 Présentiel

Pour implémenter le présentiel dans Watt, il a été décidé d'utiliser le toolkit Java fourni par le SDK de Sametime.

Cette réalisation est codée dans le package **fr.recouv.collaboratif.presentiel** qui est intégré au plugin du projet fr.recouv.collaboratif.

Ce package est découpé en deux parties une côté serveur et l'autre côté client (cf. Figure 63).

- Partie serveur

La partie côté serveur, met en œuvre le toolkit Java en initialisant une session Sametime et les listeners du toolkit nécessaires au présentiel puis expose le service d'information sur le statut d'un agent. De ces cas d'utilisations découle le diagramme de classe ci-dessous.

[38] OPUS : application du recouvrement qui a pour but d'administrer l'organigramme d'une URSAFF régionale. Tous les métiers de cette dernière y sont représentés. Chaque région à son propre organigramme.

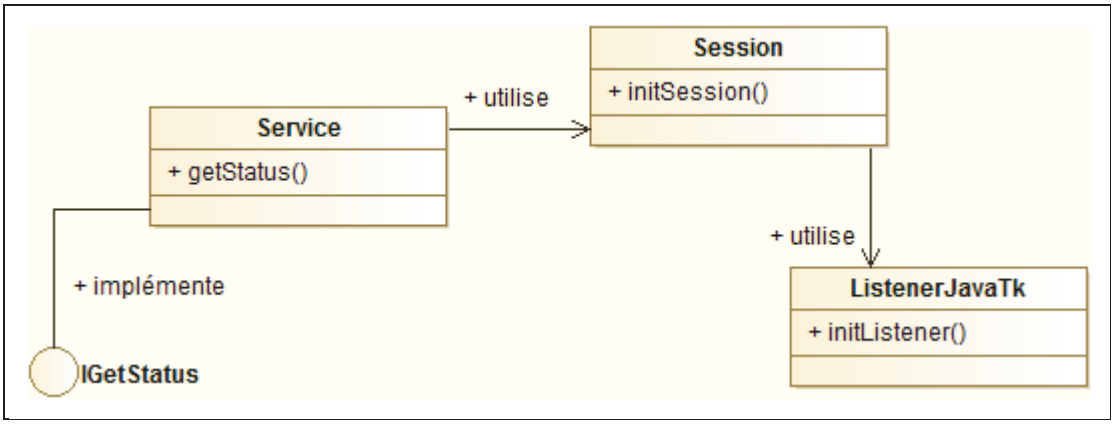


Figure 72 Diagramme des classes côté serveur

Le serveur initialise une session unique avec Sametime (avec l'utilisateur automate Watt, cf. paragraphe 6.3.1.6) via la méthode **initSession**.

La méthode **initListener** permet à la session de charger les fonctionnalités du toolkit Java en initialisant les listener fournis par le toolkit.

Le service exposé **getStatut** renvoie le statut d'un utilisateur passé en paramètre.

Pour utiliser le toolkit Java**,** il faut intégrer au package les 3 fichiers CommRes.jar, STComm.jar et stjavatk.jar, présents sous \client\stjava\bin dans le dossier du SDK.

Le pseudo code des méthodes est détaillé dans le document de spécifications techniques.

- Partie client

Côté client, le package fait appel au service exposé par le serveur et permet l'affichage d'une icône correspondant au résultat du service dans un couple nom_prénom.

La classe Presentiel couvre les fonctionnalités non développées dans le package foncNomPrenom (cf. paragraphe 6.3.2.5).

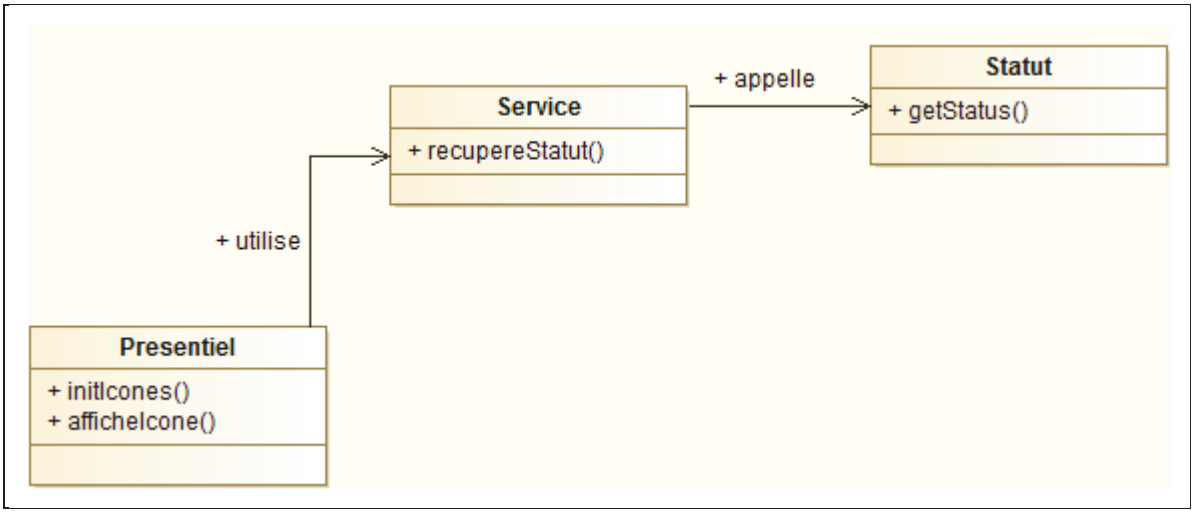


Figure 73 Diagramme de classe côté client

La méthode **initIcones** permet de charger les icônes de présentiel stockées dans le package, elle est appelée par la méthode afficheIcone.

La méthode **afficheIcones** retourne l'icône correspondant au statut récupéré côté serveur via la méthode recupererStatut. Les différentes valeurs retournées possibles sont :

- ✓ 32 ou 512 : statut « en ligne »
- ✓ 64 ou 96 : statut « absent »
- ✓ 8 ou 128 : statut « occupé »
- ✓ dans tous les autres cas le statut est considéré comme hors ligne

Le code ou pseudo code des méthodes est détaillé dans le document de spécifications techniques.

- Partie communication

La communication entre la partie client et la partie serveur se fait par service web (cf. Figure 74). La couche cliente appelle le service en utilisant une façade. La façade récupère l'interface service et lance l'appel à la méthode concernée qui se trouve dans l'implémentation du service.

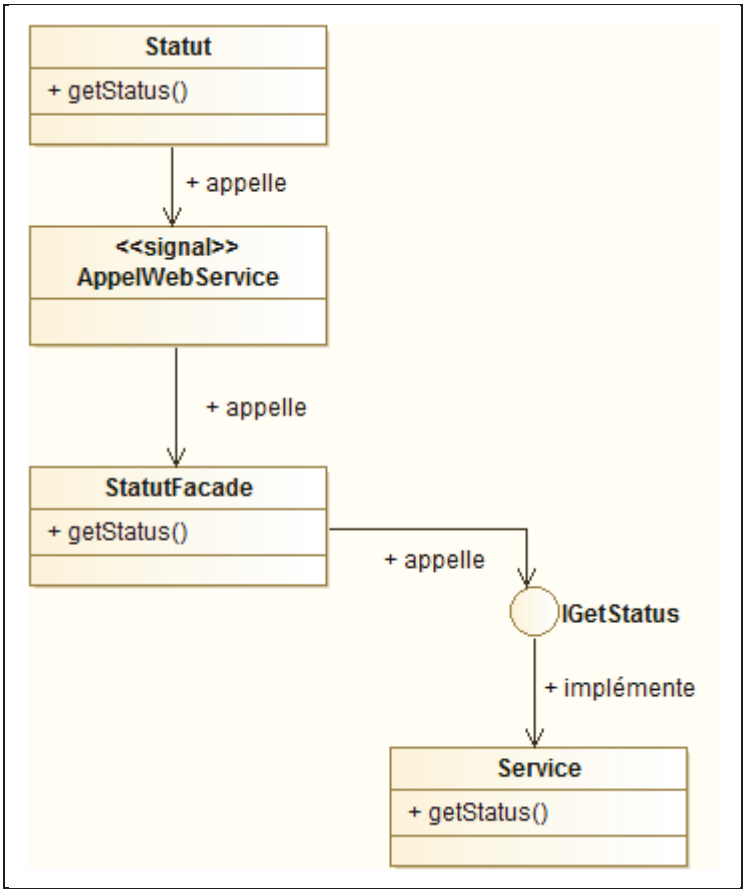


Figure 74 Diagramme des classes qui communiquent par service web

7 Bilans

7.1 Bilan et perspective projet

- ✓ Bilan

Comme expliqué au chapitre 5.1, la méthodologie de rigueur au CIRSO est la méthodologie CAIRN mise au point par ALTAIR. Le cycle de vie d'un projet est découpé en plusieurs phases au cours desquelles des livrables sont initialisés et /ou validés.

J'ai pu opérer dès la mise en place du projet qui s'est traduite par une réunion de cadrage (lors de l'étape de cadrage dans la figure ci-dessous). Le projet n'était pas encore véritablement défini. Seuls quelques besoins avaient été émis au travers de l'enquête de satisfaction des utilisateurs de Watt. La définition du projet et l'émergence des besoins a par conséquent nécessité beaucoup de temps.

Mon intervention a donc pour particularité le fait d'avoir participé à la définition du projet et à la production du cahier des charges (lors de l'étape de définition des besoins dans la figure ci-dessous). J'ai ainsi en partie endossé le rôle de MOA pour mener à bien ce projet puis assumé le rôle de MOE en participant à la proposition de solution pour la mise en œuvre des exigences (étape de réalisation).

