Modélisation des processus utilisateurs à partir des traces d'exécution, application aux systèmes d'information faiblement structurés

Marwa Hamdi

LA ROCHELLE UNIVERSITÉ

ÉCOLE DOCTORALE EUCLIDE

LABORATOIRE L3i

THÈSE présentée par :

Marwa TRABELSI

soutenue le : **01/09/2022**

pour obtenir le grade de : **Docteur de La Rochelle Université**

Discipline : **Informatique et Applications**

Modélisation des processus utilisateurs à partir des traces d'exécution, application aux systèmes d'information faiblement structurés.

Rapporteurs	Sébastien George	Professeur des universités	Le Mans Université
	Agnès Front	Professeur des universités	Université Grenoble Alpes
Examinateurs	Adam Jatowt	Professeur des universités	University of Innsbruck
	Koraljka Golub	Professeur des universités	Linnaeus University
	Antoine Doucet	Professeur des universités	La Rochelle Université
	Cyrille Suire	Enseignant Chercheur	La Rochelle Université
Direction	Ronan Champagnat	Maître de conférences HDR	La Rochelle Université
	Jacques Morcos	Maître de conférences	La Rochelle Université

Résumé

La nécessité pratique d'étudier la manière dont les utilisateurs interagissent avec les systèmes d'information en considérant leurs traces numériques augmente considérablement. En effet, les processus métiers fournis par les entreprises au travers de ces systèmes sont morcelés et laissent le plus souvent les utilisateurs déterminer seuls leur parcours pour atteindre leur objectif. Dans un tel contexte, le parcours de l'utilisateur en vue d'exécuter une tâche (achat d'un produit sur un site marchand, rechercher un document dans une bibliothèque numérique, etc.) correspond à un « processus non structuré ». La signification, la structure et les résultats varient en fonction des pratiques.

Ce travail de recherche porte sur l'extraction de parcours utilisateurs dans des systèmes d'information avec des processus métier faiblement structurés. Nous nous penchons plus particulièrement sur le cas des bibliothèques numériques. Les bibliothèques numériques sont des systèmes complexes et avancés qui attirent une multitude d'utilisateurs pour des tâches diverses de recherche d'information. Ces dernières stockent et gèrent une quantité importante de documents numériques et de documents numérisés. Elles offrent à leurs utilisateurs de nombreux services comme le système de recherche d'informations, la personnalisation, etc. Dans le cadre de cette thèse, afin de modéliser les différents parcours menant les utilisateurs à la résolution de leurs tâches de recherche d'information, nous avons choisi d'utiliser les techniques de fouille de processus.

La fouille de processus offre divers moyens pour découvrir, évaluer la qualité et améliorer les modèles de processus minés à partir des traces d'exécution. L'avantage principal des techniques que nous employons réside dans leur capacité à traiter le processus de recherche d'information dans son ensemble (un schéma de navigation complet ayant une activité de début et une activité de fin). De ce fait, un modèle décrivant les interactions des utilisateurs peut aider les concepteurs des systèmes à répondre efficacement aux besoins des utilisateurs, d'une part, et à leur présenter un ensemble de recommandations, d'autre part. En outre, comprendre et modéliser les interactions des utilisateurs peut aider les concepteurs à optimiser leurs systèmes et à améliorer la qualité de leurs services. Cependant, la découverte des modèles de comportements à partir d'un énorme nombre de traces réelles conduit à des modèles complexes difficilement exploitables.

Dans le cadre de cette thèse nous proposons une méthode pour identifier les parcours des utilisateurs de la bibliothèque numérique *Gallica*, le portail web de la Bibliothèque Nationale de France. Tout d'abord nous déterminons si la fouille de processus nous permet d'extraire de la connaissance sur les parcours utilisateurs d'une bibliothèque numérique. Pour cela nous réalisons une étude exhaustive sur les techniques de la fouille des processus afin de déterminer leurs capacités de découvrir des modèles pertinents sur la base des traces réelles de taille raisonnable issues d'une bibliothèque numérique. Nous montrons que la fouille de processus permet d'extraire des modèles de comportements pertinents et que l'algorithme *Inductive Miner* est le plus performant.

Cependant, si nous souhaitons passer à un cas réel, comme les traces brutes de navigation des utilisateurs de *Gallica*, découvrir directement des modèles à l'aide des algorithmes de la fouille des processus ne sera pas possible. Ces traces brutes ne sont pas directement exploitables. Nous proposons donc une méthodologie de transformation des traces brutes vers des traces modélisées. Ce pré-traitement consiste à nettoyer les traces afin d'éliminer les requêtes non

pertinentes (liées au routage) ainsi que les traces laissées par les robots, puis à étiqueter les logs afin de faire apparaître les activités des utilisateurs et à sessioniser, c'est-à-dire à découper les logs de connexion en sessions utilisateur, et ainsi obtenir un modèle d'évènements utilisable par les algorithmes de la fouille de processus.

En outre, pour de telles traces réelles volumineuses, complexes et non structurées, il est établi que le résultat de l'extraction des processus fait apparaître des modèles qui contiennent peu d'informations pertinentes. Nous avons proposé une nouvelle méthode de regroupement des parcours similaires des utilisateurs des bibliothèques numériques en tenant compte des caractéristiques existantes dans les traces avant de passer à la modélisation. Cette méthode est basée sur l'extraction et l'encodage des sous-séquences fréquentes. Nous validons notre méthode en la comparant avec deux couramment observées dans l'état de l'art et en utilisant un jeu de données simulé inspiré d'une bibliothèque numérique.

Nous finissons en proposant une méthodologie afin de pouvoir appliquer notre approche sur les traces réelles issues de *Gallica*. Les expérimentations montrent que notre méthode surpasse celles existantes à la fois au niveau du regroupement et de la modélisation des parcours sur l'ensemble de données synthétiques. De plus elle génère des modèles exploitables qualitativement sur les données réelles.

Mots-clés : Systèmes d'information, Découverte des modèles de comportements, Fouille de processus, Bibliothèque numérique, Regroupement des traces.

User's Journey modeling Based on Process Mining for Information Systems with Unstructured Processes

Abstract

The practical need to study how users interact with information systems by considering their digital traces is significantly increasing. Indeed, the business processes provided by companies through these systems are fragmented and most often leave users to determine their own path to reach their goals. In such a context, the user's journey to perform a task (purchase of a product on an e-commerce website, search for a document in a digital library, etc.) corresponds to an "unstructured process". The meaning, structure and results depend on the user's skills. This work focuses on the extraction of users' journeys in information systems with unstructured business processes. We investigate more particularly the case of digital libraries.

Digital libraries are complex and advanced systems that attract a multitude of users for various information retrieval tasks. They store and manage a large amount of digital documents and objects. They offer to their users many services such as the information retrieval system, personalization, etc. In order to model the different paths leading users to the resolution of their information retrieval task, we chose to use process mining techniques.

Process mining offers a variety of techniques to discover, assess quality, and improve process models mined from execution traces. The main advantage of these techniques is their ability to handle the whole information retrieval process (the sequence of activities carried out by users from the beginning to the end of their navigation). Therefore, a model depicting the users' interactions can help system designers to answer users' practical requirements, on the one hand, and to present a set of recommendations to them, on the other hand. Moreover, understanding and modeling user's interactions could be useful to optimize systems' design and improve the most used features. However, the large number of generated logs leads to complex models that are not generic for all users and do not allow achieving all their objectives.

In this thesis, we proposed a new method, using process mining techniques, to identify the user paths in a digital library, for instance, Gallica, the web portal of the National Library of France. The first question to address in our work is the ability of process mining techniques to discover relevant models from digital library users' journeys. After performing an exhaustive study on real traces of reasonable size, we showed that process mining techniques are fitted to extract relevant behavioural models and that the Inductive Miner algorithm is the most efficient.

However, applying process mining algorithms directly to a real case, such as the raw browsing traces of Gallica users, will not be possible. These raw traces are not directly exploitable. We, therefore, proposed a 3-steps methodology to transform raw traces into modelled traces. The first step of this process consists of cleaning the traces in order to eliminate irrelevant requests (related to routing) as well as the traces left by the robots, the second is labelling the logs in order to show user activities and finally splitting the connection logs into user sessions (sessionization), and thus obtain an event model that can be used by the process mining algorithms.

Furthermore, for such large, complex and unstructured real traces, process mining techniques usually return "spaghetti-like" models, which provide limited support to analysts. To remedy this issue, we proposed a new method for clustering similar users' journeys in digital libraries by taking into account the existing characteristics in the traces before proceeding to modeling. Our method is based on the encoding of the extracted frequent subsequences. We compared our proposed method to two state-of-the-art methods over a synthetic resource manually annotated

used for validation.

Finally, we discuss the adaptability of our method and give a way to be used in the digital libraries and more specifically in Gallica. The experiments to assess the performance of our proposed method were carried out over a synthetic dataset as well as samples of real data. The obtained results proved that our method outperforms existing methods in both clustering and modeling users' journeys and generates interesting models on real-world data.

keywords : Information systems, User's journey modeling , Process mining, Digital library, Users' traces clustering.

Remerciements

Tout d'abord, je tiens à remercier mon directeur de thèse, Ronan Champagnat, pour son suivi, sa disponibilité, ses conseils et pour les discussions enrichissantes que nous avons partagées durant ces quatre années de thèse. Ses suggestions m'ont permis de prendre les bonnes décisions et de mener à bien ce travail. Toute ma gratitude va également à mon co-directeur de thèse Jacques Morcos, son soutien et son encadrement m'ont été très utiles dans l'élaboration de ce travail.

Ensuite, je voudrais remercier mes rapporteurs Agnès Front et Sébastien George d'avoir accepté de rapporter cette thèse et pour l'intérêt qu'ils ont accordé à mon travail. Je tiens également à remercier les autres membres de mon jury, Antoine doucet, Adam Jatowt et Koraljka Golub pour avoir accepté d'évaluer mon travail.

Mes remerciements vont également aux membres de l'équipe E-adapt du L3i, pour leur soutien durant ces travaux de thèse et le cadre de travail toujours agréable qu'ils ont su m'offrir. Je remercie tout particulièrement Cyrille Suire pour ses conseils, son implication, ses réflexions pour l'organisation des expérimentations de ce travail ainsi que ses relectures avisés de mes recherches.

Je remercie par ailleurs tous mes collègues admirables, que j'ai eu la chance de les rencontrer à l'instar de : Thomas, Salah, Muzzamil, Beatriz, Adyson, Marie-neige, Viviana, Imane, Damien, Jeremy, Lionel ... Merci à tous pour la bonne ambiance, les discussions, le soutien...

Merci également, à toute l'équipe du département informatique de l'IUT de La Rochelle, avec qui j'ai appris à enseigner durant mon contrat doctoral et avec qui j'ai eu la chance de vivre une année complète durant mon ATER.

Enfin, et surtout, je ne remercierai jamais assez mon époux Ahmed pour son soutien quotidien indéfectible et son enthousiasme contagieux à l'égard de mes travaux comme de la vie en général. Mes remerciements vont également à mes parents pour leur soutien sans faille ainsi qu'à mon cher fils, ma famille et ma belle famille pour leurs encouragements, sans oublier tous mes chers amis.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Contexte et motivations

Avec la transformation digitale, il est devenu difficile pour les entreprises d'évoluer en marge des solutions technologiques. L'intégration d'un système d'informations se situe dans ce sillage. Elle consiste à mettre en place un système de gestion automatisé qui donne accès à des données bien plus pertinentes. Le système d'information aide à mieux répondre aux besoins des utilisateurs en mettant en place des données plus accessibles. Par ailleurs, le système d'information est aussi un outil social. Il permet de mettre toutes les informations utiles à la disposition des utilisateurs. Ces derniers peuvent, en effet, consulter à tout moment les données utiles et les utiliser selon leurs besoins.

Par ailleurs, les systèmes d'information sont des systèmes comprenant des processus métier (c'est-à-dire une succession d'activités permettant d'atteindre un objectif). Par exemple lorsque nous voulons faire un achat d'un produit sur un site e-commerce, nous effectuons une suite d'opérations bien déterminée. Bien que ces systèmes soient établis par des modèles de processus explicites, une des difficultés est née du fait que les processus ne sont pas tous clairement définis. Cela est dû également à la diversité des tâches, des intervenants (concepteurs, utilisateurs, gestionnaires, etc.) et d'autres paramètres imprévisibles qui se manifestent après l'établissement du processus (besoins des utilisateurs, échec ou exception d'exécution inattendues, etc.).

En théorie, les modèles initialement définis par les concepteurs de ces systèmes sont censés reproduire le comportement suivi par les utilisateurs lorsqu'ils sont en situation de recherche d'information. Malgré les efforts sur l'ergonomie et les services d'aides fournis, les utilisateurs parviennent toujours à trouver des cheminements complexes différents des chemins prévus théoriquement par les concepteurs. Les utilisateurs ont tendance à construire leurs parcours de recherche en fonction de leurs objectifs et leurs comportements (Piccoli, 2007). De plus, ils peuvent se

trouver dans des situations d'échecs et de stress, ce qui rend l'utilisation du système moins pertinente.

La figure 1 [1], par exemple, montre que le modèle théorique proposé par les concepteurs pour le besoin de paiement de leurs factures n'est composé que de trois étapes consécutives alors que les utilisateurs ont suivi d'autres chemins pour réaliser ce besoin. Cela implique que le modèle des concepteurs pour passer d'une étape à une autre n'est toujours pas le plus pratique ni peut-être le plus simple pour certains utilisateurs. De son côté, l'utilisateur ne fait généralement pas l'effort de comprendre le modèle conçu par le concepteur Hansen and Järvelin (2005). Beaucoup d'utilisateurs trouvent que les systèmes d'information sont compliqués et destinés à des utilisateurs experts et optent rapidement pour demander la réalisation de leurs services auprès des structures physiques telles que les banques, les boutiques ou les entreprises concernées.

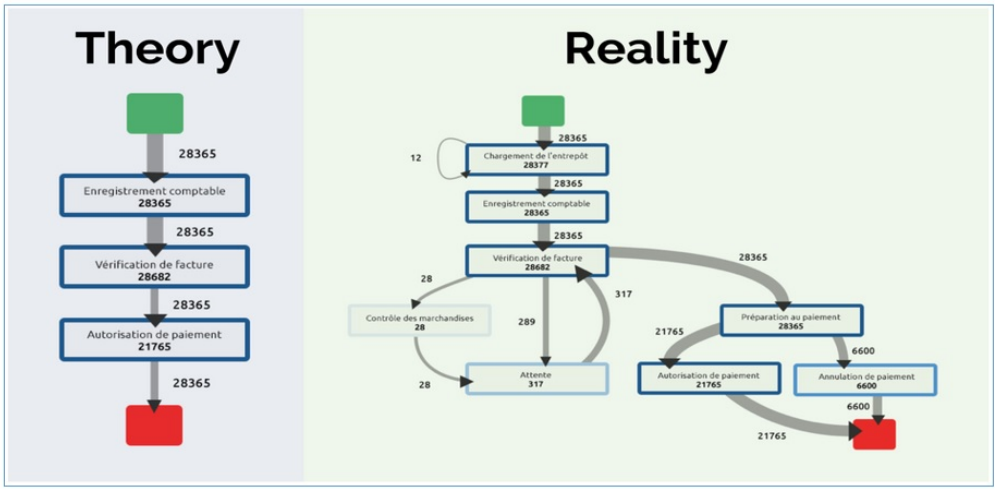


FIGURE 1 – Différence entre un modèle théorique et le modèle suivi par les utilisateurs d'un processus de paiement de facture

Par conséquent, les recherches se sont orientées vers l'analyse du comportement des utilisateurs des systèmes d'information. Cette analyse est basée sur l'interprétation d'informations, recueillies pendant les sessions de navigations, appelées traces. Ces traces, définies comme une séquence temporelle d'activités, fournissent des connaissances sur les cheminements réalisés par les utilisateurs.

Nous abordons dans cette thèse des systèmes d'information complexes et avancés avec des processus métier faiblement structurés qui engendrent des traces de

1. `https://www.logpickr.com/fr/accueil.html`

navigation à la fois homogènes et hétérogènes, complètes et incomplètes, on parle ici des processus d'usage faiblement structurés. Parmi les systèmes d'information qui correspondent à ces caractéristiques, nous trouvons les bibliothèques numériques. Ces systèmes stockent, recensent et assurent la disponibilité des ressources. Ils mettent en œuvre l'accessibilité de contenus, des documents numériques ou numérisés, qui se comptent en millions voire en milliards et qui ne cessent d'évoluer. Ces dispositifs offrent toute une diversité de fonctionnalités complémentaires telles que la personnalisation, l'extraction du texte à partir des images, le stockage des ressources, etc.

Les utilisateurs des bibliothèques numériques en situation de recherche d'information adoptent des comportements qui dépendent de nombreux facteurs, depuis leur niveau de connaissance du domaine jusqu'à leurs compétences spécifiques. Ces variables se traduisent par des processus de résolution d'une tâche de recherche d'information qui peuvent être variés. Afin d'étudier les différents comportements des utilisateurs, suite à leurs interactions avec une bibliothèque numérique, nous utilisons les techniques de fouille de processus.