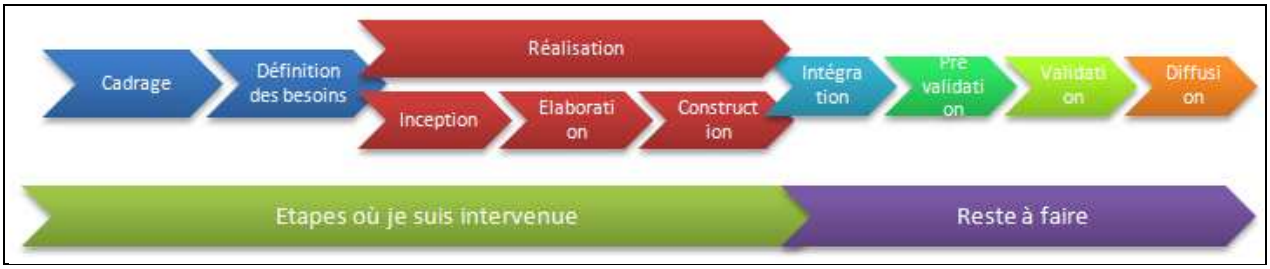


Figure 75 Positionnement de mon intervention sur le cycle de vie du projet

La méthodologie CAIRN met à disposition une bibliothèque de documents livrables que chaque projet s'approprie. Une fois le cahier des charges rédigé, le projet est entré dans sa phase de réalisation. Celle-ci se décompose en trois étapes (voir figure ci-dessus). A défaut d'utiliser une méthode ou modélisation réfléchie pour la mise en place de pratiques collaboratives dans le workflow comme nous l'avons vu dans l'état de l'art, je me suis appuyée sur la modélisation UML qui est employée dans certains documents proposés par CAIRN (méthode requise au CIRSO). Au stade de l'inception, j'ai regroupé les besoins et modélisé des diagrammes de cas d'utilisations. Ceux-ci ont été consignés dans le DCU, document des cas d'utilisations et ont permis de définir les spécifications fonctionnelles du projet.

Lors de la phase d'élaboration, j'ai généré le document de matrice de couverture des exigences. Celui-ci m'a permis de m'assurer que chaque besoin émis dans le cahier des charges était bien couvert par une exigence fonctionnelle.

A ce moment du projet, nous avons décidé de le scinder en deux parties en détachant tout ce qui concernait les notifications dans un lot à part. J'ai donc continué les phases d'élaboration et de construction sans traiter les exigences de ce lot.

J'ai attaqué la phase de construction par le choix d'une solution technique. L'outil à utiliser, le toolkit Sametime d'IBM, était déjà choisi par le CIRSO. L'étude n'a donc pas porté sur les outils pouvant apporter une solution au projet mais sur les possibilités offertes par le toolkit. J'ai alors élaboré les spécifications techniques en fonction des choix faits sur l'utilisation du toolkit. Ici encore la modélisation UML m'a guidée pour rédiger ces documents. A partir des travaux précédents j'ai construits les diagrammes de classe qui ont servi de base aux fichiers.

La figure ci-dessous synthétise les différents documents réalisés au cours du projet et en fonction de l'étape en cours.

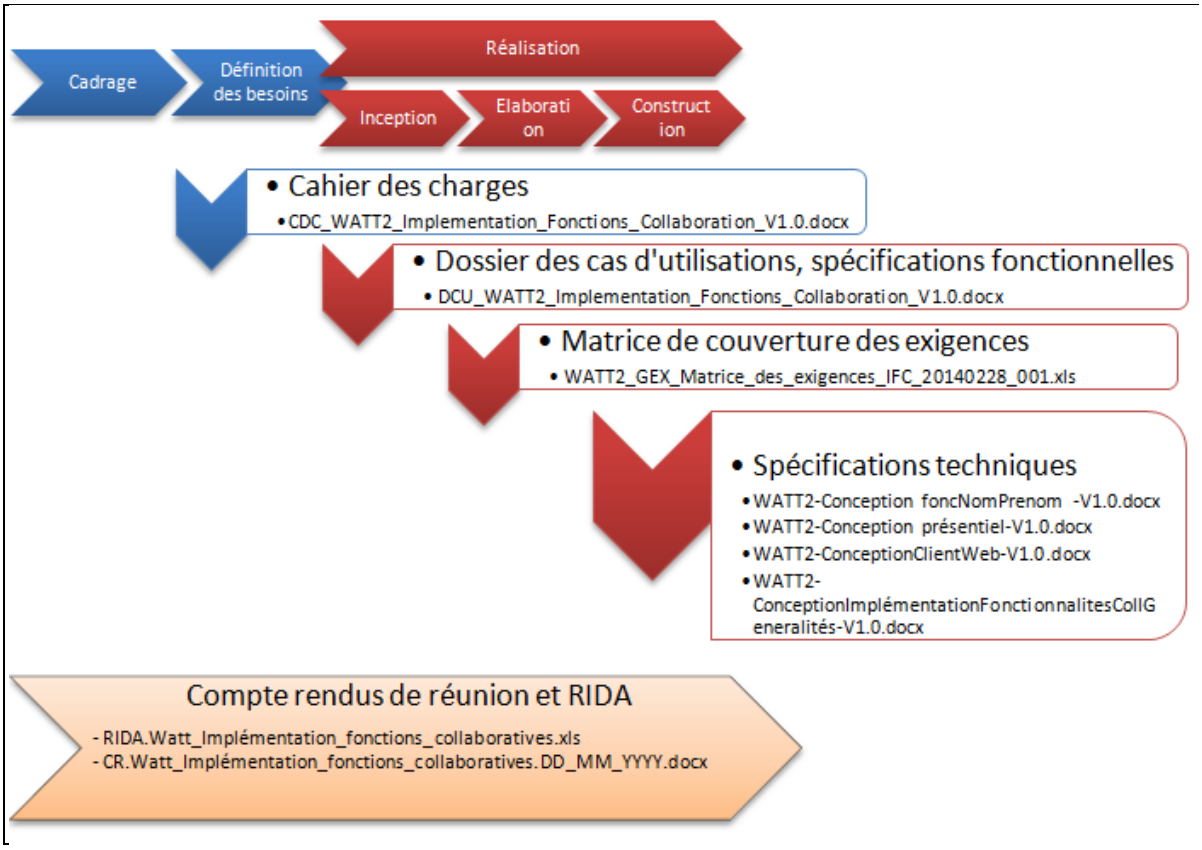


Figure 76 Documents élaborés au cours du projet

L'infrastructure du moment n'était pas adéquate pour mettre en œuvre les fonctionnalités décrites dans les spécifications techniques. J'ai donc réalisé un POC pour vérifier la faisabilité des documents. J'ai créé une mini application en java sous Eclipse RCP pour simuler l'utilisation des services à développer. Ces services ont été regroupés dans un plugin. A l'issue du POC, j'ai mis à disposition un paquet contenant un document de cadrage du POC ainsi que les développements effectués et un compte rendu permettant de les exploiter.

Les points négatifs qui ressortent de ce projet sont les difficultés rencontrées pour démarrer le projet.
Tout d'abord, la définition du projet même a été assez fastidieuse. Les termes et les besoins pour le projet n'étaient pas encore totalement connus. Il a fallu trouver des temps de concertation où la majorité des acteurs était disponibles. Ceci étant rendu difficile de par la période (Septembre) qui était très chargée et par l'approche de la dernière phase de régionalisation qui mobilisait tout le service.
Ensuite, l'élaboration du cahier des charges a souffert des mêmes problèmes. Du fait de la difficulté à réunir suffisamment d'acteurs en même temps, la rédaction et la validation de celui-ci s'est éternisé dans le temps. S'ajoutant à ce problème, le fait que les besoins ont beaucoup variés jusqu'à la validation du cahier des charges.

La suite du projet a moins souffert car moins d'acteurs étaient concernés au fur et à mesure de son avancée. De mon côté j'avais pris mes marques au sein des équipe et du projet, ce qui m'a permis d'avancer plus aisément.

La prise de connaissance de l'outil fournit par IBM (SDK) et sa mise en œuvre m'ont aussi apporté leur lot de complexité. Il m'a fallu beaucoup chercher et tâtonner afin d'arriver à les exploiter. En effet, la documentation et les exemples étaient souvent incomplets et erronés.

Du côté des points favorables je citerai la satisfaction d'avoir su surmonter les problèmes techniques et fonctionnels rencontrés pour arriver à la réalisation d'une première maquette. Il ne m'a pas été possible de terminer la réalisation du projet mais j'ai pu fournir tous les livrables qui permettront de le faire.

Pour parvenir à ces résultats, je me suis fortement appuyée sur la démarche méthodologique employée au CIRSO. Sur le feu de l'action, j'avais parfois la sensation de ne pas avancer et ne plus respecter cette méthodologie. C'est en prenant du recul que chaque chose a repris sa place et du sens.

L'utilisation de la modélisation pour faciliter la compréhension du projet par tous les acteurs est une nouveauté pour moi. J'avais déjà étudié la conception objet et l'UML mais n'avait pas encore été amenée à la mettre en œuvre dans le cadre d'un projet. Aujourd'hui, en prenant un peu de recul sur le projet, je réalise que mes diagrammes ne sont pas totalement conformes. Néanmoins, ils ont contribués à une bonne compréhension par tous et nous ont servi de fil conducteur entre les différentes phases du projet. Comme pour tout, seul l'expérience nous permet de bien maîtriser les techniques. La modélisation a donc apporté une valeur constructive au projet.

- ✓ Perspectives

Les perspectives ne manquent pas face à l'étendue de ce projet.

La première étant bien étendu l'achèvement des développements consécutifs à l'étude que j'ai menée. Le budget actuel du CIRSO ne permettant pas de le faire dans le cadre des activités du

personnel du service, un stagiaire est pressenti pour assurer la continuité du projet. A ce jour il n'a pas encore été identifié. Cependant, comme nous l'avons vu dans le document, il n'est pas nécessaire de précipiter sa réalisation car l'infrastructure nécessaire n'est pas encore totalement en place. L'aboutissement du projet sera alors effectif lorsque les développements auront été faits à l'aide des spécifications techniques puis testés, intégrés, livrés en production et enfin validés à chaque étape.

Un autre besoin ressorti de l'étude concerne la mise en place d'alertes et de notifications mais aussi d'un système d'historisation de ces items sous forme de fil d'activité propre à chaque utilisateur. L'outil Activity Stream d'IBM est pressenti pour la réalisation de ces fonctionnalités. La mise en œuvre de ces besoins peut être faite conjointement à la réalisation des développements évoqués ci-dessus ou bien à la suite. Ces projets peuvent être dissociés. Au jour où je termine de rédiger ce mémoire (Mai 2015), j'apprends qu'un stagiaire est actuellemnt entrain de reprendre cette partie au CIRSO.

Les fonctionnalités implémentées dans ce projet peuvent ensuite servir à mettre en place d'autres options. L'information de présentiel, par exemple, peut être utilisée par un manager lorsqu'il souhaite réaffecter une affaire à un agent. Dans ce cas, il s'agira d'une affaire à traiter instantanément. Cependant, le SDK met aussi des outils à disposition permettant de récupérer des informations de l'agenda (celui-ci est géré par les utilisateurs depuis l'outil Lotus Notes, cf. 3.2.1.1). Le manager, pourra alors affecter une affaire en fonction de la priorité de celle-ci et du temps de disponibilité de l'agent.

Un gros avantage du produit final est qu'il est livré sous forme de plugin. Il pourra alors être utilisé par une multitude d'applications du recouvrement. Celles-ci auront la possibilité de réutiliser les fonctionnalités mises en œuvre et donc d'élargir leur faculté de collaboration.

7.2 Bilan personnel

Du point de vue personnel, j'ai été confronté à beaucoup de difficultés mais ai aussi pu avoir la satisfaction de voir mon travail se concrétiser petit à petit.

Un des principaux freins a été la méconnaissance technique d'Eclipse. J'avais, lors de mes études, appréhendé la conception objet et le langage Java. L'utilisation d'Eclipse demande cependant une connaissance et une maitrise plus élargie de ces concepts. J'ai avancé à tâtons et repris les notions au fur et à mesure que j'évoluais dans mon travail. J'ai aussi bénéficié des aptitudes des personnes qui m'ont accompagné sur le projet. J'ai conscience qu'il me faudra d'avantage d'expérience pour pouvoir me targuer de savoir travailler sous Eclipse mais je serais ravie d'approfondir mes compétences dans ce domaine.

La prise de connaissance de l'outil fournit par IBM (SDK) et sa mise en œuvre m'ont aussi apporté leur lot de complexité. Il m'a fallu beaucoup chercher et tâtonner afin d'arriver à les exploiter. En effet, la documentation et les exemples étaient souvent incomplets et erronés.

L'utilisation de la modélisation pour faciliter la compréhension du projet par tous les acteurs est une nouveauté pour moi. J'avais déjà étudié la conception objet et l'UML mais n'avait pas

encore été amenée à la mettre en œuvre dans le cadre d'un projet. Aujourd'hui, en prenant un peu de recul sur le projet, je réalise que mes diagrammes ne sont pas totalement conformes. Néanmoins, ils ont contribués à une bonne compréhension par tous et nous ont servi de fil conducteur entre les différentes phases du projet. Comme pour tout, seul l'expérience nous permet de bien maîtriser les techniques. La modélisation a donc apporté une valeur constructive au projet.

Je me suis aussi heurtée à des problèmes d'ordre organisationnel. Leur cause peut être partagée entre les différents protagonistes du projet et moi-même. Mais elle trouve aussi son origine dans les aléas de la conduite de projet, en l'occurrence lorsqu'il était nécessaire d'avoir un maximum d'acteurs réunis afin de prendre des décisions et de continuer le travail. Lorsque j'ai débuté mon stage, tout le monde avait beaucoup à faire avec l'achèvement de la régionalisation. De mon côté, je n'ai pas su prendre directement mes marques. Le projet m'ayant été confié, il m'incombait donc la responsabilité de le piloter et d'assurer une cohérence entre les participants lors des réunions. Après une période de flottement, j'ai enfin réussi à prendre en main petit à petit la conduite du projet et ce grâce aux outils fournis par la méthodologie CAIRN ainsi qu'à tout ce que j'ai pu apprendre au cours de ma formation au Cnam sur le management, la communication, le travail en équipe…

La définition du projet a été assez fastidieuse. Les termes et les besoins pour le projet n'étaient pas encore totalement connus. Comme je l'évoque dans le paragraphe précédent, il a fallu trouver des temps de concertation où la majorité des acteurs était disponibles. Ceci étant rendu difficile de par la période (Septembre) qui était très chargée et par l'approche de la dernière phase de régionalisation qui mobilisait tout le service.
Ensuite, l'élaboration du cahier des charges a souffert des mêmes problèmes. Du fait de la difficulté à réunir suffisamment d'acteurs en même temps, la rédaction et la validation de celui-ci s'est éternisé dans le temps. S'ajoutant à ce problème, le fait que les besoins ont beaucoup variés jusqu'à la validation du cahier des charges.

J'ai attaqué ce stage alors que je terminais un congé parental. Cela a représenté pour moi un défi inattendu. Je n'avais pas réalisé que la reprise du travail, après une période d'inactivité moyenne accompagné d'un état de fatigue général, pouvait être aussi contraignante. Cet aspect strictement personnel, a largement contribué à ralentir mon dynamisme pour démarrer le travail.

En m'appuyant sur mes connaissances, mes acquis lors de ma formation au Cnam, et le soutien des personnes qui m'entouraient sur le projet et en suivant la méthodologie du CIRSO, j'ai pu graduellement apporter un aspect concret au projet. Progressivement j'ai pris mes marques, commencé à ébaucher les exigences découlant des besoins, dressé des diagrammes que j'ai ensuite soumis à validation. Le projet a pris vie petit à petit pour au final aboutir à une maquette puis des spécifications techniques. J'ai souvent eu le sentiment de piétiner. Aujourd'hui avec du recul, je réalise l'ampleur des travaux menés jusque là et je suis satisfaite du chemin parcouru et d'avoir réussi à produire tous les livrables attendus en dépit de tous les obstacles qui ont interféré.

Conclusion

Dans le but d'écrire ce mémoire, j'ai effectué mon stage au CIRSO. Mon travail a porté sur l'étude d'implémentation de fonctionnalités collaboratives dans l'application Watt de workflow procédural aux URSSAF.

La définition des besoins a été un processus assez long car les exigences n'étaient pas formulées. De plus, la disponibilité de la MOA était très entamée par les grands projets prioritaires ainsi que le manque de clarté sur le fonctionnement MOE/MOA lié à la « réorganisation » en cours des conventions d'objectifs et de gestions. J'ai alors endossé à la fois le rôle de MOA et de MOE pour définir le cahier des charges puis décliner les exigences dans des spécifications techniques.

La réalisation des livrables a suivi une méthodologie de rigueur au CIRSO, la méthodologie CAIRN. J'ai donc dû évoluer dans le projet au moyen d'étape et livrables proposés par cette méthodologie. J'ai pu à cette occasion mettre en œuvre mes connaissances en modélisation. Les documents de spécifications fonctionnelles et techniques utilisent comme fil conducteur des diagrammes UML.

Une fois les besoins exprimés, ceux-ci ont été déclinés dans des diagrammes de cas d'utilisations. Cela a permis d'extraire les spécifications fonctionnelles..J'ai ensuite rédigé les spécifications techniques que j'ai illustré au moyen de diagrammes de classe.