À notre connaissance, la fouille des processus n'a jamais été utilisée dans la modélisation des comportements d'utilisateurs des bibliothèques numériques. Les techniques de fouille de processus sont capables d'extraire des connaissances à partir des événements de traces d'exécution communément disponibles dans les systèmes d'information actuels. Ces techniques offrent de nouveaux moyens pour découvrir, surveiller et améliorer les processus dans une variété de domaines d'application (Van der Aalst, 2016). L'intérêt croissant pour la fouille de processus est justifié, d'une part, par le nombre énorme de traces enregistrées fournissant, par conséquent, des informations détaillées sur l'histoire du processus et, d'autre part, par la capacité de ces techniques de traiter le processus dans son ensemble (un processus complet ayant une activité de début et une activité de fin). Elles diffèrent ainsi des techniques employées, par exemple, par Nouvellet et al. (2017), qui ont l'avantage de permettre une fouille de données des sessions utilisateurs, mais se limitent à la déduction d'un modèle fondé sur une chaîne de *Markov* d'ordre 1 et non pas sur l'ensemble de la séquence d'activités réalisées par l'utilisateur.

L'objectif de cette thèse consiste à extraire des modèles de parcours utilisateurs dans des systèmes d'information avec des processus métiers faiblement structurés. Nous choisissons comme cas d'étude les bibliothèques numériques car elles correspondent aux systèmes d'information qui révèlent les défis, en matière d'extraction de parcours, les plus difficiles. Nous choisissons en particulier d'aborder la recherche

de parcours utilisateurs au sein du site *Gallica*[2], librairie numérique de la Bibliothèque Nationale de France, car il contient un volume de traces conséquent. Notre méthodologie doit rassembler les différentes variantes que l'on peut trouver dans les traces d'exécution. En d'autres termes, nous cherchons à identifier des familles, ou faisceaux, ou traces similaires, de trajectoires qui émergent spontanément des comportements des utilisateurs. Cette analyse permettra de simplifier l'utilisation de ce genre de systèmes, prédire les fonctionnalités utiles et recommander les services intéressants qui peuvent échapper à un simple utilisateur. Cela va aussi permettre aux concepteurs d'améliorer leurs systèmes en prenant en considération les cheminements réels des utilisateurs.

Questions de recherche

Cette thèse a été l'occasion pour nous de proposer et d'étudier un certain nombres de concepts liés à l'analyse et la modélisation des parcours utilisateurs dans les bibliothèques numériques en utilisant la fouille de processus. D'une part, nous avons étudié l'état de l'art sur les quatre axes de recherches liés à cette problématique, à savoir : l'analyse et la modélisation des comportements de navigation sur le web, les caractéristiques de la navigation dans une bibliothèque numérique ainsi que la fouille de processus et le regroupement des traces dans le domaine de la fouille de processus. Nos contributions ainsi que nos questions de recherche se déclinent en quatre points :

A) La première question que nous nous sommes posée porte sur la capacité de la fouille des processus à extraire des modèles pertinents qui couvrent efficacement les différents parcours utilisateurs des bibliothèques numériques. Pour répondre à cette question nous avons étudié les capacités d'extraction des modèles de parcours des principaux algorithmes de découverte de processus appliqués à des traces de navigation provenant d'une expérimentation réalisée sur une bibliothèque numérique.

B) Notre deuxième question de recherche consiste à déterminer la possibilité de transformer les logs des interactions utilisateurs provenant des bibliothèques numériques afin d'extraire les informations nécessaires pour les rendre exploitables par les algorithmes de la fouille de processus. Nous nous plaçons

2. `https://gallica.bnf.fr/`

dans le contexte de la bibliothèque numérique *Gallica* qui est soumise à un fort trafic (plusieurs centaines de millions de requêtes par mois).

C) Notre troisième question de recherche consiste à déterminer s'il est possible de proposer une méthode de regroupement de traces volumineuses, complexes et non structurées, en tenant compte de la qualité des groupes obtenus afin d'améliorer le temps d'exécution ainsi que la qualité des modèles de parcours utilisateurs découverts. Pour cela, nous proposons une méthode basée sur une hybridation entre l'analyse des séquences et le regroupement des traces. Elle est basée sur l'extraction et l'encodage des **S**ous-**S**équences **F**réquentes (*FSS-encoding*) trouvées dans les traces de navigations. Nous proposons une nouvelle mesure afin d'effectuer l'encodage des sous-séquences fréquentes en se basant sur des caractéristiques bien définies. Nous validons notre méthode en la comparant avec d'autres méthodes, identifiées dans l'état de l'art, sur un jeu de données simulé d'une bibliothèque numérique.

D) La dernière question de recherche consiste à déterminer le volume de traces nécessaire pour extraire des regroupements de parcours utilisateurs pertinents. Nous définissons une méthodologie afin de choisir l'échantillon de traces qui couvrent les différents types de parcours utilisateurs et qui nous permet d'obtenir des modèles de comportements pertinents.

Organisation du manuscrit

Ce manuscrit de thèse est organisé en six chapitres. À la suite de cette introduction où nous exposons notre problématique, nos enjeux et la volonté de construire une solution applicable sur un cas réel, le premier chapitre est consacré a une étude de l'état de l'art afin de justifier le choix de la méthode d'extraction de modèles que nous avons fait. Les chapitres suivants traitent de problématiques variées et ils incorporent les états de l'art nécessaires à leur compréhension.

Le chapitre 1 présente une étude des différentes méthodes existantes d'analyse et modélisation des parcours utilisateurs dans un système d'information en se basant sur leurs traces de navigation. Nous étudions deux domaines de fouille pour extraire des modèles de parcours utilisateurs : le domaine du web et la fouille de processus métiers. Le chapitre 2 présente, ensuite, le domaine des bibliothèques numériques. Nous identifions les spécificités des parcours utilisateurs lors de la recherche d'information. Le chapitre 3 est consacré à la fouille de processus. Nous présentons les principes et démarches existants pour la découverte de processus

métier, puis nous menons une étude comparative des différents algorithmes de fouille de processus afin d'identifier le ou les algorithmes les plus pertinents pour notre cas d'étude. Cette étude nous donne des contraintes quant au format des données utilisées pour la fouille de processus. Dans le chapitre suivant, chapitre 4, nous décrivons en détail les traces de navigation issues des interactions des utilisateurs avec *Gallica*, puis notre méthodologie de passage des traces brutes issues des bibliothèques numériques à des traces modélisées exploitables par les algorithmes de la fouille des processus. Le chapitre 5 est consacré à la définition d'une nouvelle méthode de regroupement des traces basée sur l'encodage des sous-séquences fréquentes puis nous validons notre méthode à l'aide d'un ensemble des traces simulées d'une bibliothèque numérique et nous la comparons à des méthodes de regroupement du même type. Le chapitre 6, est dédié à l'application de notre méthode sur des données réelles issues des traces de navigations des utilisateurs de *Gallica* et nous proposons une évaluation quantitative et qualitative des modèles obtenus en fonction du volume des traces étudié. Le dernier chapitre, chapitre 6.4 conclut la thèse et présente les perspectives.

Chapitre 1

Analyse et modélisation des comportements des utilisateurs dans les systèmes d'information

Sommaire

1.1 Introduction

Comme nous l'avons évoqué dans l'introduction, l'analyse et la modélisation des comportements des utilisateurs est un exercice extrêmement utile pour les entreprises disposant d'un système d'information. Cette analyse leur permet de comprendre le modèle réel de la navigation de leurs utilisateurs et de le comparer au modèle initial proposé dans le but d'améliorer leur système d'information. Le modèle initial de navigation est conçu par les concepteurs. Il est unique et adressé à tous les utilisateurs quelque soit leur type du novice jusqu'à l'expert. L'analyse de la manière dont les utilisateurs effectuent leur parcours pour évaluer et satisfaire leurs besoins permet alors de concevoir le modèle adapté à chaque utilisateur et de suggérer une expérience de plus en plus personnalisée à ses préférences.

La modélisation des comportements des utilisateurs permet, ainsi, de cartographier le cheminement mental de l'utilisateur et les questionnements associés. L'intérêt consiste alors à imaginer les réponses que l'interface doit être en mesure d'apporter à chaque parcours. Ces réflexions peuvent ensuite nourrir le travail de conception qui posera les bases pour la définition des parcours principaux des utilisateurs (Poggi et al., 2013). Nous retrouvons cette notion de parcours utilisateur dans les travaux principalement liés aux domaines de la **fouille d'usage du web** (Talakokkula, 2015), de l'**analyse du web** (Kumar et al., 2012) ainsi que de la **fouille de processus** développée par Aalst (Van der Aalst, 2016).

Dans ce chapitre, nous allons présenter dans la section 1.2, les notions de base des comportements des utilisateurs des systèmes d'information et la liaison entre ces comportements et leurs parcours de navigation. Nous définissons, par la suite, plus concrètement la notion de parcours utilisateurs dans la section 1.3. La section 1.4 fera l'objet d'un succinct état de l'art sur les méthodes d'analyse et de modélisation des comportements des utilisateurs basées sur leurs parcours. Ce chapitre se termine par une synthèse des outils existants pour analyser et modéliser le comportement d'un utilisateur dans un système d'information. Nous présentons le bilan d'utilisation des différentes méthodes dans la section 1.5.

1.2 Comportements des utilisateurs

Le comportement d'un utilisateur au sein d'un système d'information est caractérisé par sa manière d'interagir avec ce dernier pour répondre à ses intentions et satisfaire ses besoins. Il permet l'adaptation du système d'information dans

sa stratégie de recherche d'information. Le comportement des utilisateurs est influencé par plusieurs facteurs à l'instar de sa situation à l'instant de la navigation, sa maîtrise du système, ses connaissances sur le sujet et même parfois sur ses compétences informatique.

Chaque utilisateur possède sa propre individualité, son affectivité, sa culture et ses connaissances du système. Ces facteurs parmi d'autres décrivent le comportement d'un utilisateur au sein d'un système d'information. L'analyse des comportements permet de catégoriser les utilisateurs et de proposer des modèles communs aux profils de comportements similaires. Les premiers travaux qui analysent les comportements des utilisateurs étaient basés sur des méthodes d'entretien dans laquelle les interrogés divulguent leurs perceptions, leurs pensées et leurs points de vue sur leurs expériences avec les systèmes d'informations (Whyte and Bytheway, 1996). Cet entretien avait pour objectif de trouver les facteurs qui affectent le « succès » des systèmes d'information. Néanmoins, cette méthode n'est pas destinée à un public hétérogène mais, au contraire, elle est adressée à une catégorie d'utilisateurs particulière –des hommes d'affaires– et a recueilli leurs pensées et opinions concernant les systèmes d'information qu'ils utilisent. Les réponses collectées ont permis la catégorisation des comportements en *accidentel* et *naïf*. Bien que la méthode d'entretien semble convenir parfaitement à obtenir des réponses fiables, elle présente plusieurs inconvénients. Premièrement, la méthode nécessite d'énormes ressources financière et institutionnelle. Deuxièmement, elle ne peut pas cibler une catégorie variée d'utilisateurs avec des cas d'utilisation différents. Troisièmement, ce travail manuel laborieux ne peut pas satisfaire toutes les contraintes actuelles telles que l'analyse automatique en temps réel et périodique.

Dans le cadre de notre travail, l'analyse des comportements des utilisateurs doit être fondée sur leurs parcours afin de concevoir et optimiser les modèles proposés par les systèmes d'information. Dans la conception, elle permet l'évaluation des opportunités et des risques ainsi qu'une meilleure compréhension des futurs utilisateurs. Dans l'optimisation, elle permet de remettre à plat la qualité d'un produit ou service, de prendre du recul sur sa qualité et les pistes d'amélioration.

Aux sein des systèmes d'information, l'impact des comportements des utilisateurs sur leurs parcours est remarquable. Les hésitations chez les utilisateurs, par exemple, se traduisent par un retour récurrent en arrière dans leurs navigations (retour à l'étape précédente dans une application web). En outre, la sollicitation fréquente et rapide de l'interface d'accueil au milieu d'un parcours peut potentiellement indiquer une maîtrise insuffisante du système. C'est pour cette raison que les concepteurs des systèmes d'information cherchent toujours à *connaître l'utilisateur*

afin de répondre à leurs besoins. Le comportement des utilisateurs nous parait le concept le plus pertinent pour *connaître l'utilisateur* et ainsi atteindre l'objectif d'un système d'information qui consiste à offrir les outils (processus et données) nécessaires pour permettre à un utilisateur de réaliser son activité (tâche/objectif, etc.).

1.3 Parcours utilisateurs

Le parcours utilisateur (souvent connu par le terme anglais *Customer's Journey*) est l'ensemble d'actions que les utilisateurs effectuent dans leurs interactions avec un système d'information. Ces parcours peuvent être aussi considérés comme des procédures visuelles, orientées processus, qui permettent d'analyser les expériences des utilisateurs. Dans le cadre d'un site web marchand, par exemple, le parcours se déclenche généralement lorsque l'utilisateur désire un produit ou un service et se poursuit jusqu'à l'obtention du produit ou du service (Poggi et al., 2013). Selon Djouad et al. (2010), le parcours utilisateur dans un système d'information, est le chemin parcouru par l'utilisateur du système de son arrivée à son départ même si aucun besoin particulier n'est achevé.

Techniquement, le parcours utilisateur se représente par une suite de clics de souris et de saisies sur un clavier. Ces informations déclenchent des requêtes qui ont pour résultat l'affichage de certaines pages du site. Lorsqu'un utilisateur navigue sur les pages web d'un système d'information, le navigateur envoie des requêtes au serveur demandant les ressources auxquelles l'utilisateur a accès. Chaque demande est enregistrée par le serveur dans des fichiers logs. Ces fichiers constituent des journaux d'utilisation et capturent l'historique des évènements sous forme d'un enregistrement séquentiel des processus réalisés par chaque utilisateur. En effet chaque enregistrement (ligne) dans les fichiers logs représente une requête pour une ressource. Il est ainsi possible d'identifier des informations telles que l'adresse *IP* du visiteur, la date et l'heure d'accès, l'*URL*[1], le code d'état renvoyé par le serveur, les octets transférés à l'utilisateur, etc.

Les systèmes d'information sont couramment l'environnement idéal pour explorer le concept de suivi du parcours utilisateur. L'objectif de la section suivante sera la description des différentes méthodes existantes pour l'analyse et la modélisation des comportements des utilisateurs des systèmes d'information en se basant

1. Le principe de base du web est que chaque ressource (document ou site) est identifiée par une adresse unique : une *URL* (*Uniform Ressource Locator*) de la page consultée.

sur leurs traces/parcours de navigations.

1.4 Étude de l'existant

Les méthodes d'analyse et modélisation des comportements des utilisateurs peuvent être regroupées en trois grandes catégories. Nous décrivons les méthodes de **fouille d'usage du web** dans la section 1.4.1, d"**analyse du web** dans la section 1.4.2 et de **fouille des processus** dans la section 1.4.3.

1.4.1 Fouille d'usage du web (*Web Usage Mining*)

La fouille d'usage d'un site web ou d'un système d'information, plus connue sous le nom du *Web Usage Mining*, est le processus d'extraction d'information à partir de l'historique de parcours utilisateurs (Talakokkula, 2015). Lors de sa visite d'un site web, l'utilisateur laisse derrière lui des traces sur sa navigation. Ces traces sont collectées, analysées et stockées dans les fichiers logs, ce qui permet de comprendre le comportement de l'utilisateur et d'améliorer ensuite la structure du site web. Le *Web Usage Mining* se focalise généralement sur des techniques qui pourraient prédire le comportement de l'utilisateur pendant qu'il interagit avec le web. Plus précisément, ces techniques consistent à découvrir ce que les utilisateurs cherchent sur internet et leurs manières d'utiliser les systèmes d'information. Par exemple, certains utilisateurs ne recherchent que des données textuelles, tandis que d'autres sont intéressés par des données multimédia alors que d'autres s'intéressent à des données multimodales. Le *Web Usage Mining* permet également de déterminer les pages les plus visitées par les utilisateurs, celles consultées en même temps ou l'une après l'autre ou encore le profil utilisateur ayant accéder à une catégorie spécifique de sites web et dans quel but.

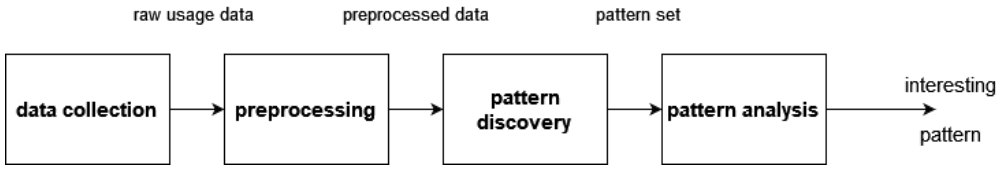


FIGURE 1.1 – Les phases du *Web Usage Mining* (Talakokkula, 2015)

La figure 1.1 présente les étapes réalisées dans le cadre du *Web Usage Mining*. Suite à leur collection, les données passent par trois étapes de traitement.

Nous décrivons dans les sous-sections suivantes ces trois phases : le pré-traitement, l'exploration ou la découverte des motifs et l'analyse des motifs découverts.