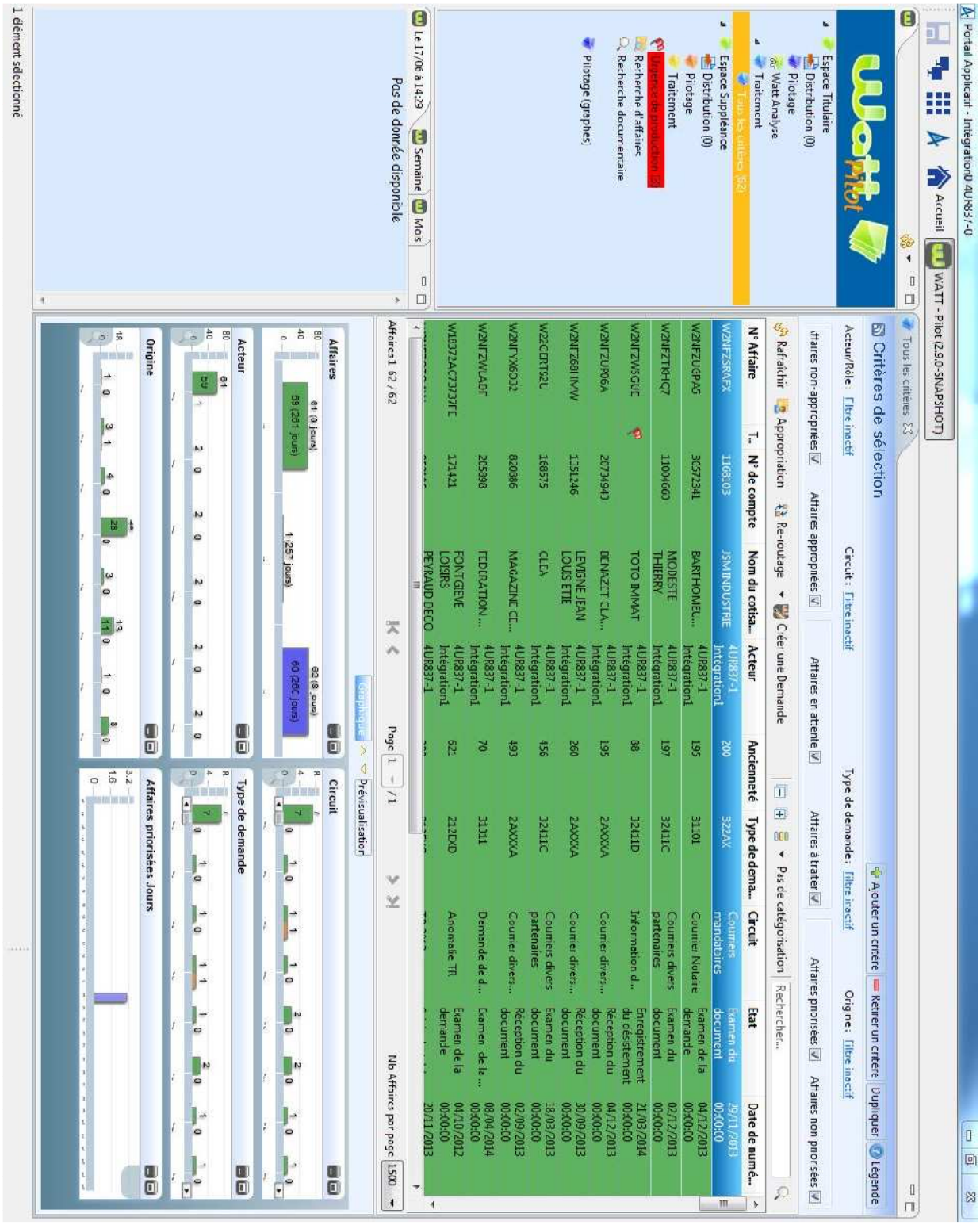
Le choix de l'outil à implémenter dans Watt s'est porté sur la messagerie instantanée utilisée aux URSSAF Sametime. Il a été décidé d'utiliser un toolkit de développement mis à disposition par IBM. Ce toolkit propose plusieurs outils permettant l'intégration de fonctionnalités offertes par le toolkit dans des applications de divers types (client, serveur…).

Une première phase d'étude de faisabilité a déterminé les outils choisis dans le toolkit. La réalisation des fonctionnalités n'a pas pu se faire en grandeur réelle car la structure actuellement déployée ne supportait pas cette solution. J'ai donc été amené à réaliser un POC pour vérifier la faisabilité des outils à mettre en œuvre sur une simulation de liste comme celle présente dans Watt. Les développements se sont faits en environnement Eclipse RCP. La réalisation du POC a confirmé la faisabilité du projet. J'ai ainsi réalisé un plugin qui pourra servir à la réalisation réelle dans Watt.

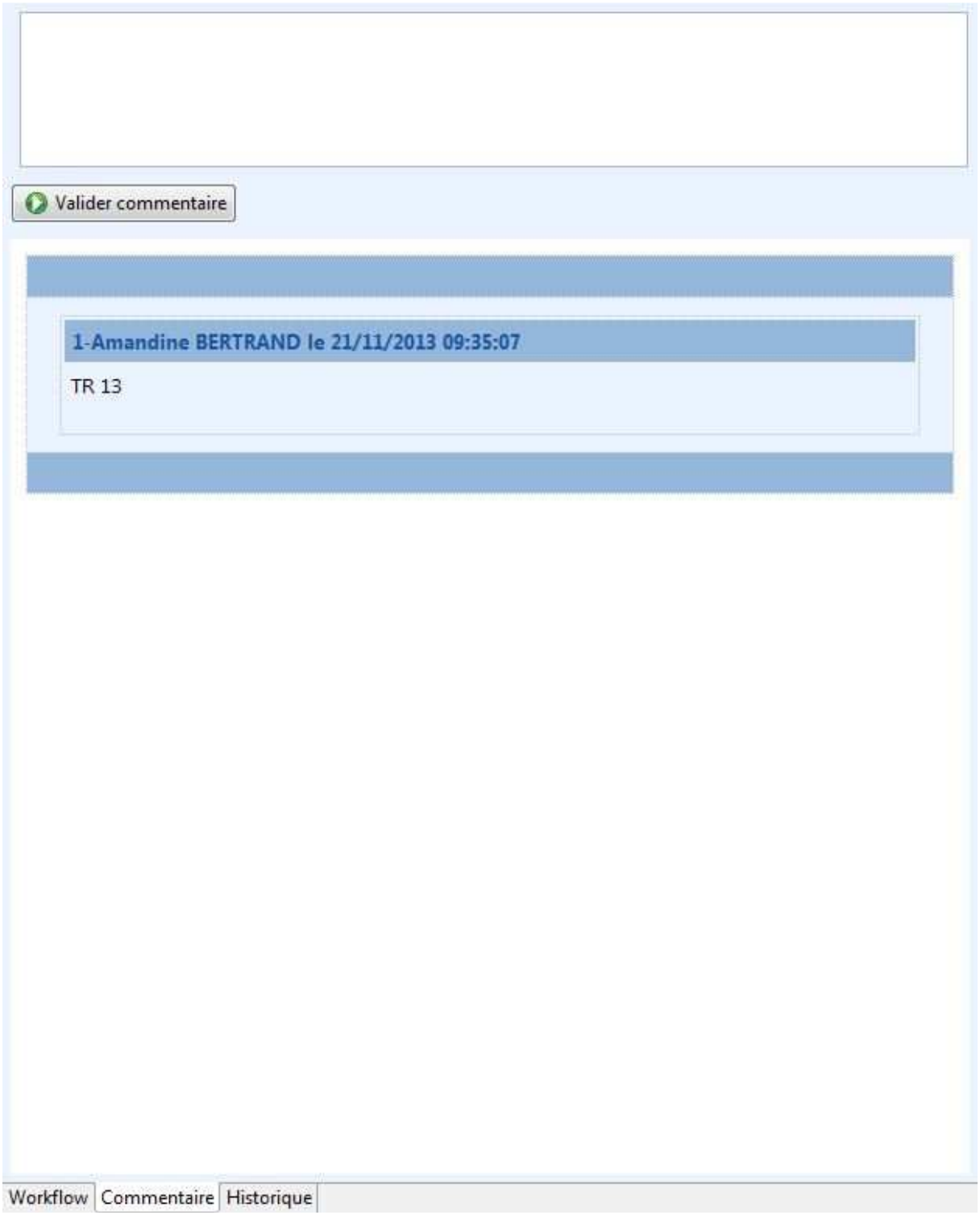
La solution étudiée pourra donc être mise en œuvre dès que la structure des serveurs le permettra, c'est-à-dire que le serveur proxy aura été déployé en production et sera un domaine différent de celui du serveur de communauté. A ce stade, il restera donc à développer pour Watt la solution en réutilisant le plugin conçu lors du POC et en s'appuyant sur les spécifications techniques. Puis pour respecter la bonne démarche du projet, il faudra tester la solution et la déployer.

Annexes

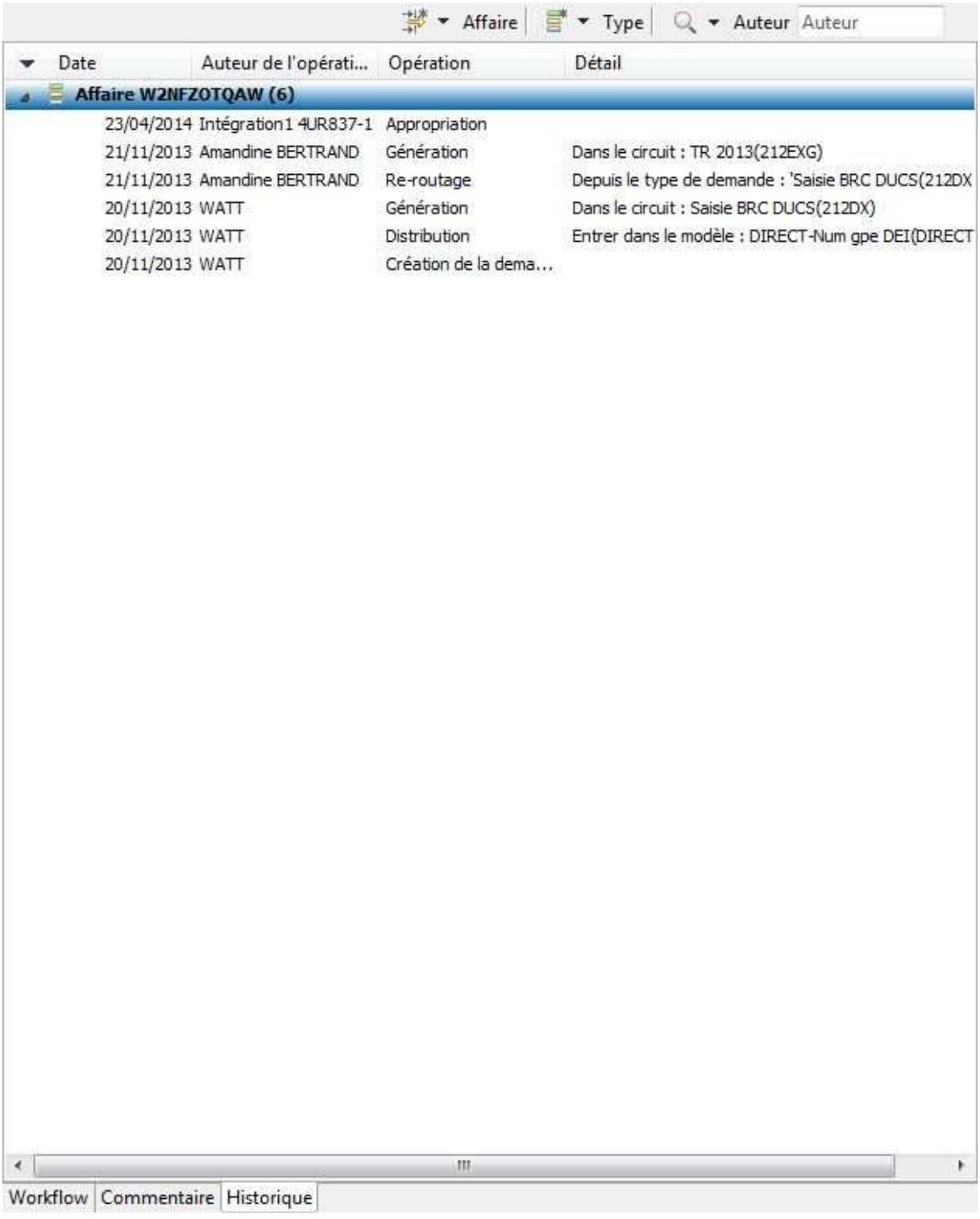
Annexe I - Corbeille dans Watt Pilot



ANNEXE II - Watt traitement – Onglet Commentaire



ANNEXE III - Watt traitement – Onglet Historique



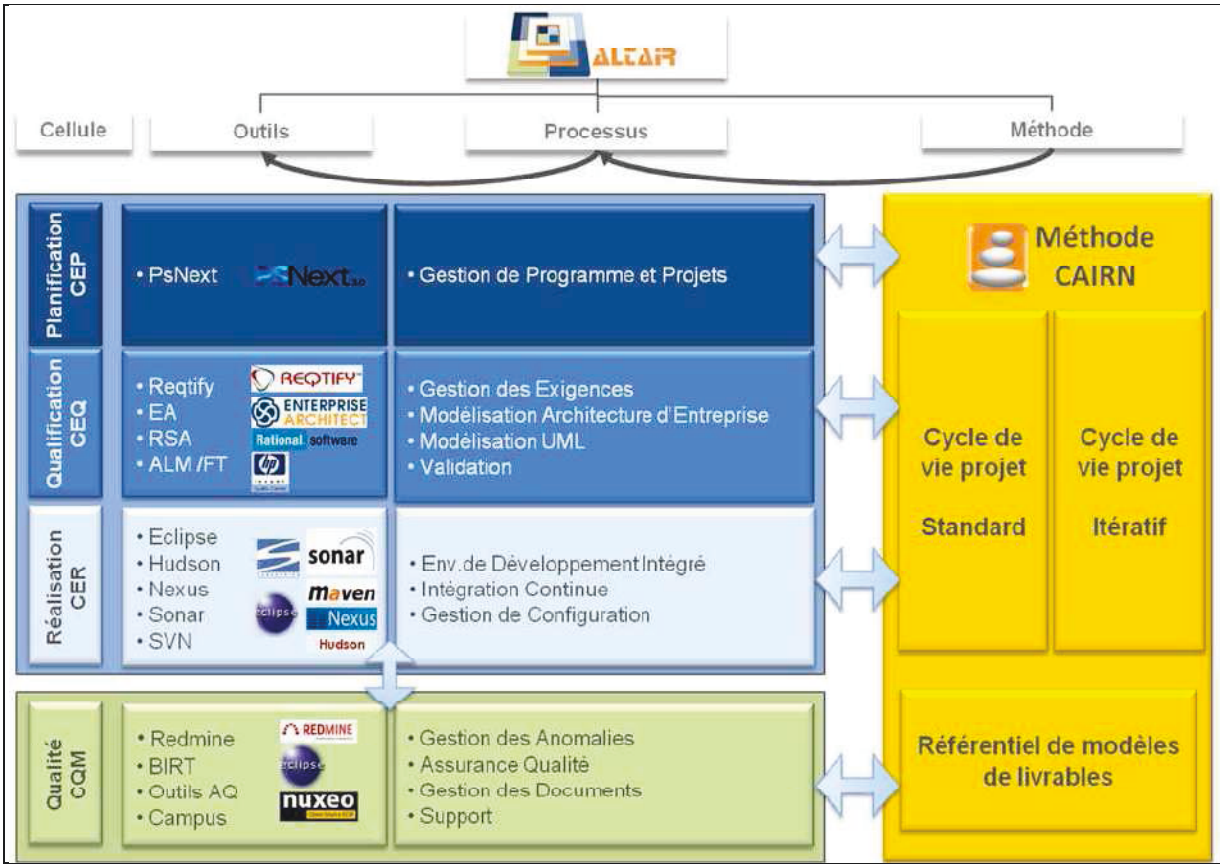
ANNEXE IV - ALTAIR

Source wiki ALTAIR : http://wiki.altair.recouv/doku.php?id=altair:start

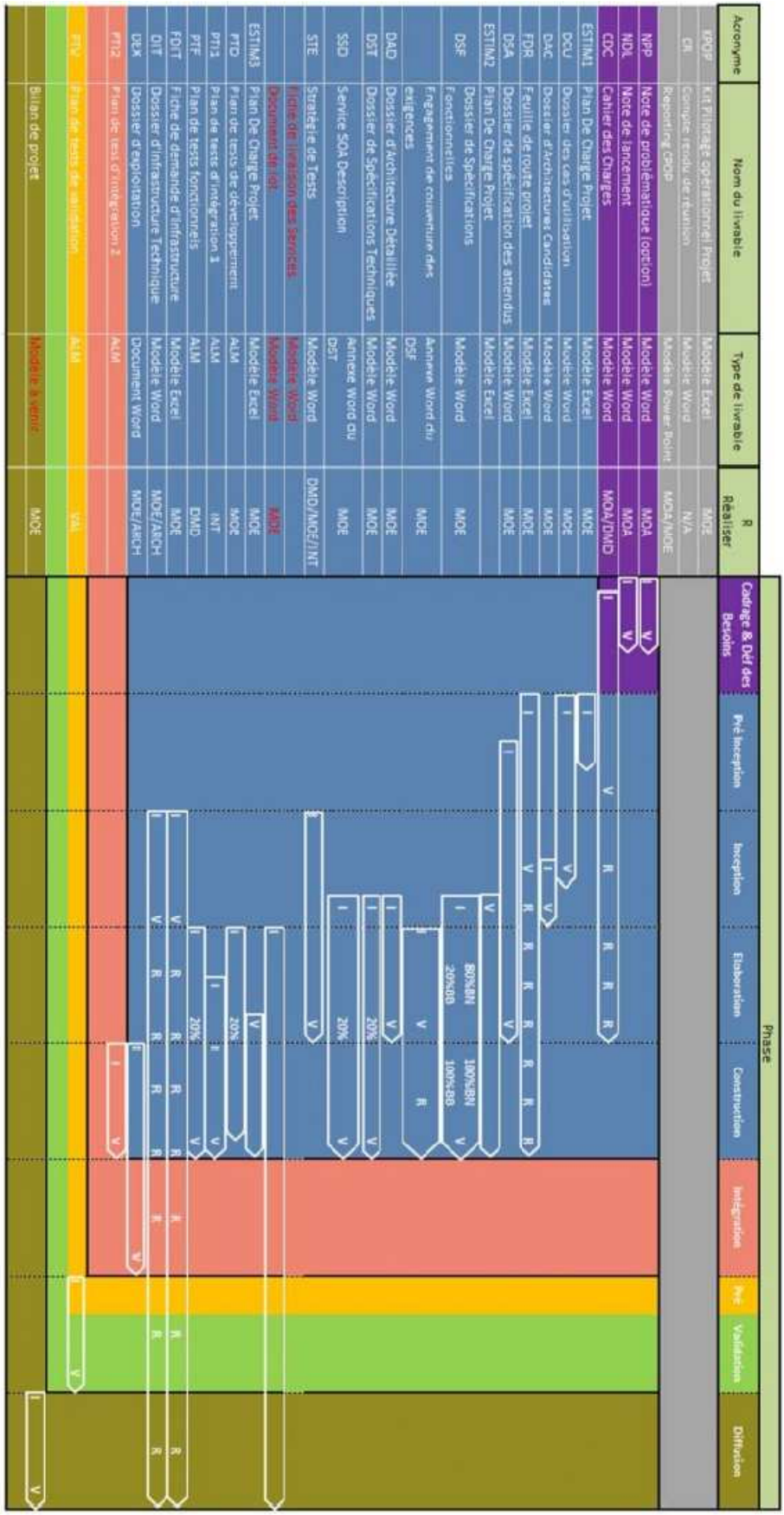
Atelier Logiciel Transverse Appliqué à l'Informatique du Recouvrement (ALTAIR)

L'atelier ALTAIR regroupe les 4 composantes suivantes :

- 11 **domaines d'expertise** couvrant la chaine de fabrication d'un logiciel. Chaque domaine est décrit par un processus issu de pratiques éprouvées dans chacun des centres informatique de la branche et d'un modèle de référence (CMMI)
- 14 **outils** associés aux domaines
- Une **méthodologie projet** détaillant les activités à mener dans chaque phase du **cycle de vie d'un projet** en exploitant l'ensemble des 11 domaines d'expertise de la chaine de fabrication logicielle.
- Un **support** sur l'ensemble de ces éléments aux travers de 4 cellules d'expertises relayées par un réseau de référents et d'utilisateurs clés dans chaque centre informatique de la branche.