1.4.1.1 Pré-traitement (*Preprocessing)*

Le pré-traitement des enregistrements dans les fichiers logs a pour objectif le nettoyage et la structuration des données pour les préparer à une analyse d'usage. Cette étape est souvent la plus laborieuse et qui demande le plus de temps à cause de l'importante présence du bruit et l'absence d'une structuration claire de données. Le pré-traitement doit distinguer les requêtes effectuées par des robots web et les supprimer des fichiers logs afin de ne garder que les requêtes des utilisateurs humains. Ces requêtes doivent être nettoyées de toute information liées à leurs développement comme les entêtes *HTML/CSS* ou les objets images. Nous revenons en détail sur le pré-traitement que nous avons effectuées sur nos données dans le chapitre 4. Les requêtes conservées doivent ensuite être structurées en sessions. Une session désigne la visite d'un utilisateur d'un ensemble de pages web pendant une durée quelconque sans que la navigation ne soit interrompue. Le pré-traitement constitue donc une série d'actions sur les fichiers logs, à savoir la conversion et le nettoyage de données et l'identification des utilisateurs et des sessions de navigation (Michel, 2002; Mughal, 2018).

1.4.1.2 Découverte des motifs (*Pattern Discovery*)

La technique de découverte des motifs est considérée comme une composante clé dans le *Web Usage Mining*. Suite au pré-traitement, certaines techniques de la fouille des données (*Data Mining*) sont utilisées pour découvrir des modèles et des motifs intéressants. La découverte des motifs consiste en l'application des techniques de la fouille de données pour découvrir des modèles d'utilisation du web afin de mieux comprendre et répondre aux besoins de l'utilisateur. Généralement, les modèles de comportement représentent les relations entre les pages web d'un site particulier, les classes et les différents groupes d'utilisateurs, les règles d'association ainsi que les informations personnelles sur les personnes qui naviguent sur le site (Talakokkula, 2015; Patil and Patil, 2012).

Parmi les techniques de la fouille des données qui ont été utilisées pour la découverte des motifs, nous pouvons citer les techniques suivantes :

- l'analyse des séquences (Masseglia et al., 2006; Jalali et al., 2010)
- les règles d'associations (Kumar and Rukmani, 2010)

- la classification (Patil and Patil, 2012; Santra and Jayasudha, 2012; Talakokkula, 2015)
- le regroupement (Srivastava et al., 2000; Patil and Patil, 2012; Talakokkula, 2015; Nouvellet et al., 2017)
- l'analyse statistique (Srivastava et al., 2000)

Nous allons présenter ces techniques dans les paragraphes suivants.

L'analyse des séquences (*Sequence Mining*)

Cette technique consiste généralement à trouver les occurrences répétées dans un ensemble de données. Dans le cadre des données de la navigation web, l'analyse des séquences sert à extraire les séquences fréquentes d'activités réalisées par un simple utilisateur durant une session de navigation. Pour un site e-commerce, par exemple, l'objectif de cette technique peut être de découvrir un ensemble d'articles qui sont fréquemment commandés dans une seule transaction (Masseglia et al., 2006). La découverte des modèles séquentiels permet ainsi d'identifier les tendances utiles des utilisateurs. Plusieurs algorithmes d'analyse de séquences ont été utilisés pour l'extraction des séquences fréquentes à partir des parcours de navigation. Jalali et al. (2010) ont utilisé l'algorithme des plus longues sous-séquences communes (*Longest common subsequences* (LCS)) pour l'extraction des patterns de navigations. Afin de classer les sessions des utilisateurs, ils ont cherché les modèles qui contiennent le plus grand nombre de pages web similaires dans chaque session.

Les règles d'association (*Association Rules*)

Toujours dans le cadre des données de la navigation web, la découverte des motifs à l'aide des règles d'association consistent à découvrir les corrélations entre les pages web. Ces pages sont les plus souvent référencées ensemble dans une même session utilisateur. Par exemple, une règle peut être obtenue dans le format suivant :

$$A.html, B.html \Rightarrow C.html$$

Cette règle montre que si l'utilisateur observe les pages A et B, il est fort probable qu'il observera la page C lors de la même session de recherche. Un algorithme courant pour extraire les règles d'association est l'algorithme *Apriori* (Kumar and

Rukmani, 2010). Prenant encore une fois l'exemple de la commerce électronique, l'extraction des règles d'association peut être utilisée pour trouver les pages pertinentes parcourues par les clients. Les règles extraites peuvent fournir par exemple les informations suivantes : quel est l'ensemble des pages fréquemment consultées ensemble par les utilisateurs ? Quelle page sera recherchée prochainement ? Quels sont les chemins fréquemment consultés par les utilisateurs ? De ce fait, l'application des règles d'association sur le parcours d'un acheteur en ligne permet généralement de connaître ses habitudes de consommation pour certains produits connexes.

La classification

Cette technique consiste à assigner une classe à chaque donnée parmi un ensemble de classes prédéfinies. Dans le domaine du web, elle développe des modèles de profil utilisateur appartenant à une classe ou une catégorie particulière. Cela nécessite l'extraction et la sélection de caractéristiques qui décrivent le mieux les propriétés d'une classe ou d'une catégorie donnée. La classification peut être effectuée en utilisant des algorithmes d'apprentissage supervisé tels que les arbres de décision, les classifieurs bayésiens naïfs (*Naif Bayes*)(Santra and Jayasudha, 2012), les machines à vecteurs de support (*Support Vector Machines*), etc (Patil and Patil, 2012; Talakokkula, 2015). Santra and Jayasudha (2012), par exemple, ont utilisé les classifieurs bayésiens naïfs pour classer les utilisateurs intéressés et non intéressés à partir des fichiers logs nettoyés (cf. figure 1.2). La figure 1.2 illustre un arbre de décision formé en utilisant l'algorithme de classification *Naif Bayes* qui permet de déterminer si un utilisateur qui se connecte au système est un « utilisateur intéressé » ou un « utilisateur non intéressé ».

Le regroupement (*Clustering*)

Cette technique permet de regrouper un ensemble d'éléments ayant des caractéristiques similaires. Dans le domaine de l'utilisation du web, il existe deux types de *clusters* intéressants à découvrir : les *clusters* des utilisations et les *clusters* de pages web (Srivastava et al., 2000). Le *clustering* des utilisations tend à établir des groupes d'utilisateurs ayant des habitudes de navigation similaires. Ces connaissances sont particulièrement utiles pour déduire les caractéristiques démographiques des utilisateurs afin d'effectuer une segmentation du marché dans les applications de commerce électronique ou de fournir un contenu web personnalisé

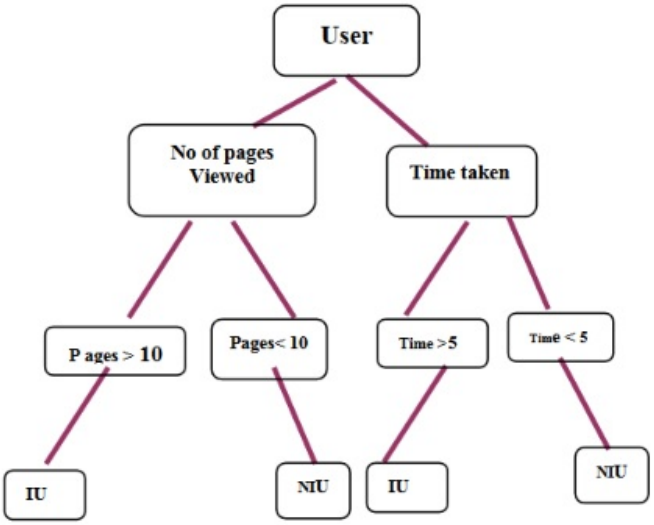


FIGURE 1.2 – Un arbre de décision généré pour les utilisateurs intéressés et les utilisateurs non intéressés

aux utilisateurs. Le regroupement de pages, par contre, permet de découvrir des groupes de pages web ayant un contenu apparenté. Ces informations sont utiles pour les moteurs de recherche et les fournisseurs d'assistance web (Talakokkula, 2015; Patil and Patil, 2012).

Dans l'optique de rechercher des groupes similaires d'utilisateurs du portail web de la Bibliothèque Nationale de France [2], Nouvellet et al. (2017) ont proposé également un algorithme de *clustering* fondé sur un modèle de mélange fini de modèles de *Markov* d'ordre 1 (cf. figure 1.3). Ce modèle considère que chaque session est générée par un modèle tiré au hasard avec des probabilités a priori parmi K modèles de chaînes de *Markov*. L'algorithme proposé estime, dans un premier temps, les paramètres d'un modèle probabiliste à partir d'un ensemble de données observées, et dans un deuxième temps, utilise le modèle pour catégoriser l'ensemble des sessions de recherche considérées en les associant à un comportement prototype. L'emploi des chaînes de *Markov* permettait de prendre en compte la séquentialité des événements ou activités réalisées par les utilisateurs dans une session. Les auteurs ont aussi utilisé l'algorithme Espérance-Maximisation pour apprendre les mélanges de modèles de *Markov* à partir des données de sessions. La motivation sous-jacente à cette analyse est de découvrir par exemple les activités

2. `https://gallica.bnf.fr`

susceptibles de déclencher le téléchargement d'un document dans la bibliothèque numérique.

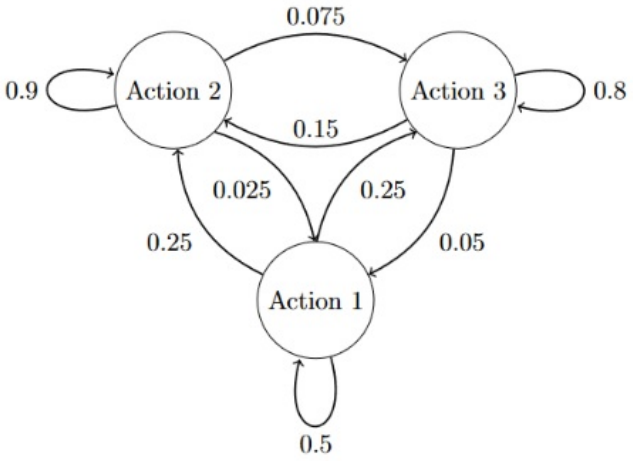


FIGURE 1.3 – Schéma d'une chaîne de Marcov de trois activités (Nouvellet et al., 2017)

L'analyse statistique

L'analyse statistique consiste à extraire des connaissances sur les utilisateurs. En analysant les fichiers logs, différents types d'analyses statistiques descriptives peuvent être effectuées (fréquence, moyenne, médiane, etc.) sur des variables telles que les pages web visitées, le temps de consultation et la longueur du chemin de navigation. De nombreux outils d'analyse statistique du trafic web produisent un rapport périodique contenant les informations que nous venons de citer. Ces rapports peuvent inclure une analyse limitée des erreurs de bas niveau, comme la détection des points d'entrée non autorisés ou la recherche des *URLs* invalides sollicités. Malgré le manque de profondeur de ces analyses, ce type de connaissance peut être potentiellement utile pour améliorer les performances des systèmes, renforcer leurs sécurités, faciliter la tâche de modification des sites et fournir un soutien aux décisions de marketing (Srivastava et al., 2000).

Nous venons de présenter le processus suivi pour effectuer du *Web Usage Mining*, le pré-traitement des données et la découverte des motifs. Nous avons ainsi présenté plusieurs méthodes pour la découvertes des motifs. Nous allons maintenant aborder la dernière étape du *Web Usage Mining*, l'analyse des motifs décou-

verts.

1.4.1.3 Analyse des motifs découverts (*Pattern Analysis*)

La motivation derrière l'analyse de motifs découverts est de filtrer les motifs non intéressants parmi ceux trouvés dans la phase de découverte de motifs. Les techniques de visualisation, telles que la génération de graphes ou l'attribution de couleurs à des valeurs différentes, peuvent souvent mettre en évidence les tendances générales dans les données. Les informations sur le contenu et la structure peuvent être utilisées pour filtrer les modèles contenant des pages d'un certain type d'utilisation, d'un certain type de contenu, ou des pages qui correspondent à une certaine structure d'hyperliens [3]. Parmi les outils de visualisation, nous pouvons citer l'outil *webCANAVAS* de Cadez et al. (2000), conçu pour la visualisation des classes des utilisateurs d'un site d'actualité (*msnbc.com*).

Enfin, le *Web Usage Mining* peut être donc utilisé à différentes fins. En fonction des modèles de comportements découverts, la navigation au sein des sites web peut être améliorée et le contenu peut être adapté pour chaque utilisateur ainsi que les recommandations automatiques qui correspondent le mieux aux profils des utilisateurs. Les habitudes de visite de l'utilisateur peuvent également être prédites.

Dans cette partie nous avons décrit la première catégorie pour l'analyse et la modélisation des comportements des utilisateurs (le *Web Usage Mining*). La partie suivante sera dédiée à la deuxième catégorie, l'analyse du web (connu par le *Web Analytics* en anglais).

1.4.2 Analyse du web (*Web Analytics*)

L'analyse du web, regroupe la mesure, la collecte et l'analyse des parcours de navigation des utilisateurs dans le but de comprendre et d'optimiser leurs expériences web. Cette catégorie peut être définie comme un ensemble d'analyses permettant d'étudier d'une manière précise les habitudes des utilisateurs sur les sites web ou sur les applications mobiles. Le *Web Analytics* est aujourd'hui l'un des éléments clés du marketing digital et de l'optimisation des sites, e-commerce notamment. Habituellement, cette catégorie d'analyse se base sur les données de

3. `https://www.ionos.fr/digitalguide/sites-internet/developpement-web/liens-hypertextes-ou-hyperliens-definition-et-exemples/`

trafic recueillies par les solutions de mesure et d'analyse d'audience afin optimiser un système précis (Kumar et al., 2012).

Généralement, le *Web Analytics* permet de fournir aux propriétaires des sites web des informations sur le comportement des utilisateurs afin d'améliorer la navigation sur leurs sites (Kumar et al., 2012; Zheng and Peltsverger, 2015). Cependant, les améliorations ne sont possibles qu'avec une bonne compréhension de la structure du site et les besoins des utilisateurs. L'objectif du *Web Analytics* est de fournir la bonne direction aux utilisateurs en ligne. Cela peut être fait en étudiant, par exemple, la manière d'utilisation du site web, comment l'utilisateur navigue de page en page et sur quelle page l'utilisateur reste le plus longtemps.

FIGURE 1.4 – Le processus du *Web Analytics* (Kumar et al., 2012)

Dans la plupart des cas, le processus du *Web Analytics* comprend une étude complète qui débute par l'identification des objectifs du concepteur jusqu'à l'amélioration du système étudié (cf. figure 1.4). Dans l'ensemble du processus, plusieurs étapes sont impliquées comme la définition des objectifs, la détermination des indicateurs clés de performance (KPIs), la collecte et l'analyse des données. Nous décrivons ces étapes dans les sous-sections suivantes.

1.4.2.1 Définition des objectifs du concepteur

Le *Web Analytics* vise à garantir le succès d'un système d'information à long terme. Cependant, les entreprises présentent également des priorités à court terme. Ces priorités peuvent être liées à l'acquisition de nouveaux clients, la fidélisation des utilisateurs, l'augmentation du nombre de visiteurs ainsi que l'amélioration de l'ergonomie du site (Waisberg and Kaushik, 2009). Pour chaque objectif, le choix pertinent des indicateurs clés de performance est primordial pour cette catégorie d'analyse (Kumar et al., 2012; Zheng and Peltsverger, 2015).

1.4.2.2 Indicateurs clés de performance (KPIs)

Habituellement, les KPIs sont créés afin d'évaluer le succès du site dans l'atteinte de ses objectifs. Chaque indicateur proposé pour un site web doit être associé

à une action. Parmi les KPIs les plus connus dans le *Web Analytics*, nous pouvons citer :

- Les types d'acquisitions : il est toujours pertinent de savoir d'où proviennent les utilisateurs des sites web et de savoir quel type de trafic est le plus efficace pour mener à un site web. La fonctionnalité Social », par exemple, du *Google Analytic*, le concepteur peut voir les réseaux sociaux qui amènent le plus de visiteurs.

- Les pages de destination : les pages sources qui amènent plus de trafic ? Pour analyser le comportement des utilisateurs, il est intéressant de connaître les pages par lesquelles ils entrent sur le site web. Cela permet de savoir si c'est cohérent par rapport à l'actualité, à la campagnes marketing de l'entreprise concernée, etc.

- Les pages de sortie : contrairement à l'indicateur précédent, les pages de sortie perquisitionnent les dernières pages consultées par les visiteurs qui les ont probablement poussé à quitter le site web ? Dans un site d'achat en ligne, par exemple, l'utilisateur doit arriver à une page de remerciement ou de validation de commande. Le cas échéant, le concepteur doit se poser la question de l'efficacité du parcours d'achat.

- Les flux de comportement : cet indicateur se focalise sur le parcours utilisateurs. Il aide à visualiser les sessions et d'observer les cheminements fréquents des utilisateurs. Un exemple de flux de comportement dans un site web généré par *Google Analytics* est présenté dans la figure 1.5.

- La fidélité des visiteurs : pour savoir si les contenus sont de qualité et si la communication est efficace, il est intéressant de connaître le taux de visiteurs fidèles (utilisateur qui a déjà visité le site au moins une fois). La réapparition d'un utilisateur démontre la pertinence des opérations de rétention : newsletter, stratégie social media, etc.