ANNEXE V -Livrables CAIRN



ANNEXE VI - Récapitulatif des exigences

IDExigenceCdC	Titre Exigence CdC	Description Exigence CdC	Criticité Exigence
CDC_WATT2_IFC_F_000101	Connexion messagerie instantanée	Connexion automatique ou manuelle à l'ouverture de WATT	Critique
CDC_WATT2_IFC_F_000102	Communauté	L'utilisateur est connecté à la communauté recouvrement L'utilisateur peut discuter avec n'importe quelle personne de l'annuaire DONATIR	Critique
CDC_WATT2_IFC_F_000103	Espace dédié	L'espace dédié est accessible quelque soit le contexte Watt lancé dans le portail Harmonie. Il affiche l'interface de messagerie instantanée et l'interface d'historique des notifications	Critique
CDC_WATT2_IFC_F_110101	Accès communication instantanée depuis le nom/prénom	L'accès aux fonctionnalités de communication instantanée se fait par un clic droit sur le nom et/ou prénom de la personne à contacter. Le nom d'une personne à contacter se trouve : - depuis Watt pilot dans la colonne acteur de toutes les corbeilles d'affaires, dans la liste d'acteurs (au niveau du filtre de la corbeille d'affaires) - depuis watt traitement, dans le cartouche d'information de l'affaire, dans la zone historique, dans la zone commentaire - depuis l'espace dédié	Critique
CDC_WATT2_IFC_F_110201	Emission d'un message de communication point à point	A l'activation d'un dialogue instantanée : soit une fenêtre s'ouvre en avant plan non modale, sur l'écran de l'émetteur, soit l'espace dédié se déplie avec le positionnement de la zone de dialogue (forme à étudier). Chaque communication instantanée possède sa fenêtre de dialogue.	Critique
CDC_WATT2_IFC_F_110202	Réception d'un message de communication point à point	A la réception d'un message instantanée d'un nouvel interlocuteur : soit une fenêtre s'ouvre en avant plan non modale, sur l'écran du destinataire, soit l'espace dédié se déplie avec le positionnement de la zone de dialogue (forme à étudier). Chaque communication instantanée possède sa fenêtre de dialogue	Critique

CDC_WATT2_IFC_F_110203	Fenêtre de communication point à point	Elle se présente à l'identique pour les deux utilisateurs. Elle comporte plusieurs zones : Zone historique (fil de discussion) Zone de saisie du message Zone pour la carte de visite de l'émetteur (à définir en fonction du type de fenêtre choisie).	Critique
CDC_WATT2_IFC_F_110205	gestion de la fenêtre de communication point à point	La fenêtre de dialogue peut être fermée, réduite (criticité moyenne), redimensionnée (criticité moyenne).	Critique
CDC_WATT2_IFC_F_110301	Sélection interlocuteurs pour la communication de groupe	La sélection des interlocuteurs pour la communication de groupe se fait de trois façons : - en sélectionnant tous les utilisateurs à contacter dans la liste de l'espace dédié (tout en maintenant la touche CTRL appuyée) puis en cliquant sur le bouton de lancement de communication - en sélectionnant un interlocuteur du groupe puis en cliquant sur le bouton de lancement de communication. Et une fois la fenêtre de communication ouverte, en cliquant sur le bouton « ajouter des personnes » et en ajoutant manuellement chaque utilisateur - en sélectionnant directement le nom du service à contacter	Faible
CDC_WATT2_IFC_F_110302	Emission d'un message de communication de groupe	A l'activation d'un dialogue instantanée : soit une fenêtre s'ouvre en avant plan non modale, sur l'écran de l'émetteur, soit l'espace dédié se déplie avec le positionnement de la zone de dialogue (forme à étudier). Chaque communication instantanée possède sa fenêtre de dialogue.	Faible
CDC_WATT2_IFC_F_110303	Historique communication de groupe	Dans le cas où un utilisateur rejoint la conversation en cours (ou terminée), le dialogue échangé avant son arrivée sur la fenêtre est affiché.	Faible
CDC_WATT2_IFC_F_110304	Acceptation de la communication de groupe	Chaque utilisateur invité doit accepter l'invitation pour rejoindre la communication de groupe	Faible
CDC_WATT2_IFC_F_110305	Réception d'un message de	A la réception d'un message instantanée d'un nouvel interlocuteur :	Faible

	communication de groupe	soit une fenêtre s'ouvre en avant plan non modale, sur l'écran du destinataire, soit l'espace dédié se déplie avec le positionnement de la zone de dialogue (forme à étudier). Chaque communication instantanée possède sa fenêtre de dialogue	
CDC_WATT2_IFC_F_110306	Fenêtre de communication de groupe	Elle se présente à l'identique pour tous les utilisateurs. Elle comporte plusieurs zones : Zone historique (fil de discussion) Zone de saisie du message Zone où sont listés tous les utilisateurs invités dans la communication	Faible
CDC_WATT2_IFC_F_110308	Gestion de la fenêtre de communication de groupe	La fenêtre de dialogue peut être réduite, fermée, redimensionnée.	Faible
CDC_WATT2_IFC_F_120101	Notification de type message d'alerte	Une notification de type message d'alerte est matérialisée par une fenêtre pop up qui disparait grâce à une validation de l'utilisateur. Cette fenêtre est envoyée quel que soit le statut de l'utilisateur	Moyenne
CDC_WATT2_IFC_F_120102	Liste des notifications de type alerte	Une alerte est utilisée pour alerter de : - la coupure de l'application - la mise en place d'une nouvelle version d'un circuit - à compléter	Moyenne
CDC_WATT2_IFC_F_120103	Profil alerte	Une alerte est envoyée par un administrateur fonctionnel à destination des utilisateurs Par un manager ??	Moyenne
CDC_WATT2_IFC_F_120104	Création noification	Un module de création de notifications est disponible dans Watt Administration	Moyenne
CDC_WATT2_IFC_F_120105	Stockage alerte	Les alertes applicatives sont visibles depuis l'historique.	Moyenne
CDC_WATT2_IFC_F_120201	Déclenchement automatique notification	Le gestionnaire en charge d'une affaire est notifié lorsque les actions suivantes sont faites : - la priorité de l'affaire est modifiée - l'affaire est qualifiée d' « Urgence de production ». - son manager a procédé au remplacement de l'acteur d'une affaire et lui a attribué cette affaire - un commentaire a été posté sur l'affaire - à compléter	Critique

CDC_WATT2_IFC_F_120301	Notifications	Une notification apparaît dans le centre de notification sous forme réduite (voir exigence CDC_WATT2_IFC_F_010206 Titre notification). Un clic sur la notification permet de la dérouler pour en lire le détail (voir exigence CDC_WATT2_IFC_F_010206 Contenu notification complète)	Critique
CDC_WATT2_IFC_F_120303	Titre notification	La forme réduite de la notification est composée : - de la date et de l'heure d'envoi de la notification - du libellé abrégé de la notification - s'il s'agit d'une affaire, de son numéro avec le lien vers l'affaire	Critique
CDC_WATT2_IFC_F_120304	Contenu notification complète	La notification complète est composée d'un message descriptif. Pour chaque type de notification sera défini un modèle de message.	Critique
CDC_WATT2_IFC_F_120401	Historique des notifications	Chaque notification est stockée dans l'historique	Critique
CDC_WATT2_IFC_F_120402	Classement des notifications	Les notifications sont classées par ordre chronologique de la plus récente à la plus ancienne	Critique
CDC_WATT2_IFC_F_120403	Purge historique	Une notification est conservée 7 jours dans l'historique	Critique
CDC_WATT2_IFC_F_120404	Présence de nouvelles notifications	Un indicateur graphique (par exemple de type étoile) permet de voir rapidement qu'il y a des nouvelles notifications depuis la dernière lecture	Critique
CDC_WATT2_IFC_F_120405	Nombre de notifications non lues	Un indicateur informe sur le nombre de notifications non lues	Moyenne
CDC_WATT2_IFC_F_120406	Différenciation entre notifications lues et non lues	Les notifications non lues sont visuellement faciles à différencier des notifications lues	Critique
CDC_WATT2_IFC_F_210101	Liste de contacts	Une liste de contacts favoris est accessible depuis l'espace dédié. L'utilisateur y ajoute les contacts qu'il souhaite	Moyenne
CDC_WATT2_IFC_F_210102	Liste membres du même rôle	Une liste des membres titulaires du même rôle est affichée par défaut dans l'espace dédié	Faible

CDC_WATT2_IFC_F_210103	Accès vers manager N+1	Le nom du manager est facilement identifiable, il est mis en valeur par rapport à la liste des membres du rôle	Faible
CDC_WATT2_IFC_F_210104	Recherche	Un module de recherche est accessible depuis l'espace dédié. La recherche se fait selon le nom et/ou prénom des utilisateurs	Faible
CDC_WATT2_IFC_F_210105	Saisie intuitive	La saisie dans la recherche du nom d'un utilisateur est intuitive. Une liste de personnes est proposée au fur et à mesure que le nom est saisi	Faible
CDC_WATT2_IFC_F_220101	Affichage carte de visite	Un utilisateur visualise la carte de visite d'un autre utilisateur en survolant son nom/prénom	Faible
CDC_WATT2_IFC_F_220102	Communication avec le manager d'un acteur	Le menu déroulé via un clic droit sur le nom prénom d'une personne propose la communication instantanée avec son manager	Faible
CDC_WATT2_IFC_F_220103	Indicateur graphique de nouvelle notification	Dans les corbeilles sous Watt Pilot, un indicateur graphique es affiché pour les affaires ayant fait l'objet d'une nouvelle notification	Faible
CDC_WATT2_IFC_F_230101	Statut utilisateur	Les statuts possibles sont : - Disponible - Absent (Criticité moyenne) - En réunion (Criticité moyenne) - Ne pas déranger (Criticité moyenne)	Critique
CDC_WATT2_IFC_F_230102	Personnaliser le statut	Un utilisateur peut modifier son statut ainsi que le libellé de son statut	Faible
CDC_WATT2_IFC_F_230103	Visualiser le statut des utilisateurs	Dès que le nom/prénom d'un utilisateur apparait sur les interfaces, son statut est matérialisé	Critique
CDC_WATT2_IFC_F_230104	Mise à jour du statut des utilisateurs	La modification du statut d'un utilisateur entraine une mise à jour du statut sur les interfaces. La matérialisation du statut change en fonction du nouvel état	Critique
CDC_WATT2_IFC_F_230105	Communication instantané avec statut « Ne pas déranger »	Si le statut est « Ne pas déranger » l'utilisateur ne reçoit pas de messages de communication instantanée	Moyenne

ANNEXE VII - Services proposés par le toolkit Java

Community Services

The following table lists and describes the Community Services available in the Sametime Java Toolkit.