- La durée d'une session : cet indicateur aide à savoir si les utilisateurs passent du temps sur le site ou pas. En effet, il semble logique de constater que certains utilisateurs restent peu de temps sur le site. Parfois, pour des sites de commerces électroniques, ils cherchent simplement une petite information (numéro de téléphone, prix d'un produit, adresse, etc.) qu'ils vont vite trouver et se déconnecter. Cela peut aussi s'expliquer par le fait que le site

est positionné sur des requêtes très générales et qu'il attire des visiteurs peu ciblés. Ces derniers se rendent compte rapidement que le site n'est pas celui qu'ils recherchent.

1.4.2.3 Collecte et analyse des données

Une étape du processus importante de *Web Analytics* est la collecte et l'analyse des données. Avant d'effectuer les analyses souhaitées, la collecte et le sauvegarde des données doivent être effectuées selon un modèle prédéfini (Hu and Cercone, 2004; Zheng and Peltsverger, 2015) dans les fichiers logs. Les fichiers logs seront par la suite triés avec précision et sauvegardés dans une base de données externe ou locale pour une analyse ultérieure. L'analyse des données sera par la suite basée sur les indicateurs clés de performance (KPIs) qui ont été définis pour juger et comprendre le comportement de l'utilisateur.

Au final, comme le décrit la figure 1.4, les concepteurs du site web étudié, peuvent donc le modifier en fonction des comportements et besoins des utilisateurs et en se basant sur les indicateurs calculés et affichés dans les tableaux de bord.

Parmi les fameux systèmes du *Web Analytics*, on trouve, par exemple, *Google Analytic*[4]. Ce dernier est un service proposé par *Google* pour générer des statistiques détaillées sur le trafic et les sources de trafic d'un site web enregistrées dans des fichiers logs. comme nous l'avons indiqué précédemment, les données collectées par *Google Analytics* sont utilisées pour découvrir quelles sont les pages les plus les plus consultées par les utilisateurs sur le site web d'une entreprise, le type d'informations auquel les utilisateurs souhaitent accéder, les chemins qu'ils empruntent lors de leurs navigations, le temps qu'ils passent sur chaque page, etc. La figure 1.5 montre un exemple de flux de comportement généré par *Google Analytics*.

D'autres outils plus récents du *Web Analytics* permettent également de suivre les campagnes de marketing et les objectifs de conversion, tels que l'inscription des utilisateurs à la newsletter du site ou l'achat d'un produit (Poggi et al., 2013; Saura et al., 2017). Dans le contexte des bibliothèques numériques, Suire et al. (2016), par exemple, ont défini une méthode qui permet à partir des parcours de navigations des utilisateurs, d'aboutir à des indicateurs qui permettent de discriminer les différentes pratiques de recherche d'information.

En conclusion, le *Web Analytics* peut être considéré comme une sous catégorie du *Web Usage Mining*. Par contre, le *Web Analytics* a spécialement émergé dans le monde des entreprises qui étudient la rentabilité de leurs systèmes. En effet, de

4. `https://analytics.google.com`

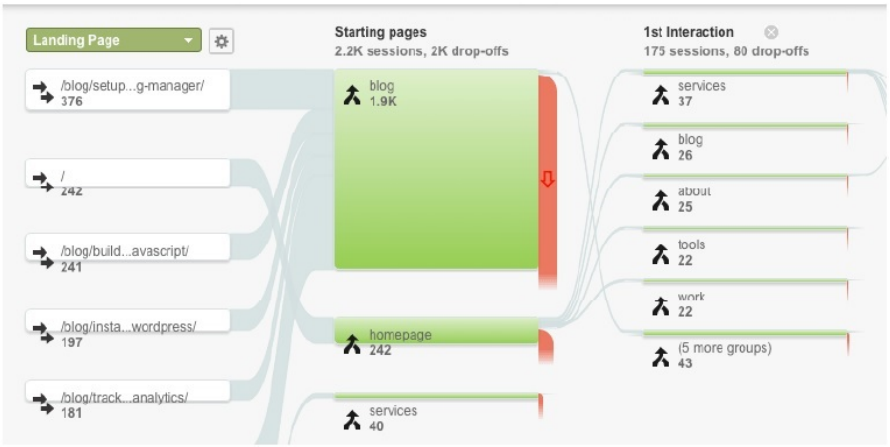


FIGURE 1.5 – Un exemple de flux de comportement généré par *Google Analytics*

nombreuses entreprises s'appuient sur des outils du *Web Analytics* pour obtenir des informations utiles sur ses utilisateurs en se basant sur les parcours de navigation. Cependant, ces programmes fournissent une vue d'ensemble trop simpliste du parcours utilisateurs et ne conduisent donc pas à une bonne compréhension du comportement de l'utilisateur (Poggi et al., 2013; Saura et al., 2017).

Dans cette section nous avons décrit la deuxième catégorie pour l'analyse et la modélisation des comportements des utilisateurs (le *Web Analytics*). La section suivante est dédiée à la dernière catégorie que nous présentons dans ce chapitre, la fouille des processus.

1.4.3 Fouille des processus (*Process Mining*)

La fouille de processus [5] (*Process Mining*) est une discipline de recherche relativement jeune qui fait le lien entre l'analyse de données et la modélisation des processus métiers. L'idée derrière la fouille de processus est de **découvrir les processus** (*Process Discovery*, section 3.5), **vérifier la conformité ou la qualité des modèles découverts** (*Conformance Checking*, section 3.6) et **améliorer les processus réels** (*Enhancement*) par l'extraction de connaissances à partir des parcours de navigation disponibles dans les systèmes d'information(Van der Aalst, 2016) (cf. figure 1.6). Un algorithme idéal pour la découverte de processus commence généralement par l'analyse les données de navigation dans les fichiers logs, identifie toutes les instances de processus (session de navigation) et tente ensuite

5. La fouille de processus a été développée par l'Université de Technologie d'Eindhoven

de définir les relations entre les activités menées par les utilisateurs.

En outre, il y a deux principaux moteurs pour l'intérêt croissant de la fouille de processus. D'une part, de plus en plus de données de navigation enregistrées fournissent, des informations plus détaillées sur l'histoire du processus. D'autre part, les systèmes d'information ont besoin de passer à une gestion d'organisation par les processus dans le but d'optimiser le fonctionnement de leurs rouages internes.

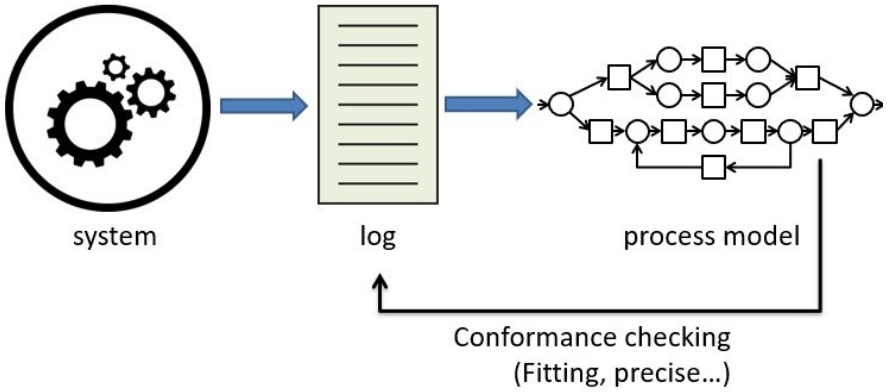


FIGURE 1.6 – La découverte des modèles de processus et la vérification de la conformité des modèles

La fouille de processus n'est apparue que récemment, mais elle a déjà été appliquée dans de nombreux domaines comme, l'analyse des processus de traitement dans les hôpitaux (Mans et al., 2008; Lu et al., 2019) et l'amélioration du processus d'audit (Jans et al., 2013; Van der Aalst, 2016). Le principal avantage des techniques de fouille de processus réside dans leur capacité à traiter le parcours de navigation dans son ensemble (un processus complet ayant une activité de début et une activité de fin). De plus, cette technique est réalisée au niveau processus contrairement aux *Web Analytic* et *Web Usage Mining* qui sont plus abstraits et fonctionnent au niveau page. Dans nos jours, avec la dominance des sites web dynamiques, l'analyse des parcours utilisateurs au sein d'un système d'information au niveau des processus est beaucoup plus utile que l'analyse au niveau des pages. En effet, la fouille de processus fournit une connaissance sur l'enchaînement des activités réalisées en produisant des processus métiers et cherche à évaluer la qualité du processus, mais ne cherche pas à uniquement identifier ni caractériser les différentes pages dans les parcours des utilisateurs (Bernard and Andritsos, 2017).

Dans cette section, nous avons donné un aperçu global de la fouille des processus. Le chapitre 3 présente plus en détail les méthodes et outils de la fouille de

processus.

1.5 Bilan et solutions

	Web Usage Mining	Web Analytics	Process Mining
Type de données	Fichiers logs	Fichiers logs marquage des pages	Fichiers logs
Méthodes	Règles d'association, clustering, classification, etc.	Calcul des indicateurs (KPIs)	Algorithmes de découverte de processus
Visualisation des résultats	Arbres de décision, chaînes de Marcov, etc.	Tableaux de bord	Réseaux de Petri, Graphes de successions directes, etc.
Schéma de navigation	Non	Non	Oui, mais pas avec des variations

TABLE 1.1 – Comparaison des catégories de modélisation et d'analyse des comportements des utilisateurs suite à leurs navigations dans un système d'information

Dans ce chapitre nous avons mené une étude sur les catégories d'analyse de modélisation des comportements des utilisateurs des systèmes d'information. Nous avons étudié le *Web Usage Mining*, le *Web Analytics* et le *Process Mining*. Une analyse plus poussée a été effectuée sur les deux premières catégories. La synthèse de cette étude est donnée par la table 1.1. Sur la première ligne nous présentons les données qui sont exploitées pour l'extraction de la connaissance. Sur la deuxième ligne nous donnons les méthodes et outils mis en œuvre par chaque catégorie. La troisième ligne décrit les éléments en sortie de ces analyses. Enfin, la quatrième ligne indique si les catégories sont en mesure de fournir un modèle abstrait de la navigation des utilisateurs (et de leurs comportements).

Les deux premières catégories le *Web Usage Mining* et le *Web Analytics* fournissent un modèle statistique de l'utilisation du site web, mais ne permettent pas d'extraire les schémas de navigation des utilisateurs du système d'information. Leurs objectifs est de construire des indicateurs statistiques qui décrivent la navigation mais ces catégories ne proposent pas d'indication sur les relations entre les différentes activités d'une séquence de navigation donnée.

Pour chaque page visitée par les utilisateurs, les outils du *Web Analytics* et du *Web Usage Mining*, identifient uniquement les pages précédentes et suivantes du parcours, présentant une vue abstraite de la relation entre les pages, les points de

sortie, les classes d'utilisateurs, etc. et n'extraient pas le comportement de l'utilisateur à un niveau d'abstraction approprié pour comprendre les chemins critiques réels empruntés par les utilisateurs. Néanmoins ces chemins sont utiles pour avoir une idée générale sur les parcours de navigation sur le site web, en montrant les principaux chemins suivis par les utilisateurs (Poggi et al., 2013; Følstad and Kvale, 2018).

La fouille de processus fournit ainsi des modèles de comportements des utilisateurs et permet d'extraire un modèle de la séquence d'activités réalisées. Elles sont en mesure de fournir des modèles de processus de bout en bout qui peuvent montrer et expliquer les choix et les décisions des utilisateurs. Ils permettent directement d'analyser les processus sous différents angles (par exemple : contraintes de temps, goulots d'étranglement ou relations entre les activités menées par les utilisateurs) (Van der Aalst, 2016). En revanche, les techniques de la fouille de processus n'ont pas la capacité de déterminer quelles sont les caractéristiques des cas (utilisateurs), qui décident de faire un choix particulier dans leurs parcours.

Dans cette thèse, nous ne nous concentrons pas sur la prédiction du prochain clic de l'utilisateur, mais nous cherchons à extraire les chemins critiques les plus pertinents qui se produisent dans le système d'information et à construire des modèles de navigation. En particulier, nous sommes intéressés par les activités réalisées par les utilisateurs et les flux de navigation pour la recherche d'information. La fouille de processus semble particulièrement adaptée à ce problème, car, comme nous l'avons souligné, elle fonctionne avec des données de navigation, un format séquentiel idéal pour représenter les parcours des utilisateurs. De plus, le travail avec des modèles de processus réels qui est au cœur de la fouille de processus, est capable de capturer la causalité et les chemins des interactions des utilisateurs qui mènent des tâches de recherche d'information.

1.6 Conclusion

Dans ce chapitre, nous avons abordé les notions de base liées au comportements des utilisateurs dans les systèmes d'information. Nous avons montré l'impact du comportement des utilisateurs sur leurs parcours de navigation. Nous avons présenté, ensuite, une étude des différentes méthodes existantes pour l'analyse et la modélisation des comportements des utilisateurs dans un système d'information. Le bilan présenté dans la section 1.4 montre que les techniques de la fouille de processus s'avèrent intéressantes. En effet, comme nous avons mentionné, les méthodes

liés aux *Web Usage Mining* ainsi que du *Web Analytics* présentent de nombreux avantages mais elles ne sont pas en mesure d'exploiter profondément les parcours de navigation et de produire des schémas de navigations complets qui décrivent les processus de navigation de bout en bout.

Dans le cadre de cette thèse, nous proposons donc d'utiliser les techniques de fouille des processus. Notre objectif consiste à étudier le parcours utilisateur dans des systèmes d'information pour lesquels les utilisateurs ne sont pas des employés mais des usagés du site, comme pour un site web marchand, ou l'externalisation d'un service d'une collectivité territoriale.

Nous avons décidé de nous concentrer sur les bibliothèques numériques. Ces dernières constituent des systèmes d'information particuliers dans lesquels des services et données sont proposées à un usager et elles produisent généralement des parcours de navigation complexes et difficile à traiter (plusieurs types d'utilisateurs et de tâches de recherche d'information). Notre première problématique est donc de tester la capacité de la fouille des processus à découvrir des modèles pertinents et qui couvrent les différents parcours de navigation des utilisateurs des bibliothèques numériques.

Toutefois, il est important pour notre démarche de cerner les caractéristiques de la recherche d'information dans une bibliothèque numérique ainsi que les types des utilisateurs identifiés. L'objectif du chapitre suivant sera donc d'étudier la particularité des parcours utilisateurs au niveau de leurs recherches d'informations dans une bibliothèque numérique.

Résumé exécutif

- Nous avons présenté une étude des différentes méthodes existantes pour l'analyse et la modélisation des parcours utilisateurs dans un système d'information.
- Dans le cadre de cette thèse, nous avons écarté les deux premières catégories, le *Web Usage Mining* et *Web Analytics*, car elles ne produisent pas des schémas de navigation complets.
- Nous avons retenu la fouille de processus comme méthode pour l'extraction des parcours utilisateurs.

Chapitre 2

Système d'information : cas des bibliothèques numériques

Sommaire

2.1 Introduction

Comme nous l'avons précisé dans le chapitre 1, notre objectif consiste à étudier le parcours utilisateur dans des systèmes d'information pour lesquels les utilisateurs ne sont pas des employés mais des usagers à l'instar d'un site web marchand ou d'un service d'une collectivité territoriale externalisé. L'usager construit sur le site son propre processus métier qui, contrairement aux processus classiques rencontrés, n'est pas entièrement maîtrisé par les applications du système d'information. Par conséquent, les zones où l'usager est laissé en autonomie mènent à produire des processus dits « non-structurés », parfois non complets et qui ont des faibles variations.

Dans le cadre de cette thèse, nous nous concentrons sur les systèmes d'information proposées par les bibliothèques numériques pour plusieurs raisons : premièrement, l'utilisation des bibliothèque numériques devient de plus en plus répandue dans plusieurs domaines et pour des divers propos (Smeaton and Callan, 2005; Paskali et al., 2021), ce qui permet d'avoir des objectifs différents et ainsi des comportements plus variés. Deuxièmement, elles sont utilisées à l'échelle mondiale ce qui nous permet d'obtenir des traces non homogènes liées à différentes communautés. Dans notre jeu de données (cf. Chapitre 4), par exemple, nous examinons les comportement des utilisateurs de plusieurs nationalités à l'instar du Japon, la France, les États-Unis, etc. Troisièmement, les bibliothèques numériques constituent des systèmes d'information qui offrent beaucoup de services, de données et de fonctionnalités aux usagers. Quatrièmement, les bibliothèques numériques sont surtout considérées parmi les systèmes d'information compatibles avec notre problématique sur l'analyse des processus faiblement ou non structurés et dont leur modèle n'est pas (complètement) défini. Les utilisateurs appliquent généralement différents chemins d'exécution afin d'atteindre leurs objectifs et chacun d'eux construit son propre parcours. En particulier, nous nous concentrerons sur le parcours utilisateurs lors de la recherche d'information ce qui constitue une problématique largement abordées suivant des aspects informatique mais également sur les aspects d'analyse de l'usage.

Le présent chapitre est consacré à la compréhension et la présentation des caractéristiques des parcours utilisateurs lors de la recherche d'informations dans une bibliothèque numérique. Ce chapitre est organisé en quatre sections principales. La section 2.2 propose différentes définitions des bibliothèques numériques. Dans la section 2.3, nous décrivons les processus utilisateurs ou parcours de navigation dans les bibliothèques numériques. Les parcours utilisateurs, fortement liés

à l'activité et aux tâches de recherche d'information de l'utilisateur, sont quant à eux décrits dans la section 2.4. Enfin, la section 2.5 présente des études qui décrivent les différences entre la conception centrée-concepteur et la conception centrée-utilisateur pour les bibliothèques numériques.

2.2 Bibliothèques numériques

Une bibliothèque numérique est définie comme une collection de services et d'objets d'information disponibles sous forme numérique. Les objets d'information peuvent être définis comme tout objet qui peut se présenter sous un format numérique, comme les livres, les articles de journaux, les sons, etc. Les bibliothèques numériques organisent et présentent les objets d'information aux utilisateurs et les aident à traiter ces objets (Kani-Zabihi et al., 2006).

D'autre définitions des bibliothèques numériques existent comme celle de Witten et al. (2009) : une bibliothèque numérique est une collection organisée et ciblée d'objets numériques, notamment de textes, d'images, de vidéos et de sons, ainsi que des méthodes d'accès et de recherche, de sélection, de création, d'organisation, de maintenance et de partage des collections. Principalement utilisées à des fins d'éducation ou de recherche, les bibliothèques numériques permettent de répondre aux besoins d'une population d'utilisateurs. La figure 2.1 résume les définitions que nous avons présentées : une bibliothèque numérique regroupe des documents numériques ainsi que des services associées pour un ou des utilisateurs cibles.

FIGURE 2.1 – Définition d'une bibliothèque numérique

Les bibliothèques numériques peuvent être utilisées pour de nombreuses raisons, néanmoins l'ensemble le plus important des cas d'utilisation se concentre sur l'accès à l'information. Trouver certains types de contenu, récupérer des informations spécifiques, localiser des éléments connus, l'accès à des documents que l'utilisateur ne connaît pas suffisamment, constituent des motivations, parmi de nombreuses autres, qui sont basées sur le contenu et plus ou moins orientées vers un but précis qui conduiront un utilisateur vers le terminal d'accès d'une collection numérique

d'informations. Il n'est donc pas nécessaire d'être spécialiste pour consulter et découvrir les documents mis à disposition des utilisateurs.

Une autre caractéristique des bibliothèques numériques est la multiplicité des sources. Ainsi, en plus de stocker certaines collections de documents, la plupart des bibliothèques numériques contiennent des collections distantes. C'est-à-dire qu'un utilisateur, après une recherche, se verra proposer des documents provenant non seulement de la bibliothèque numérique en question, mais aussi d'autres services (une autre bibliothèque numérique, des archives publiques, etc.). En effet, les bibliothèques numériques ne stockent pas toujours l'ensemble des ressources qu'elles mettent à la disposition des utilisateurs. En revanche, elles proposent aux utilisateurs de rechercher de l'information grâce aux différentes méta-données [1] collectées sur les plate-formes des différents partenaires.

Les bibliothèques numériques sont également connues comme un produit d'information numérique visant à satisfaire les besoins des utilisateurs en informations, par le biais de collections de documents auxquelles sont fournis des services à valeur ajoutée. Le portail *Gallica*[2], par exemple, qui est maintenue par la Bibliothèque Nationale de France, propose plusieurs services et fonctionnalités pour ses utilisateurs. Les services et fonctionnalités proposés concernent la consultation de documents (soit par type de documents, thème, situation géographique, etc.), modes de recherche dans les collections à l'aide du moteur de recherche (recherche simple, recherche avancée, etc.) ou aussi selon les différents modes de visualisation des documents (visualiseur simple ou mosaïque, visualiseur zoom adapté pour la presse, les images et les cartes, etc). La figure 2.2 illustre la manière de consultation des documents numériques selon la thématique « Geographie » dans *Gallica*.

En revanche, l'utilisation des bibliothèques numériques a montré que leurs conceptions ont une incidence sur la manière dont les utilisateurs interagissent avec elles (Kani-Zabihi et al., 2006). Afin d'assurer le bon fonctionnement des bibliothèques numériques et les services qu'elles fournissent, il est recommandé d'inclure les besoins des utilisateurs dans les processus de conception et de création. La modélisation des comportements et des attentes des utilisateurs en fonction de leurs parcours de navigation est un des moyens d'atteindre ces objectifs (Xie, 2008).

1. Le titre, le contenu, l'origine ou la forme d'un document
2. `https://gallica.bnf.fr/accueil/fr`

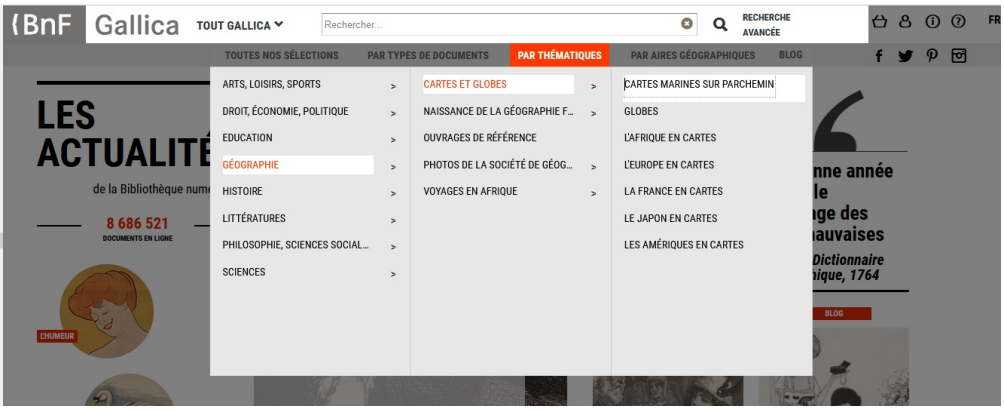


FIGURE 2.2 – Consultation des documents numériques selon la thématique dans la bibliothèque numérique *Gallica*

2.3 Parcours utilisateurs dans une bibliothèque numérique

La définition des bibliothèques numériques transmet implicitement ses avantages. En effet, l'accès à la bibliothèque peut être réalisé de n'importe où et à tout moment tant qu'il y a une connexion internet disponible et un dispositif d'accès approprié (Blandford, 2006). De ce fait, les utilisateurs peuvent naviguer aisément à travers de grandes masses de documents, souvent très hétérogènes. Le parcours de recherche d'information qu'ils mènent est un processus qui repose principalement sur les requêtes soumises au moteur de recherche, les filtres appliqués aux résultats et les documents consultés (Cole et al., 2015). Chacun de ces éléments dépend par ailleurs du type de tâche à l'origine de la recherche d'information, du niveau de connaissance de l'utilisateur pour le domaine de sa recherche et la manière dont sont indexées les données.

Comme nous avons mentionné dans l'introduction de ce chapitre, les bibliothèques numériques sont aussi considérées parmi les systèmes d'information qui ne sont pas structurés autour de processus et dont le modèle des processus n'est pas (complètement) défini. Les utilisateurs appliquent généralement différents chemins d'exécution afin d'atteindre leurs objectifs. Il existe, des utilisateurs qui peuvent finir rapidement leurs recherches avec succès et d'autres qui font des longs parcours de navigation et quittent le site web de la bibliothèque numérique sans atteindre leurs objectifs. Il existe aussi, des utilisateurs qui atteignent leurs objectifs sans utiliser les fonctionnalités existantes alors que les concepteurs prévoyaient que le

parcours des utilisateurs soient toujours à la base des leurs modèles fournis. Cette variabilité d'utilisateurs permet de produire beaucoup de types de parcours utilisateurs. Ces parcours sont aussi considérés comme des processus caractérisés par leur nature non structurée. Souvent, les séquences d'activités des utilisateurs sont désordonnées et génèrent un grand nombre de modèles de processus différents et non redondants.

Les séquences d'activités produisent des traces d'exécution des différents utilisateurs et comme nous l'avons indiqué dans le chapitre précèdent (section 1.3), elles sont enregistrées dans des fichiers logs. Ces traces témoignent de l'expérience des utilisateurs dans cet environnement numérique et constituent des sources de connaissances intéressantes qui peuvent être exploitées à des fins variées en accord avec l'utilisateur. Une fois collectées, les traces sont des conteneurs de connaissances riches en informations contextuelles et utilisables à la fois à des fins d'analyse pour inférer des connaissances pertinentes sur l'activité menée par les utilisateurs et également à des fins d'assistance à l'utilisateur.

Les parcours utilisateurs dans les bibliothèques numériques sont fortement liés à la catégorie de ses utilisateurs ainsi que les types de leurs tâches de recherche d'information (Kani-Zabihi et al., 2006). L'objectif de la section suivante est de présenter les différents usages et pratiques des utilisateurs dans les bibliothèques numériques.

2.4 Usages et pratiques

Les avantages que présentent les bibliothèques numériques par rapport aux bibliothèques traditionnelles conduisent à une utilisation massive des bibliothèques numériques. Avec l'accélération de la transformation numérique, il semble naturel voir s'élargir le spectre des utilisateurs des bibliothèques numériques. Les différents types d'utilisateurs génèrent donc des besoins d'information et des pratiques de recherche différentes.

Nous présentons dans les sous-sections suivantes les différentes catégories des utilisateurs dans les bibliothèques numériques ainsi que leurs intentions et leurs méthodologies de recherche d'information.

2.4.1 Catégorisation des utilisateurs

Les bibliothèques numériques organisent et gèrent de grandes collections d'œuvres ou d'informations numériques. Étant donné que ces dernières sont développées pour fournir des informations multimédias avec une architecture d'information riche et un contenu étendu, les bibliothèques numériques peuvent introduire des défis supplémentaires pour différents types d'utilisateurs (Xie et al., 2020).

Depuis la naissance des bibliothèques numériques, il y a eu toujours des tentatives pour développer et améliorer l'utilisation des collections en ligne. L'amélioration se base généralement sur la catégorisation de leurs utilisateurs ou bien évidement sur leurs comportements de recherche ou leurs types de navigations. La diversité des utilisateurs du patrimoine culturel numérique a conduit à une stratégie qui simplifie les possibilités virtuellement illimitées des profils d'utilisateurs en créant des groupes génériques. Ces groupes sont parfois catégorisés en des utilisateurs novices ou experts (Johnson, 2008), ou aussi des utilisateurs novices, intermédiaires et avancés (Kani-Zabihi et al., 2006) mais le plus souvent, les groupes d'utilisateurs sont créés en fonction de leurs professions (par exemple, conservateur, bibliothécaire, chercheur, enseignant ou étudiant). D'autres groupes ont été créés en fonction de leurs intérêts ou leurs motivations (par exemple, touriste, explorateur, utilisateur général), ou leurs groupes d'âges (par exemple, adulte, enfant) (Cifter and Dong, 2009).

Les utilisateurs peuvent être aussi classés en des utilisateurs professionnels ou des utilisateurs non professionnels (Cifter and Dong, 2009). Les utilisateurs professionnels ont une bonne connaissance de la tâche qu'ils effectuent : ils sont formés et peuvent avoir une expérience préalable avec la bibliothèque numérique ou de la tâche concernée. Les utilisateurs non professionnels peuvent être des utilisateurs expérimentés ou des utilisateurs novices. Les utilisateurs expérimentés quand à eux, peuvent avoir une certaine expérience avec la bibliothèque numérique ou de la tâche concernée, mais leur connaissance de la tâche est beaucoup plus limitée que celle des utilisateurs professionnels. Les utilisateurs novices ne connaissent pas la tâche ou la bibliothèque numérique et ne disposent généralement pas d'informations suffisantes pour effectuer la tâche (cf. figure 2.3).

La connaissance de la tâche est donc l'un des principaux facteurs qui séparent les utilisateurs finaux en deux groupes : les utilisateurs professionnels et les utilisateurs non professionnels. La conception pour les utilisateurs professionnels est différente de la conception pour les utilisateurs non professionnels. Les utilisateurs professionnels se préoccupent davantage de la fiabilité, de la cohérence et de l'effi-

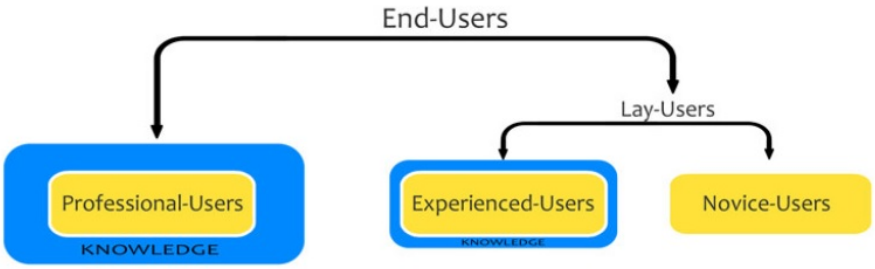


FIGURE 2.3 – Les types d'utilisateurs des bibliothèques numériques (Cifter and Dong, 2009)

cacité des bibliothèques numérique, tandis que les utilisateurs non professionnels s'intéressent davantage à la facilité d'utilisation, à l'esthétique et au plaisir. Lors de la conception pour les utilisateurs non professionnels, les concepteurs doivent rendre leurs produits plus intuitifs à utiliser, les instructions faciles à comprendre et sans jargon, et ils doivent tenir compte des différents contextes d'utilisation. En outre, les concepteurs doivent prendre en compte les parcours des utilisateurs novices, expérimentés ainsi que les professionnels, car cela les inspirera sur la manière d'aider les types d'utilisateurs dans leurs recherches d'informations.

Enfin, les utilisateurs ne sont pas tous les mêmes, leurs navigations dans les bibliothèques numériques est influencée par leurs préférences, leurs expériences et leurs connaissances. Par exemple, alors que certains utilisateurs aiment la simplicité des interfaces des bibliothèques numériques, d'autres préfèrent des interfaces plus attrayantes avec des illustrations ou des images. Bien que les concepteurs soient eux-mêmes des utilisateurs, comprendre les utilisateurs est un défi pour les concepteurs. Les concepteurs ont tendance à concevoir pour eux-mêmes, mais les utilisateurs pour lesquels ils conçoivent diffèrent souvent des concepteurs eux-mêmes. Selon Margolin (1997), les utilisateurs sont devenus un thème central du discours sur le design, mais les utilisateurs restent encore peu compris par les designers.

2.4.2 Recherche d'information

Les différents types des tâches de recherche adoptés par les utilisateurs durant une recherche d'information sont étudiés depuis plusieurs années. La recherche dans ce domaine s'est longtemps concentrée sur l'analyse des besoins informationnels des utilisateurs. Plusieurs modèles ont été développés pour tenter de décrire

les différentes étapes qui régissent le comportement d'une personne engagée dans une tâche de recherche d'information. Ces modèles détaillent plus ou moins les différents comportements et actions des utilisateurs.

Parmi les premiers modèles de recherche d'information, nous présentons le modèle de *Wilson* développé dans (Wilson, 1981). Ce modèle est conçu pour un besoin de rapprocher le monde des utilisateurs et des systèmes informatiques (cf. figure 2.4). C'est le premier modèle à s'intéresser aux raisons qui poussent les utilisateurs à agir d'une certaine manière lorsqu'ils sont face à un besoin informationnel. *Wilson* a étudié les raisons qui poussent un utilisateur à s'engager dans une recherche d'information et le rôle joué par son environnement (l'usage de technologies, sa vision du monde, etc.). Son modèle est fondé sur l'hypothèse que les besoins informationnels exprimés par les usagers des systèmes d'information étaient issus de besoins plus fondamentaux, d'ordre physiologique, cognitif ou affectif, liés au contexte personnel ou social de l'individu et à son environnement (politique, économique, technologique...). Il suggérait que les obstacles à surmonter pour accéder à l'information souhaitée provenaient de ce contexte ou de cet environnement.

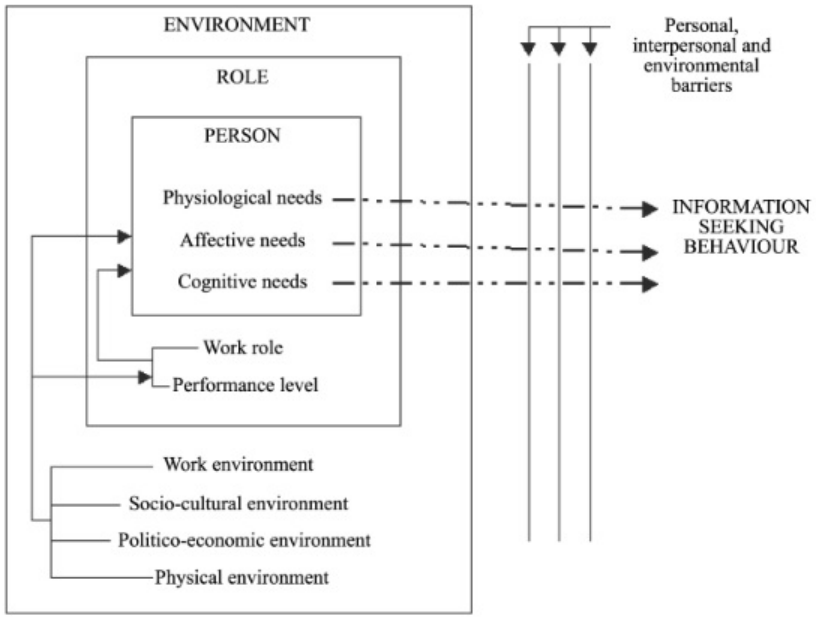


FIGURE 2.4 – Le modèle de Wilson (1981) pour la recherche d'information

Ellis and Haugan (1997) a proposé également un modèle qui définit un ensemble d'activités qui peuvent être entreprises pour réaliser une tâche de recherche d'information. Ainsi, une tâche de recherche commence par l'identification de sources.