Table listing available services in Sametime Java Toolkit.

Name of Service	Description
Community Service	This service is a required component in almost any use of the toolkit. It allows you to log in, log out, and make changes to your online status, online attributes, and privacy settings.
Announcement Service	This service provides the ability to send and receive announcements between Sametime users.
Awareness Service	This service provides notifications of changes in the online status and attributes of users in the community. This service is sometimes referred to as "People Awareness."
Buddy List Service	This service provides the ability to get and set the user contact list stored on the Sametime server and provides conversion methods from String to BL object and vice versa, without having to deal with the low level protocol as defined by the storage service. Using this service ensures the integrity and compatibility of the contact list data.
File Transfer Service	This service provides the ability to send and receive files between Sametime users.
Instant Messaging Service	This service provides one-to-one communication between clients. It is used mostly for Instant Messaging chat between users, but it can also be used to exchange any kind of text or binary data.
Places Service	This service provides the ability to create virtual meeting places that users can enter or leave, see who is in the meeting place, and share activities. This service is sometimes referred to as "Place-Based Awareness." It is a cornerstone of the Sametime architecture.
Storage Service	This service provides server-side storage of user-related data. A user can access his personal data from any Sametime-enabled application. For example, Sametime Connect uses this service to store a user's contact list on the Sametime server so that the user can access it regardless from where he logs in.
Lookup Service	This service provides name resolving and group content lookup.
Directory Service	This service provides directory-browsing services.
Post Service	This service allows posting a message to many users at once.
Token Service	This service allows generation of temporary login tokens.
Names Service	This service provides names and nicknames services.
Multicast Service	This service provides the ability to send messages to multiple recipients.

ANNEXE VIII - Services du toolkit ConnectWebAPI

Table listing parameters for Sametime supported actions.

Action (context root)	Parameters	Description
addgroup	None	Launches Add Group dialog.
addtolist	userId = User contact ID.	Launches the "Add to Sametime Contact List" dialog for the indicated user.
announce	userId = User contact ID or | delimited list of contact IDs	Send an announcement.
call	userId = User contact ID. number = The number you'd like to dial.	Start a telephony call with a user by userid or number. Requires telephony capabilities.
chat	userId = User contact ID.	Launches chat window with user.
getprivacylist	None	Returns the current local user's privacy list.
getstatus	userId = User contact ID jsonp – callback function.	Returns full status information for a specified user or users in JavaScript Object Notation (JSON) format. The status information will include the following items if values can be obtained for them: locationInfo countryName timeZoneId cityName phoneNumber postalCode stateName userDefinedLocationName resolvedName communityId gatewayCommunity hoverText displayName status isExternal statusMessage username contactId id lastChatTime
getstatusshort	userId – User contact ID jsonp = callback function.	Returns a subset of status information for a specified user or users in JSON format, namely: status statusMessage username

		displayName
instantshare	userId = User contact ID.	Start an Instant Share session with a user. Requires Sametime Advanced 8.0 or later
listservices	callback = Callback function.	Returns a list of available actions in JSON format: name path
loggedin	fqdn = the fully qualified domain name of the community server	Returns whether the local user is logged in the given community server. If fqdn is not provided, it returns for the default community.
mystatus	jsonp = callback function.	Returns the current local user's status in JSON format. The status information will include the following items if values can be obtained for them: locationInfo countryName timeZoneId cityName phoneNumber postalCode stateName userDefinedLocationName resolvedName communityId gatewayCommunity hoverText displayName status isExternal statusMessage username contactId id lastChatTime
quickfind	None	Open a mini quickfind window.
remove	userId = User contact ID.	Remove user from the Contact List
rename	userId – The contact ID. nickname – The new nickname.	Rename user from the Contact List.
setstatus	status = The new status. message = Optional status message.	Set the status of the current user on the default community.
startmeeting	userId = User contact ID.	Starts an Instant Meeting with this user. Requires the server is configured for instant meetings.

ANNEXE IX - Extrait des spécifications techniques sur le client web

Client Web

Utilisation du toolkit IM Browser

L'utilisation du toolkit est possible grâce à l'appel aux bibliothèques JavaScript présentes dans celui-ci (revoir chapitre 2.5 du document général des spécifications). Ces bibliothèques vont être incluses dans le fichier web développé (cf0). Le fichier doit être stocké au même endroit que le fichier tunnel.html du SDK (attention, pour un bon fonctionnement du toolkit, il faut reprendre celui utilisé dans le POC, en effet il existe plusieurs versions du tunnel.html entre le SDK 8.5.2 et le 8.5.2 IFR1, la dernière ne fonctionne pas, il faut donc reprendre la plus ancienne).

Dans le cadre du POC ces fichiers ont été déposés sous :

http://cer31-dvlpnots2.cer31.recouv/public/AdminNotes/connectim.nsf/

L'intégration directe dans le plugin sous Eclipse n'a pas fonctionné car le fichier tunnel.html n'est pas détecté lors de l'ouverture du fichier web. La solution de déployer ces fichiers sur un serveur Notes est la plus optimale. Attention à bien déplacer le fichier tunnel.html avec les autres fichiers s'ils doivent être déployés sur un nouveau serveur.

Fichier html

Il existe déjà un client web fonctionnel sur le serveur proxy, il est développé dans le fichier popup.jsp. Il a été pris par défaut pour le POC car il comporte toutes les fonctionnalités exigées pour le client web. Il est cependant possible de le redévelopper sur une base Notes, il faudra bien vérifier que les fonctionnalités sont identiques et conformes aux exigences décrites dans le DCU.

/!\ pour garantir le fonctionnement de l'utilisation de paramètre dans l'url il faut rendre la page publique, faire ouvrir son accès à la base Notes

Popup.jsp

Il se trouve à l'adresse :

http://dev31pacint8.cer31.recouv:9080/stwebclient/popup.jsp

Il faut vérifier que le passage d'un token LTPA en paramètre fonctionne.

ClientWeb.html

La constitution type d'un fichier est décrite dans le paragraphe « Socle de base » du chapitre 2.5 du document général des spécifications. Il est aussi possible de s'inspirer du fichier example7.html développé pour l'affichage d'une fenêtre de chat dans le POC.

- Définition stproxyConfig

```
var stproxyConfig = {

server: "http://dev31pacint8.cer31.recouv:9080",

tunnelURI: "http://cer31-
dvlpnots2.cer31.recouv/public/AdminNotes/connectim.nsf/tunnel.html",

      isConnectClient: false,

            tokenLogin : true

       };
```

Le serveur proxy actuellement mis en place est à l'adresse suivante :
http://dev31pacint8.cer31.recouv:9080

/!\ Lors du déploiement il faudra mettre à jour l'adresse tunnelURI si les fichiers sont déployés sur un nouveau serveur (elle doit être sur le même répertoire que clientWeb.html).

Le positionnement de isConnectClient à false a été choisi dans le sens où l'utilisation de Connect impliquerait l'installation du client Connect sur tous les postes et donc une assistance supplémentaire sur ce type de client.

L'attribut tokenLogin est positionné à true pour éviter l'affichage de la page de connexion à Sametime avant le chargement du client.

- Définition de l'emplacement des bibliothèques du toolkit.

```
<scripttype="text/javascript"src="http://dev31pacint8.cer31.recouv:9
080/stwebclient/latest/dojo.blue/dojo/dojo.js"></script>

<scripttype="text/javascript"src="http://dev31pacint8.cer31.recouv:9
080/stbaseapi/latest/baseComps.js"></script>

<scripttype="text/javascript"src="http://dev31pacint8.cer31.recouv:9
080/stwebclient/latest/widgets.js"></script>
```

En cas de changement d'adresse du serveur proxy il faudra mettre à jour ces adresses.

- Récupération du token

```
functiongetQueryParam(param) {

      var result =  window.location.search.match(

      newRegExp("(\\?|&)" + param + "(\\[\\])?=([^&]*)")

      );
```

```
        return result ? result[3] : false;
        }

//Récupération du paramètre token
var token = getQueryParam("token");
```

Le token est passé en paramètre lors de l'appel du fichier web. Il est récupéré dans l'URL de la page.

- Gestion de la connexion

La connexion est faite avec la fonction loginWithToken dans loginUser(). Pour développer cette partie il est intéressant de relire l'alinéa 5- du chapitre 6 (Fichier html) de la note technique du POC et de s'inspirer du code du fichier example7.html.