Plusieurs stratégies sont alors envisageables. L'approche descendante consiste à sélectionner de larges domaines d'intérêt et de préciser la recherche au fur et à mesure. Au contraire, un utilisateur avec une approche ascendante effectuera d'abord une recherche précise avant de généraliser si les résultats obtenus ne lui conviennent pas. Les sources sont ensuite triées pour ne conserver que les plus pertinentes. Finalement, les informations nécessaires sont extraites des différentes sources sélectionnées.

Le modèle de *Marchionini* (Marchionini, 2006), est un modèle simple qui regroupent les tâches qu'un utilisateur souhaite réaliser lors d'une recherche d'information en deux catégories (cf. figure 2.5). Les tâches de recherche « simple » (*Lookup* en anglais) et les tâches exploratoires (*Exploratory* en anglais). Les premières correspondent à des besoins précis comme, par exemple, de trouver la date de naissance d'une personnalité, vérifier l'existence d'un document, etc. Les deuxièmes sont plus complexes et nécessitent souvent plus d'investissement de la part des utilisateurs (préparation d'une présentation orale, acquisition de connaissances dans un domaine particulier, etc.).

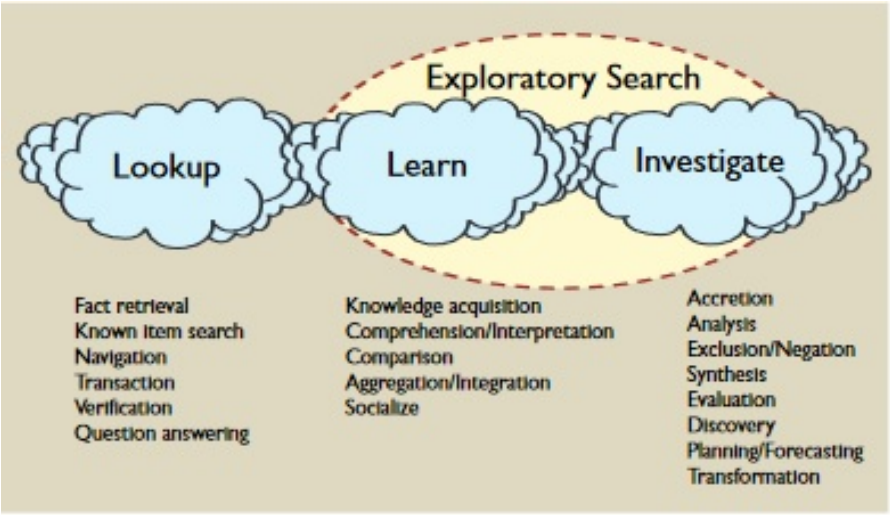


FIGURE 2.5 – Le modèle de Marchionini (2006) pour la recherche d'information

Suite à la réalisation d'une étude qualitative sur l'usage des utilisateurs du portail numérique de la Bibliothèque Nationale de France, Beaudouin et al. (2016) se sont inspirés du modèle de *Marchionini* pour définir les types de recherches d'information. Deux grands types d'usages ou de recherche dans une bibliothèque numérique ont été défini : un usage pragmatique/utilitaire/ponctuel recouvrant des pratiques de recherche d'informations sur un lieu, un événement ou de recherche

d'illustrations sur un sujet précis. La consultation est ciblée sur un objet restreint. Un usage exploratoire recouvrant des pratiques au long cours de consultation et d'exploration concernant un domaine. Cette approche s'apparente davantage à la constitution d'un corpus, à l'exploration systématique d'un territoire et de ses frontières, en s'appuyant sur une lecture des documents ou encore à la recherche méthodique d'un document original.

Les modèles présentés dans cette partie décrivent les différents comportements que peuvent adopter des utilisateurs durant une tâche de recherche d'information. Ces modèles peuvent nous aider à comprendre certains aspects du comportement des utilisateurs en observant les actions qu'ils réalisent sur l'interface graphique. La section suivante sera dédiée à la description de la problématique suivante : l'écart des bibliothèques numériques centrées sur les concepteurs avec les attendus des utilisateurs.

2.5 Conception centrée-concepteur Vs. conception centrée-utilisateur

Comme nous l'avons indiqué dans notre introduction générale, lors de la mise en œuvre d'une bibliothèque numérique, les utilisateurs ne sont souvent pas impliqués dans le cycle de conception. En effet, il parait que les bibliothèques numériques sont généralement construites en réponse aux besoins d'une communauté particulière et impliquent rarement des personnes ayant une expérience préalable du domaine (Witten et al., 2001). Les bibliothèques numériques sont ensuite évaluées en entreprenant des expériences qui font généralement appel à un petit groupe d'utilisateurs.

Certains concepteurs et décideurs dans les bibliothèques numériques semblent plus réservés sur leur mission concernant les interactions ou les parcours de navigations réelles des utilisateurs qui sont réalisées. Dans le même cadre, plusieurs études affirment que les concepteurs d'un système d'information n'ont pas fondé leurs conceptions sur des usages réels (Dinet, 2009). Toujours dans le cas des bibliothèques numériques, *Wilson Lizabeth* définit l'approche des concepteurs comme « *si nous le construisons, ils viendront* » (Wilson, 2003). Le concepteur affirme ici qu'il ne considère pas les attentes des utilisateurs dans leur conception des modèles théoriques d'utilisation des systèmes d'information et que c'est à eux de s'adapter à leurs modèles. Cela conduit presque à un grand écart entre le modèle de

comportements centré sur le concepteur et le modèle centré sur l'utilisateur des bibliothèques numériques.

Généralement, l'information dans les bibliothèques numériques est limitée aux utilisateurs disposant de compétences sophistiquées en matière d'information (Blandford, 2004). Une autre problématique aussi qui se pose, est que la réussite des bibliothèques numériques est généralement liée au nombres de connexions des utilisateurs, alors que les milliers de connexions n'impliquent pas forcement la satisfaction des utilisateurs. L'inadéquation des bibliothèques numériques aux attentes des utilisateurs peut se mesurer, par exemple, par la non utilisation des forums (générique et thématique) intégrés au portail web ou le peu d'utilisation des fonctionnalités avancées telles que la recherche avancée ou l'extraction des textes à partir des images (Papy and Jakubowicz, 2021).

Plusieurs études ont été réalisées dans le but de récolter les suggestions des utilisateurs pour les bibliothèques numériques ainsi que leurs opinions sur le classement d'une gamme de fonctionnalités suggérées. Dans le cadre d'une étude effectuée par Kani-Zabihi et al. (2008), il est estimé que 76% des utilisateurs ne parviennent pas à effectuer une simple tâche de recherche (*Lookup search*) d'une référence dans les bibliothèques numériques. En outre, plus de 80% des utilisateurs qui ont participé à l'étude n'ont pas l'intention d'utiliser le manuel d'utilisation disponible. Les utilisateurs considèrent que le manuel proposé n'est pas assez intuitif et plutôt destiné aux experts. Les utilisateurs considèrent que le vocabulaire utilisé dans le manuel est trop technique et ainsi complexe pour une bonne partie d'usagers. Toutes ces considérations font que pour les concepteurs et les commanditaires des bibliothèques numériques, la performance des utilisateurs apparaît souvent « décevante » (Hansen and Järvelin, 2005).

Dans une étude connexe, Kani-Zabihi et al. (2006) ont révélé que les attentes des utilisateurs en matière de fonctionnalité des bibliothèques numériques sont les mêmes quels que soient les antécédents des utilisateurs en matière de technologie de l'information. Cependant, leurs exigences concernant des caractéristiques spécifiques sont dissemblables et dépendent considérablement des expériences antérieures des utilisateurs.

Par conséquent, les concepteurs ont adopté le choix d'analyser les traces d'interactions générées par les utilisateurs pour faire face à l'écart entre les modèles des bibliothèques numériques centrés sur les concepteurs et les attendus des utilisateurs. Les traces de navigations collectées ont été fréquemment étudiées [Huurnink et al. (2010); Meij et al. (2011); Bogaard et al. (2019, 2018); Walsh et al. (2019); Liang and Leng (2020)]. Dans certains cas, les études se concentrent sur la dé-

tection et l'analyse des intérêts (thématiques) des utilisateurs, par exemple, pour catégoriser les sujets de recherche suite à l'analyse des requêtes [Huurnink et al. (2010); Meij et al. (2011); Liang and Leng (2020)], ou pour identifier des modèles d'utilisation dans différentes parties de la collection [Bogaard et al. (2019, 2018)]. Ces études se concentrent sur une analyse statistique des journaux de recherche. Leur objectif principal est de mieux comprendre les besoins et les intérêts des utilisateurs. Contrairement à ces travaux, nous proposons, dans cette thèse, de découvrir des modèles de processus basées sur les traces de navigations à l'aide des techniques de la fouille des processus. Au mieux de nos connaissances, la modélisation des parcours utilisateurs à l'aide des algorithmes de la fouille des processus dans les bibliothèques numériques est une première.

2.6 Conclusion

Ce chapitre établit donc un socle de connaissances nécessaire à la compréhension des bibliothèques numériques. Celles-ci sont considérées parmi les systèmes d'information qui produisent des processus d'utilisation complexes étant donné les différents types d'utilisateurs ainsi que leurs comportements de recherche d'information. Les parcours de recherche produits sont ainsi très variés. Dans cette thèse, notre choix pour les bibliothèques numériques s'est appuyé sur plusieurs raisons dont la principale est la nature des parcours (traces) générées par les utilisateurs lors de leurs navigations. Ces traces sont généralement à la fois incomplètes et non structurées et qui constituent pour nous un défi particulier.

Ce chapitre nous aura permis en premier lieu de comprendre la particularité des parcours utilisateurs lors de la recherche d'information dans une bibliothèque numérique. Nous avons également expliqué pourquoi les catégories des utilisateurs ainsi que leurs types de tâches de recherche d'information ont un impact sur leurs parcours de navigations. Nous avons aussi présenté une problématique non négligeable dans les bibliothèques numériques, qui est la nécessité de s'orienter vers une conception centrée-utilisateurs pour les bibliothèques numériques.

Suite à plusieurs études qui signalent la problématique de l'écart entre ce que proposent théoriquement les concepteurs et les besoins et cheminements réels des utilisateurs, les concepteurs des bibliothèques numériques se concentrent de plus en plus sur les besoins et les navigations réels de leurs utilisateurs. Leur objectif est de comprendre comment les utilisateurs travaillent et utilisent l'information, et comment ces informations sont utilisées. Ces caractéristiques doivent être prises en

compte ensuite dès la conception. La modélisation des traces d'exécution réelles des utilisateurs semble être la meilleure solution pour concevoir un modèle d'exécution de processus efficace. Ces modèles permettront aux concepteurs des bibliothèques numériques de réaliser un gain de temps et de moyens. Les techniques d'analyse et de modélisation adoptées par les chercheurs et les concepteurs fournissent une analyse superficielle des traces et ne permettent pas d'obtenir des modèles de processus complets qui expliquent les choix et les décisions des utilisateurs.

Comme nous l'avons indiqué dans le chapitre précédent, nous proposons donc dans cette thèse, de modéliser les comportements réels des usagers des bibliothèques numériques à l'aide de la fouille des processus. La fouille de processus a montré sa capacité de découvrir et modéliser des modèles de comportements qui reflètent la réalité des cheminements dans plusieurs domaines (systèmes d'information des hôpitaux par exemple) (Van der Aalst, 2016). La problématique qui se pose maintenant est sur la capacité des algorithmes de découverte de processus de produire une telle qualité de modèles en se basant sur les parcours des utilisateurs des bibliothèques numériques.

Résumé exécutif

- Nous avons décrit les principes des bibliothèques numériques ainsi que les typologies d'utilisateurs.
- Nous avons identifié les particularités des parcours utilisateurs lors de la recherche d'information dans une bibliothèque numérique.

Chapitre 3

Fouille de processus pour la découverte des modèles de comportement des utilisateurs des bibliothèques numériques

Sommaire

3.1 Introduction

Dans le chapitre 1, nous avons précisé ce que nous souhaitons analyser et modéliser au niveau du comportement de l'utilisateur. Nous avons identifié que notre objectif concernait l'étude et la modélisation du parcours utilisateur dans le but d'améliorer la conception des systèmes d'information ainsi que de satisfaire les besoins des utilisateurs. Nous avons également comparé différentes méthodes pour analyser le parcours utilisateur et nous avons retenu les approches fouille de processus développées initialement par Van der Aalst (2016) dans le cadre de la modélisation de processus métiers. Dans le chapitre 2, nous avons présenté la particularité et les caractéristiques des parcours utilisateurs dans les bibliothèques numériques.

La question de recherche que nous abordons dans ce chapitre consiste à déterminer si la fouille de processus peut permettre d'extraire de la connaissance sur les parcours utilisateurs d'une bibliothèque numérique. Pour cela nous allons mener une étude sur des traces de navigation provenant d'une expérimentation réalisée dans une bibliothèque numérique pour laquelle nous disposons déjà d'une analyse de l'usage réalisée par Suire et al. (2016) en utilisant des indicateurs statistiques. Ces traces sont d'une taille maîtrisable et ont été générées afin d'analyser les comportements des utilisateurs à l'aide des techniques de l'analyse du web (cf. section 1.4.2).

L'objectif de ce chapitre est de prouver et tester la capacité des techniques existantes de la fouille des processus dans la génération des modèles de processus précis qui décrivent d'une manière efficace les parcours des utilisateurs des bibliothèques numériques. Pour atteindre cet objectif, nous avons commencé par l'étude et la comparaison des différents algorithmes de la fouille des processus existants dans la littérature, ainsi que l'évaluation des modèles de processus découverts (cf. figure 1.6). Nous proposons dans ce chapitre, une comparaison des résultats de ces algorithmes en terme de performances vis-à-vis des données de test d'une part et de leur capacité à permettre à un regard expert (Suire et al., 2016) de comprendre et analyser les processus détectés d'autre part.

Le présent chapitre est organisé en deux parties. La première partie introduit la fouille de processus. Elle présente l'organisation et le format des données avec un objectif de fouille des processus dans les sections 3.2 et 3.3, les modèles utilisés en sortie de la fouille de processus (cf. section 3.4), les grande familles d'algorithmes de découverte en section 3.5 et les mesures définies pour évaluer la qualité des modèles découverts (section 3.7). La deuxième partie du chapitre a comme objectif,

la découverte et l'évaluation des modèles de comportement des utilisateurs d'une bibliothèque numérique. Nous décrivons donc, en premier lieu l'expérimentation sur laquelle nous basons notre comparaison des algorithmes de découverte. Puis la partie se poursuit par une présentation des résultats de la découverte et l'évaluation des modèles selon leur capacité à extraire des informations pertinentes sur le parcours des utilisateurs de notre expérimentation (cf. section 3.8).

3.2 Organisation des données

Le système d'information représente l'ensemble de ressources et d'outils permettant d'organiser et de donner accès à l'information d'une organisation. Il est vu comme un système socio-technique incluant à la fois la structure organisationnelle, les processus métier ainsi que les données afférentes (Piccoli, 2007). La structure organisationnelle représente les règles de répartition de l'autorité, des tâches, de contrôle et de coordination. Elle permet notamment de définir les différents acteurs du système d'information, leurs rôles et leurs droits. Les processus métier représentent les activités ou les enchaînements d'activités relatives aux affaires de l'organisation concernée. Ainsi, un processus $\mathcal{P}$ est caractérisé par un ensemble ordonné d'activités A.

Pour mener des analyses sur les données avec un objectif de fouille de processus, il est tout d'abord nécessaire de collecter les informations adéquates. Les fichiers logs constituent une bonne source de données et peuvent être considérés comme des traces brutes. Pour pouvoir les analyser, il est nécessaire de les transformer en un ensemble de traces modélisées (un journal d'événements) portant les informations nécessaires, permettant notamment d'identifier les données représentant les activités. Une fois le modèle de traces défini, il faut ensuite sélectionner l'algorithme de fouille de processus le plus adéquat à ces traces pour obtenir le modèle de processus.

Dans cette section, nous présentons la terminologie liée à l'organisation des données avec un objectif de fouille de processus qui permet de faciliter la lecture de ce document. Un journal d'événements issu des fichiers logs fait référence à un ensemble d'événements (activités ou actions réalisées par les utilisateurs) consécutifs enregistrés lors de l'exécution d'un processus. Ces événements sont regroupés par instances de processus (Van der Aalst, 2016). Dans les bibliothèques numériques, par exemple, chaque utilisateur pourrait être une instance qui suit un processus de recherche. La séquence des événements liés à une instance particulière est ap-

CaseId	User	Timestamp	Activity	Task category
1	$User_1$	2016-01-12 10 :34 :25	request	Lookup
2	$User_2$	2016-01-12 10 :36 :25	request	Lookup
1	$User_1$	2016-01-12 10 :34 :26	scroll	Lookup
1	$User_1$	2016-01-12 10 :34 :28	ressourceAccess	Lookup
3	$User_3$	2016-01-12 10 :36 :26	request	Exploratory
3	$User_3$	2016-01-12 10 :36 :27	scroll	Exploratory

TABLE 3.1 – Un fragment d'un journal d'événements : chaque ligne correspond à un événement

pelée une trace. Chaque ligne du fichier logs enregistre des informations liées à un événement exécuté telles que son identifiant (*caseId*), l'identifiant de l'utilisateur (*user*), l'événement lui-même, l'estampille temporelle *timestamp* (c'est-à-dire le jour, l'heure, la minute et la seconde) et quelques attributs supplémentaires. Le tableau 3.1 montre un extrait d'un journal d'événements enregistré suite à une expérimentation (section 3.7) effectuée dans une bibliothèque numérique (Suire, 2018).

Définition 3.2.1 (Un journal d'événements)**.**

Un journal d'événements $L = \{t_1, t_2, ..., t_k\}$ est un ensemble fini de traces et k est un nombre fini de traces.

Définition 3.2.2 (Une trace)**.**

Chaque trace t_i $(1 \leq i \leq k)$ contient un ensemble fini d'événements $t_i =< e_{i1}, e_{i2}, ...e_{in_i} >$ et n_i est un nombre fini d'événements.

Définition 3.2.3 (Un événement)**.**

- Chaque événement de la trace $e_{i1} = (c_j; a_w; s)$ doit faire référence à une activité a_w de l'ensemble d'activités $A = \{a_1; a_2; ...a_{|A|}\}$, à un *timestamp* s, et un identifient de trace *caseId* c_j, qui identifie de manière unique l'utilisateur qui effectue l'action (cf. figure 3.1).
- Chaque événement e du journal d'événements est caractérisé par sa fréquence f_e qui correspond au nombre de fois que e apparaît dans toutes les traces.

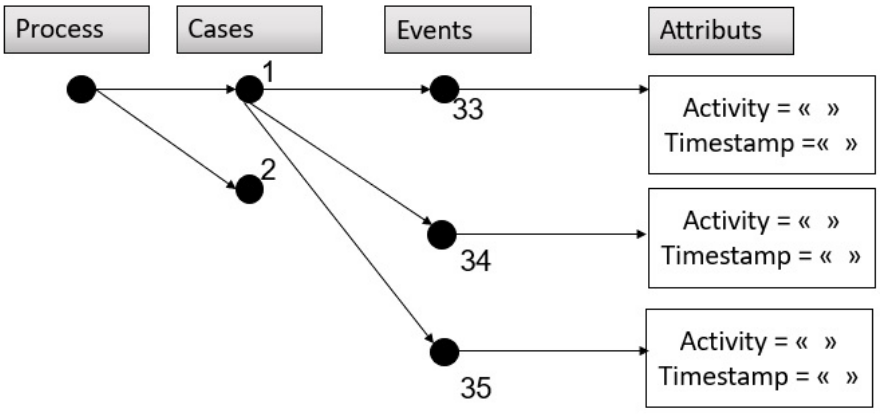


FIGURE 3.1 – Organisations des données dans un journal d'événements

Définition 3.2.4 (la relation entre les événements)**.**

Deux événements e_i et e_j sont en **relation** r dans une trace t lorsque e_j $(2 \leq j \leq n)$ est exécuté directement après e_i $(1 \leq i \leq n-1)$, $r_{i,j}$ est appelé la succession directe entre e_i et e_j. Sa fréquence $f_{r_{i,j}}$ est le nombre de fois où e_i apparaît directement après e_j dans toutes les traces.

Plus formellement, les traces d'événements se présentent comme un ensemble d'éléments où chaque élément représente une séquence d'événements (aussi appelée « cas »). Par exemple, dans l'ensemble de traces d'événement $L = [\langle a, b, c\rangle^2, \langle b, a\rangle^3]$, $\langle a, b, c\rangle$ et $\langle b, a\rangle$ sont des représentations d'exécutions sous forme de **séquence d'événements** effectuant successivement les **activités** a, b et c pour la première trace et b puis a pour la seconde trace. Dans cette représentation, l'exposant décrit le nombre d'exécutions d'une séquence dans l'ensemble des traces. Ainsi, il y a dans l'ensemble des traces L deux occurrences de la séquence d'événements $\langle a, b, c\rangle$ et trois occurrences de $\langle b, a\rangle$.

3.3 Format des données

Les traces d'événements suivant le modèle d'événement (cf. figure 3.1) peuvent être représentées par le format XES (*Extensible Event Stream*), un format XML [1]

1. Les fichiers XML (Extensible Markup Language) sont des documents texte qui utilisent des balises personnalisées pour décrire et structurer des données.

utilisé comme entrée par tous les algorithmes populaires de la fouille des processus (Van der Aalst, 2016). Le format XES a été commercialement standardisé par IEEE (Günther and Verbeek, 2016) en raison de sa simplicité, extensibilité et expressivité. Suivant la définition de Van der Aalst (2016), un document XES contient un journal d'événements composé d'un nombre quelconque de traces. Chaque trace décrit une liste séquentielle d'événements correspondant à un cas particulier. Le journal, ses traces et ses événements peuvent avoir un nombre quelconque d'attributs.

3.4 Modélisation des processus

Les algorithmes de fouille de processus n'utilisent pas tous le même modèle pour la modélisation des processus. Il est donc nécessaire de présenter quelques modèles avant de présenter les algorithmes utilisés dans la fouille de processus. Nous présentons dans ce qui suit les formalismes les plus répandus permettant de représenter et de manipuler les processus dans le cadre de la fouille de processus. Nous commençons par définir le modèle général des réseaux de *Petri*, puis nous décrivons les restrictions *Workflow Net* et *Process Tree* qui permettent plus particulièrement la modélisation des processus métier. Nous présentons également les graphes des successions directes (connu par les *Directly Follows Graphs*). Dans le cadre de cette thèse, nous avons fait le choix de nous baser que sur les réseaux de *Petri* et les *Directly Follows Graphs* pour la modélisation des processus.

3.4.1 Réseaux de *Petri* (*Petri Nets*)

Les réseaux de *Petri* sont des modèles mathématiques avec une représentation graphique qui permet d'exprimer des séquences d'événements avec des synchronisations et des partages de ressources (Murata, 1989). Graphiquement, les réseaux de *Petri* sont des graphes orientés composés de deux types de nœuds : des places et des transitions. Ces nœuds sont reliés alternativement par des arcs. Si un arc relie une place à une transition, il est dit « entrant » ; s'il relie une transition à une place, il est dit « sortant ». Des jetons, ou marques, sont associés aux places. Ils décrivent l'état courant du système (cf. figure 3.2). Leurs évolutions dépendent d'un ensemble de règles que décrivent les transitions. Formellement, les réseaux de *Petri* sont définis comme suit.

Définition 3.4.1 (Réseau de *Petri* marqué)**.** Un réseau de *Petri* marqué est donné par le couple : $N =< R, s >$ où :

- R est un réseau de *Petri* défini par le quadruple $< P, T, Pre, Post >$ avec :
  - P un ensemble fini de places.
  - T un ensemble fini de transitions (représentent les activités existantes dans le journal d'événements).
  - $P \cap T = \emptyset$
  - $\mathcal{F} \subseteq (P \times T) \cup (T \times P)$ est l'ensemble des arcs reliant les places et les transitions. Pre est la matrice d'incidence avant, elle décrit l'ensemble des arcs entrant des transitions et $Post$ est la matrice d'incidence arrière, elle définit l'ensemble des arcs sortant des transitions.
- s est le marquage, c'est une application qui associe à chaque place du réseau un ensemble de jetons. Ainsi, dans chaque place, il peut y avoir un certain nombre de jetons, pouvant être nul. On note s_0 le marquage initial du réseau de *Petri*.

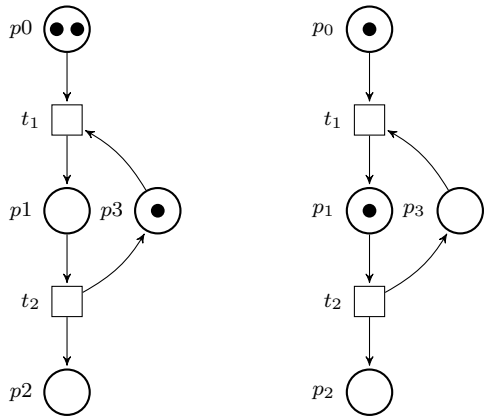


FIGURE 3.2 – Évolution du marquage d'un réseau de *Petri*

Les jetons « se déplacent » dans le réseau de *Petri* en respectant des règles d'évolution. Une transition est franchissable si chacune de ses places d'entrée contient

au moins un jeton ($\forall p \in P,\ s(p) \geq Pre(p,t)$), dans le cas le plus simple des réseaux pour lesquels aucun poids n'est attaché aux arcs.

Le franchissement (ou tir) d'une transition correspond à retirer un jeton de chaque place amont à la transition, et à ajouter un jeton à chaque place aval (place de sortie). Par exemple, la figure 3.2 illustre un exemple où le franchissement de la transition t_1 consomme un jeton dans les places p_0 et p_3 pour en produire un dans la place p_1 puis le franchissement de la transition t_2 consommera un jeton de p_1 pour en produire un dans p_2 et un dans p_3. Le franchissement d'une transition correspond à l'occurrence d'un événement. Une séquence de franchissement de transitions décrit l'exécution d'un processus.

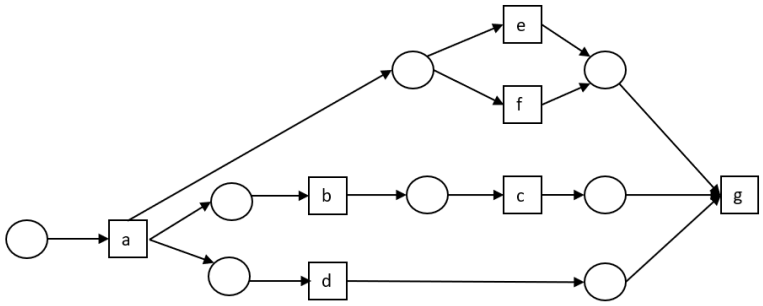


FIGURE 3.3 – Un exemple de réseau de *Petri*

Un réseau de *Petri* contient également un ensemble défini de relations concernant l'ordre d'exécution des événements. Le réseau de *Petri* de la figure 3.3 indique que l'activité a commence le processus et g la termine. Parmi les relations entre les activités , on peut trouver la relation parallèle (*AND template*) pour le cas des activités b et d. Cela signifie que nous pouvons exécuter les activités (b et d) dans n'importe quel ordre. Nous pouvons également trouver des relations de causalité (*Sequence template*) comme le cas de b et c. Les relations causales signifient que l'activité c est exécuté seulement après l'exécution de b. De plus, nous avons les relations *OR* entre deux activités qui indiquent que seul l'un d'entre eux est autorisé à être exécuté (le cas des activités e et f).

3.4.2 *Workflow Nets*

Une sous-classe importante des réseaux de *Petri* est les *Workflow nets*, dont les caractéristiques les plus importantes sont d'avoir un « début » et une « fin » dédiés. Les *Workflow nets* introduits par Van der Aalst (Van Der Aalst, Van Hee,

Ter Hofstede, Sidorova, Verbeek, Voorhoeve and Wynn, 2011) sont les modèles les plus utilisés pour modéliser, analyser et vérifier les *workflows*. Ce sont des réseaux de *Petri* possédant une unique place marquée qui correspond au début du processus et une unique place de terminaison du processus.

Définition 3.4.2 (*Workflow Net*)**.** Un réseau de *Petri* R est un *Workflow Net* si et seulement si :

1. P contient une place p_i sans arcs entrants (le point de départ du processus) ;
2. P contient une place p_o sans arcs sortants (le point de fin du processus)
3. Connexité : on ajoute une transition particulière $\bar{t}$ reliant p_o à p_i et on obtient ainsi $\bar{R} = (P, T \cup \bar{t}, \mathcal{F} \cup \{(p_o, \bar{t}), (\bar{t}, p_i))\}$ fortement connecté.

Dans les travaux de l'Université de Technologie d'Eindhoven (Van der Aalst, 1996; Leemans et al., 2013) l'auteur définit la notion de « soundness ». Un *Workflow Net* est dit *sound* si :

- $\forall_S([i] \xrightarrow{*} s) \Rightarrow (s \xrightarrow{*} [o])$, pour chaque marquage accessible depuis la place initiale, il existe une séquence de transitions qui mène au marquage final avec $[i]$ le marquage ne contenant que la place initiale p_i et $[o]$ le marquage ne contenant que la place finale p_o ;
- $\forall M([i] \xrightarrow{*} s \wedge s \geq [o]) \Rightarrow (s = [o])$, l'état final est le seul accessible depuis l'état initial ;
- le réseau est vivant.

Cela implique que quel que soit le marquage, il est possible d'atteindre le marquage final $[p_o]$. Les *Worfklow Nets* peuvent être organisés en blocs, appelés *Block-Structured Workflow Nets*, qui constituent une représentation hiérarchisée ou chaque partie représente un *workflow* qui peut être divisé de la même manière récursivement (Leemans et al., 2013).

3.4.3 *Process Trees*

Le *Process Tree* est une représentation abstraite compacte d'un *block-structured workfow* net(Bose and Van der Aalst, 2009 a; Leemans et al., 2013). L'arbre représente le processus complet. Les feuilles décrivent les activités du processus et les nœuds donnent les relations entre les activités.

Nous définissons formellement les arbres de processus de manière récursive. Nous supposons qu'un alphabet fini d'activités A et un ensemble d'opérateurs x sont donnés. Le symbole $T \notin A$ désigne l'activité silencieuse.

Définition 3.4.3 (*Process Tree*)**.** Soit A un ensemble fini d'activités.

- a, avec $a \in A \cup T$, est un *Process tree* ;
- soient $M_1 ... M_n$ avec $n > 0$ des *Process trees* et x un opérateur de *Process tree* alors $x(M_1, ..., M_n)$ est un *Process tree* ; x pouvant être une composition séquentielle ($\rightarrow$), un choix exclusif ($\times$), une composition parallèle ($\wedge$) ou une boucle ($\circlearrowleft$).

3.4.4 *Directly Follows Graphs*

Les *Directly Follows graphs* sont des graphes où les nœuds représentent les activités existantes dans le journal d'événements et les arêtes dirigées sont présentes entre les nœuds s'il y a au moins une trace dans le journal où l'activité source est suivi par l'activité cible. En plus de ces arêtes dirigées, il est facile de représenter des mesures comme la fréquence (en comptant le nombre de fois que l'activité source est suivi par l'activité cible) et la performance (une certaine agrégation, par exemple la moyenne, du temps écoulé entre les deux activités).

Définition 3.4.4 (*Directly Follows Graphs*)**.** Étant donné A un ensemble fini d'activités tel que $a_i \notin A$ une activité de départ et $a_o \notin A$ une activité de fin,

- un *Directly Follows Graph* est un graphe dirigé (N, E), tel que $N : A \cup a_i, a_o$ est un ensemble de nœuds et $E : N \times N$ est un ensemble d'arêtes ;
- l'activité de début a_i n'a pas d'arêtes entrantes et l'activité de fin a_o n'a pas d'arêtes sortantes.

3.5 Algorithmes de découverte des processus

Nous venons d'introduire les outils pour comprendre les principes des algorithmes de fouille de processus. Nous allons maintenant présenter les algorithmes de découvertes les plus courant en expliquant leurs principes et principaux intérêts.

Les techniques de découverte de processus ont émergé au début des années 90 (Van der Aalst et al., 2004). Elles permettent d'extraire des modèles de processus à partir d'une liste de traces d'exécution. Les processus cibles étaient propres, complets et structurés (exemple dans la figure 3.4). Au fil des années, les recherches concernant la découverte des processus se sont focalisées sur des problématiques de plus en plus complexes comme la génération de modèles à partir des traces d'exécution bruitées, incomplètes et/ou non structurées.

D'une manière générale, les techniques de découverte de processus se basent sur l'identification des liens de causalité entre les activités dans un ensemble de traces (Van der Aalst, 2012). Soit un ensemble de traces $L = \{t_1, t_2, ..., t_n\}$ contenant des activités A. Par exemple, $L = [\langle a, b, c\rangle^2, \langle b, a\rangle^3]$ qui contient 5 exécutions réparties en 2 séquences avec les activités possibles a, b ou c.