Le token LTPA est stocké dans la variable token initialisé précédemment.

- Body

```
<body class="stbody tundra" >

  <div class="stmain">
     <div id="myClient" dojoType="sametime.WebClient" ></div>
  </div>

</body>
```

Pour garantir le fonctionnement des fonctionnalités, l'utilisation de la classe "stbody tundra" dans la balise body est indispensable.

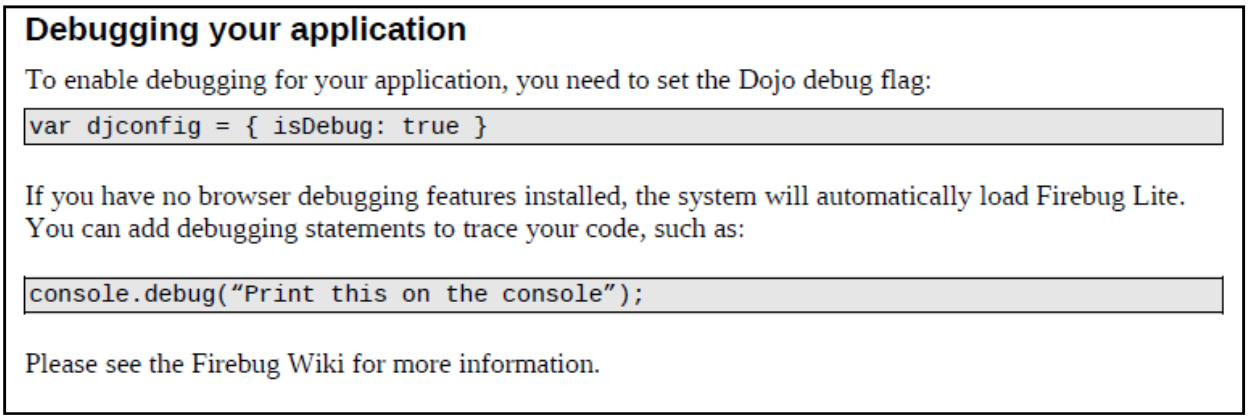
L'intégration du client web se fait grâce à la balise div avec l'attribut dojoType= « sametime.WebClient ».

- Autre

Si le client web créé ne satisfait pas les exigences, il est possible de modifier les paramètres de création du client. Dans ce cas, il faudra créer l'objet « myClient » par programmation JavaScript en définissant dans le corps html de la page un objet Div ayant pour identifiant « myClient » et dans l'entête du fichier le code suivant :

```
dojo.byId("myClient").appendChild(newComponent.domNode);
```

De plus amples informations seront trouvées dans le document STBrowserSDK.pdf du toolkit. La documentation n'étant pas très exhaustive il peut être intéressant d'actionner le mode debug pour mieux comprendre comment est créé un objet par les bibliothèques.



Debugging your application

To enable debugging for your application, you need to set the Dojo debug flag:

```
var djconfig = { isDebug: true }
```

If you have no browser debugging features installed, the system will automatically load Firebug Lite. You can add debugging statements to trace your code, such as:

```
console.debug("Print this on the console");
```

Please see the Firebug Wiki for more information.

[39]

[39] Source p12 du fichier STBrowserSDK.pdf

Bibliographie

AF94	AFCET, Enquête sur la pratique de la collectique (groupware) en france. Rapport d'étude, Septembre, 1994, p. 83
BA04	BALMISSE Gilles, "les enjeux et la réalité française du travail collaboratif et des communautés de pratiques", 2004
BBD04	Farouk Belkadi, Eric Bonjour, Maryvonne Dulmet, "Démarche de modélisation d'une situation de conception collaborative", Document numérique : coopération et organisation numériques (p93-106), 2004
BE09	Djamel Benmerzoug, Modèles et outils formels pour l'intégration d'applications d'entreprises, Thèse en Cotutelle Université Mentouri de Constantine - Faculté des sciences de l'ingénieur - Département d'Informatique, Université Pierre et Marie Curie de Paris 6 - École doctorale : Informatique, télécommunication et électronique, 2009
BLC95	Michel Beaudouin-Lafon and Joëlle Coutaz. Scoop / rapport de recherche 1994-1995. Technical report, SCOOP working group, November 1995.
BO10	Mohamed Boukhebouze, Gestion de changement et vérification formelle de processus métier : une approche orientée règles, Thèse présentée à l'institut national des sciences appliquées de Lyon, 2010
BR08	Patrice Briol, Ingénierie des processus métiers, de l'élaboration à l'exploitation, ISBN 978-1-4092-0040-6, 2008, p201, 202, 261, 139
CAIRN	http://wiki.altair.recouv/doku.php?id=altair:domaines:methodologie_projet:start
CSSH98	Clancey W. J., Sachs P., Sierhuis M., van Hoof R., Brahms: Simulating practice for work systems design. International Journal of Human-Computer Studies, 49: 831-865., 1998
EGR91	ELLIS C. A., GIBBS S.J., and REIN G.L., "Groupware: some issues and experiences", Communications of the ACM, Jan 1991, vol. 34, No 1, pp.39-58
FR00	Fielding, Roy Thomas. Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000
GO09	Claude Godart, Olivier Perrin, Les processus métiers - Concepts, modèles et systèmes, Hermès - Lavoisier, 2009, p52, 18, 32, 27, 276, 276, 107, 43
GR94	Grudin J., CSCW : History and Focus, IEEE Computer 27 P19-26, 1994
JU06	Jungmin Ju, State-of-the-art of Standards in Business Process Modeling and Execution, Division of Mechanical and Industrial Engineering Pohang University of Science and Technology, December 6, 2006
PA06	Papazoglou, M. P. and W. J. van den Heuvel, Service-Oriented Design and Development Methodology, International Journal of Technology (IJWET), Vol. 4, No. 2, 2006, pp. 412-442
RAACO2012	Rapport d'activité ACOSS 2012
RACIRSO0714	Rapport d'activité du CIRSO – Juillet 2014
RBLO	Rémi Badonnel, CoursWebServices2.ppt, Web services, Architecture et Protocoles, LORIA, Ecole des mines de Nancy
SB10	Zohra SBAÏ, Contribution à la Modélisation et à la Vérification de Processus Workflow, Thèse soutenue au Cnam le 13 novembre 2010
SL05	Eddie Soulier et Myriam Lewkowicz, L'intégration des pratiques collaboratives dans la modélisation des processus métier – Une approche « centrée humain» de la conception du SI, Actes de la conférence AIM'05, 2005, Laboratoire CNRS ISTIT/équipe Tech-CICO, Université de Technologie de Troyes
SL07	Leonard Richardson, Sam Ruby, RESTful Web Services, O'Reilly Media, 2007
SN96	Selmin Nurcan, "Analyse et conception de systèmes d'information coopératifs", Recherche Université Paris 1 - Sorbonne, 1996

Wf99 The workflow coalition management coalition, « Workflow Coalition Terminology and Glossary », Rapport numéro WFMC-TC-1011 Issue 3.0 Février 1999

Webographie

ALW http://alwin.developpez.com/tutorial/JavaThread/
BAR http://fr.slideshare.net/baronm/comprendre-le-style-architecture-rest
CLE http://blog.clever-age.com/fr/2006/10/27/soap-vs-rest-choisir-la-bonne-architecture-web-services/
ECL http://eclipse.developpez.com/faq/?page=plateform#definitionView
ERI http://erickstattner.com/enseignement/2013-14/M2_AppliEtServicesWeb_2013-14.pdf
JOE http://www.xml.com/lpt/a/2004/12/01/restful-web.html
JOE2 http://www.xml.com/lpt/a/2005/03/02/restful.html
JOE3 http://www.xml.com/lpt/a/2005/04/06/restful.html
SPO http://spoonless.github.io/epsi-i4-web-services/01_http.html
WIK http://fr.wikipedia.org/wiki/Representational_State_Transfer

Liste des figures

Implémentation de fonctions de travail en mode collaboratif dans une application de workflow procédural en vue de faciliter les échanges entre les différents profils utilisateurs.

Mémoire d'ingénieur CNAM, Toulouse 2015

RESUME

Un changement organisationnel dans l'utilisation du workflow procédural des URSSAF a amené les utilisateurs à nécessiter de moyens de communication plus facile d'accès. Celui-ci pouvant permettre une collaboration plus forte entre les utilisateurs et compenser la perte d'efficacité liée à l'éloignement des utilisateurs entre eux. Il a été choisi d'implémenter l'application de messagerie instantanée Sametime d'IBM déjà déployée aux URSSAF. Celle-ci a fait l'objet d'une étude d'intégration dans le workflow développée en Java sur Eclipse RCP. IBM met à disposition un toolkit composé de plusieurs outils développés à base de service REST mais rendus plus accessibles grâce à des couches plus simples d'utilisation. Une solution utilisant certains de ces outils a été proposée à l'issue du stage.

SUMMARY

Organizational change in the use of the procedural workflow in URSSAF led users to require more accessible capabilities of communication. This may allow stronger collaboration between users and compensate for the loss of efficiency due to the distance between users. It has been decided to implement the Sametime instant messaging application already deployed to IBM URSSAF. This was the subject of a study of integration in the workflow in Java on Eclipse RCP. IBM provides a toolkit composed of several tools developed with REST services but made more accessible through more simple layers. A solution using some of these tools was proposed at the end of the course.

Mots clés : REST, Workflow, Sametime, Eclipse RCP, Applications collaboratives, Processus métiers, SDK, Messagerie instantanée

Key words : REST, Workflow, Sametime, Eclipse RCP, Collaborative software, Business Process, SDK, Instant messaging