Quatre relations sont définies entre deux activités quelconques a_1 et a_2 :

- Succession directe (*direct succession*), notée $a_1 > a_2$ indique que a_1 est immédiatement suivi de a_2 dans certaines séquences (nous avons dans l'exemple $a > b$ pour la première séquence et $b > a$ pour la seconde) ;

- Séquence (*causality*), notée $a_1 \rightarrow a_2$, si a_1 est situé avant a_2 et pas a_2 avant a_1 (par exemple nous avons $a \rightarrow c$) ;

- Parallèle, noté $a_1||a_2$, si on a à la fois $a_1 > a_2$ et $a_2 > a_1$ (par exemple nous avons $a||b$) ;

- Choix (*choice*), noté $a_1\#a_2$, s'il n'y a jamais ni $a_1 > a_2$ ni $a_2 > a_1$ (par exemple nous avons $a\#c$)

3.5.1 Algorithmes de bases

L'algorithme ***Alpha*** **(α)** est parmi les premiers algorithmes développés pour la découverte des modèles de processus. Il génère un *Workflow-Net* (partie 3.4.2) qui traduit les processus enregistrés dans les journaux d'événements (logs) (Van der Aalst et al., 2004). L'algorithme α consiste à identifier les relations observées entre les activités dans les logs. Il se fonde sur les relations suivantes : succession immédiate, causalité, parallélisme et choix. L'algorithme α est incapable de découvrir des boucles, ainsi que des activités dupliquées. Afin de remédier à ces problèmes, l'algorithme α a été étendu dans un premier temps en α^+ puis en α^{++} (Alves de

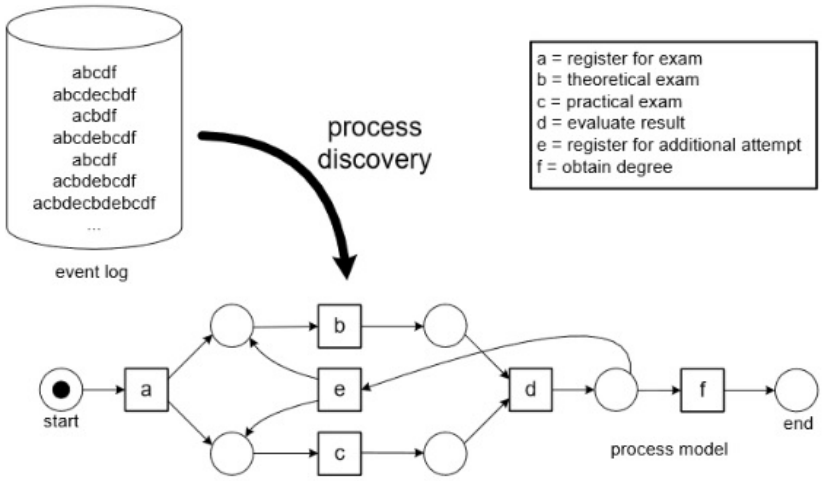


FIGURE 3.4 – Découverte d'un modèle de processus (Van der Aalst, 2016)

Medeiros et al., 2004) ainsi que d'autres versions (Van Dongen et al., 2009). Cependant, ces algorithmes présentent toujours quelques limites parmi lesquelles leur sensibilité au bruit dans les journaux d'événements. Dans ce chapitre nous considérons l'algorithme α^{++} uniquement.

3.5.2 *Heuristic Miner*

L'algorithme ***Heuristic Miner*** a été développé par Weijters and Ribeiro (2011). Il se base sur une approche heuristique afin de résoudre les problèmes rencontrés par l'algorithme α. L'idée générale de cet algorithme consiste à identifier l'ensemble des relations à partir des logs et de construire le modèle des processus en se basant sur les relations identifiées. La différence principale entre l'algorithme α et l'*Heuristic Miner* vient du fait que ce dernier se base sur des mesures statistiques afin de déterminer les relations entre les activités. L'algorithme est divisé en trois étapes. Il commence par produire un graphe de dépendance en comptant toutes les successions directes présentes dans les logs (les fréquences des successions sont ensuite stockées dans une matrice). Puis, il calcule le taux de dépendance entre chaque activité afin de ne conserver que celles qui ont une relation de causalité significative. Le graphe de causalité découvert (Van der Aalst, Adriansyah and Van Dongen, 2011) sera construit après avoir fixé un seuil sur le taux de dépendance et la fréquence des successions. La seconde étape consiste à identifier les divergences et les synchronisations en utilisant une approche heuristique. Enfin, le graphe

de causalité peut être transformé en un réseau de *Petri* si nécessaire. L'un des inconvénients de l'*Heuristic Miner* est qu'il produit des modèles qui contiennent des anomalies et ne sont pas capables de rejouer la majorité des logs (on parle alors de modèles non *sound*)

3.5.3 *Genetic Miner*

L'algorithme ***Genetic Miner*** commence par créer aléatoirement des modèles de processus à partir des logs (Van der Aalst et al., 2005). La métrique de Justesse (cf. partie 3.6.1) de chaque modèle de processus est ensuite calculée. Par la suite, l'ensemble des modèles de processus construits évoluent à l'aide d'opérateurs génétiques : le croisement qui permet de combiner deux modèles et la mutation qui permet de modifier un modèle de processus. L'objectif de l'algorithme génétique est d'améliorer d'une façon itérative les modèles de processus jusqu'à atteindre un ensemble de modèle qui satisfait un critère d'arrêt. Cet algorithme retourne un modèle qui peut être converti en un réseau de *Petri*. L'algorithme *Genetic Miner* aborde les problèmes tels que le bruit, les traces incomplètes, les choix non libres, la concurrence et les activités dupliquées (Van der Aalst, 2016). Parmi les limites de cet algorithme, nous pouvons citer la complexité de construction des modèles de processus à partir des jeux de données réels.

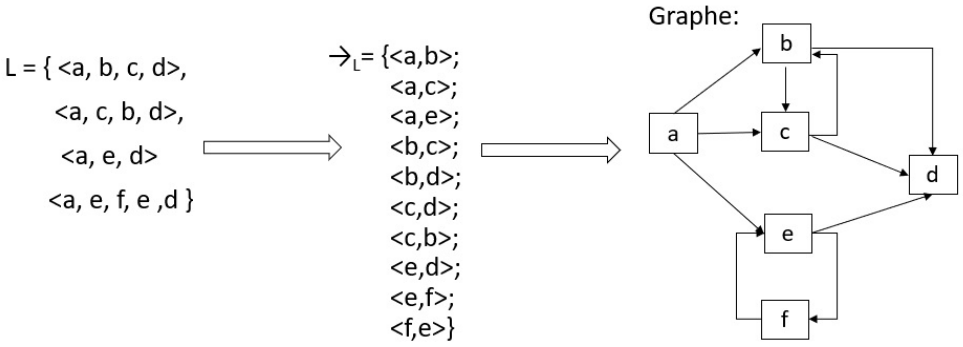


FIGURE 3.5 – Construction d'un *Directly Follows Graph*

3.5.4 *Inductive Miner*

L'algorithme ***Inductive Miner*** (Leemans et al., 2013) applique une approche basée sur le principe « diviser pour mieux régner ». Il procède en deux étapes :

il commence d'abord par la construction d'un *Directly Follows Graph* (Définition 3.4.4) à partir du fichier logs entier comme le montre la figure 3.5 et directement trouve le meilleur découpage à effectuer sur le graphe. Son principal objectif est de déterminer les opérateurs qui décrivent le mieux le découpage. L'opérateur de composition séquentielle ($\rightarrow$) dans la figure 3.6, par exemple, implique que les activités b, c, e, f se produisent à la suite de l'activité a, mais pas réciproquement. Ces deux étapes se répètent successivement sur les sous-ensembles des logs. Le modèle de processus obtenu est hiérarchiquement structuré sous la forme d'un *Process Tree* (partie 3.4.3). Le *Process Tree* miné peut être facilement transformé en une autre notation (idéalement en un réseau de *Petri*). En outre, l'*Inductive Miner* a été développé afin de produire des modèles *sound* et qui ne contiennent pas d'anomalies 3.4.2. Il traite efficacement le bruit dans les logs en supprimant les traces non fréquentes (dont la fréquence d'apparition est rare vis-à-vis de l'ensemble totale des traces) (Van der Aalst et al., 2004) ainsi que le problème des logs non complets. En contrepartie, il n'arrive pas à identifier des motifs complexes et non locaux de contrôle de processus.

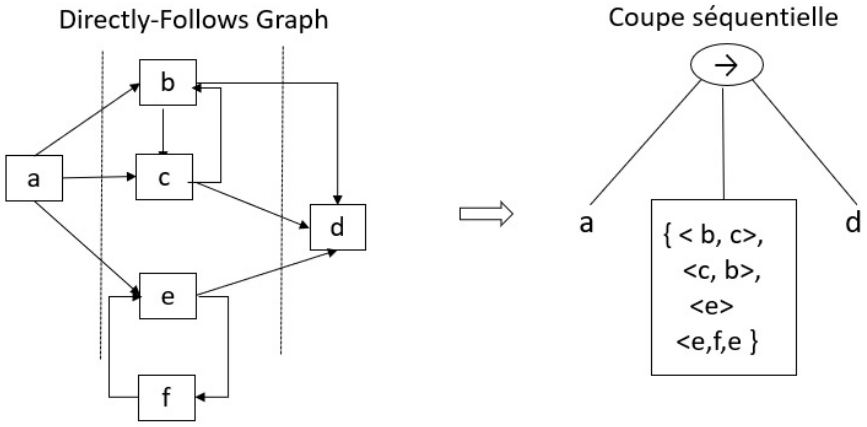


FIGURE 3.6 – Coupe séquentielle sur le *Directly Follows Graph*

3.5.5 *State Based Regions*

Le ***State Based Regions*** (Van der Aalst et al., 2010) est parmi les algorithmes fondés sur les régions (*Region Based algorithms*) qui génèrent des modèles sous la forme de réseaux de *Petri*. Leurs principaux buts est d'équilibrer la sur-précision (*overfitting*) et la sous-précision (*underfitting*) des modèles minés. Le ***State Based***

Regions est un algorithme qui génère un réseau de *Petri* à partir d'un système de transitions. Il offre la possibilité d'utiliser des abstractions afin de construire le système de transitions. Les techniques de découverte de processus qui utilisent les abstractions sont capables d'obtenir leur modèle à partir des traces partielles contrairement aux techniques sans abstractions qui doivent utiliser l'ensemble complet des traces afin d'engendrer leurs modèles. De nombreux types d'abstractions ont été proposés pour l'algorithme *State Based Regions* à l'instar des : *abstractions Set* (ni l'ordre ni la fréquence des activités dans les traces sont considérés), *Multi-Set* (seulement la fréquence des activités est prise en compte) et *Sequence* (l'ordre et la fréquence des activités sont considérés) et autre type d'abstraction dans lesquels chaque état du système de transitions peut être représenté par une trace complète ou partielle. Le système de transitions obtenu est ensuite transformé en un réseau de *Petri* en utilisant la théorie des régions. Cette dernière ne tient compte que des régions minimales non triviales comme les ensembles vides et les ensembles incluant tous les états du système de transitions. Pour chaque région, l'algorithme *State Based Regions* génère une place dans le réseau de *Petri* et pour chaque événement du système de transitions, il génère une transition.

3.5.6 *Language Based Regions*

Tout comme le *State Based Regions*, l'algorithme ***Language Based Regions*** appartient à la famille des algorithmes fondés sur les régions (**Region Based algorithms**). Le but de l'algorithme ***Language Based Regions*** est de trouver les places dans le réseau de *Petri* découvert. Il commence par traduire les logs en un *language process* (Van der Werf et al., 2008). L'idée principale de l'algorithme est de trouver des places en se basant sur le *language process*. Toutes les places candidates correspondent à une région du langage. Le réseau de *Petri* découvert sera construit en ajoutant une place pour chaque solution non négative trouvée suite à la résolution d'un programme linéaire. Chaque solution est représentée sous la forme d'un triplet (X, Y, c), où X est l'ensemble des arcs d'entrée d'une place, Y est l'ensemble des arcs de sortie et c est le nombre de jetons dans la place. Enfin, l'algorithme *Language Based Regions* utilise les propriétés issues des logs (relations de causalité) pour déterminer le modèle final en assurant de ne mettre que les places expressives (ayant plus d'arcs entrants et sortants). Les algorithmes basés sur les régions assurent la Justesse (sous-section 3.6) dans la découverte des modèles de processus, ainsi que l'identification des structures de contrôles complexes. En revanche, ils sont incapables de traiter les logs incomplets et bruités.

3.5.7 *Fuzzy Miner*

Le dernier algorithme que nous présentons dans ce chapitre est le ***Fuzzy Miner***. Ce dernier est un algorithme performant introduit par Günther (2009). Il est considéré comme une solution pour traiter le cas des processus métiers, issus des logs réels, en forme de spaghetti. La particularité de ces processus repose sur leur important niveau de flexibilité, qui réduit leur structuration. L'idée principale de cet algorithme consiste à construire une carte géographique simplifiée basée sur l'abstraction ou l'agrégation. Les activités et leurs relations (les arcs) peuvent être regroupés ou supprimés en fonction de leurs rôles dans le processus. L'algorithme *Fuzzy Miner* s'appuie sur deux concepts : la *signification* qui mesure l'importance relative de chaque activité, et la *corrélation* qui mesure la proximité de deux activités successives. L'importance de chaque activité peut être évaluée selon sa fréquence alors que la corrélation peut être définie en mesurant le temps d'occurrence entre deux activités. Les activités qui se produisent peu de temps l'une après l'autre sont fortement corrélées. À partir de ces deux concepts, il est possible de produire un modèle simplifié et cohérent en utilisant des heuristiques (Günther, 2009). Le *Fuzzy model* miné contient des *nœuds primitifs*, qui ne contiennent qu'une activité et des *nœuds cluster* qui contiennent des activités agrégées. L'algorithme *Fuzzy Miner* offre aussi la possibilité aux utilisateurs de *zoomer* ou d'agréger le modèle. Parmi les limitations de cet algorithme, nous pouvons citer le fait qu'il n'arrive pas à découvrir la concurrence entre les activités dans le logs (choix et parallélisme). Une autre limite est que le *Fuzzy* modèle découvert par l'algorithme *Fuzzy Miner* ne peut pas être comparé avec les autres modèles découverts vu qu'il possède ses propres mesures d'évaluation.

Le tableau 3.2 compare les différents algorithmes de découverte de processus selon des critères que nous jugeons utiles par apport à la modélisation des parcours utilisateurs dans les bibliothèques numériques. Les trois premières critères sont forcément liés à la qualité des logs traités. Nous donnons une attention particulière au algorithmes qui tiennent en compte les problèmes liées au logs réels vu qu'on s'intéresse au traces issues des bibliothèques numériques. Les deux dernières critères liés à la qualité des modèles découverts semblent également très importants pour notre étude : les algorithmes doivent produire des modèles qui tiennent compte des différents choix des utilisateurs ainsi que les modèles ne doivent pas contenir des anomalies (*soundness*) afin de pouvoir les évaluer.

	Qualité du log			Qualité du modèle	
	Bruit	Non complet	Log réel	*Soundness*	choix et parallélisme
Alpha ++	−	−	−	−	−
Heuristic Miner	+	−	+	−	−
Inductive Miner	+	+	+	+	+
Genetic Miner	+	+	+	+	−
Language Based Regions	−	−	−	+	−
State Based Regions	−	−	−	+	−
Fuzzy Miner	+	+	+	−	−

TABLE 3.2 – Analyse qualitative des algorithmes de découverte des processus (« + » pour soutenu et « - » pour non soutenu)

3.6 Évaluation de la qualité des modèles découverts

Afin d'évaluer la performance des algorithmes de découverte de processus, les modèles de processus découverts ont été transformés sous la forme d'un réseau de *Petri* pour que l'on puissent vérifier la conformité des modèles par rapport aux logs de départ. D'après Rozinat et al. (2007); Van der Aalst (2016), un bon modèle de processus découvert doit trouver un équilibre entre la Justesse, la Précision, la Généralisation et la Simplicité. La plupart du temps ces critères s'opposent, améliorer l'un affaibli l'autre. Par conséquent, les algorithmes de découverte de processus doivent trouver un équilibre (cf. figure 3.7) entre ces propriétés (Buijs et al., 2012; Adriansyah, 2014).

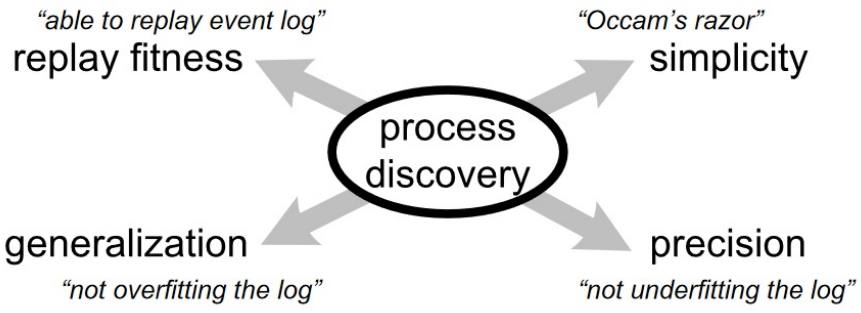


FIGURE 3.7 – Les Mesures d'évaluation des modèles de processus

Afin de déterminer la qualité des modèles extraits/découverts, nous utiliserons les trois métriques suivantes : la Justesse (connu par *Fitness*), la Précision et la Généralisation. Puis nous reviendrons sur le cas de l'algorithme *Fuzzy Miner* pour lequel nous ne pouvons pas appliquer ces mesures mais nous donnerons une piste afin d'évaluer sa qualité.

3.6.1 Justesse (*Fitness*)

La **Justesse (J)** décrit le fait qu'un modèle de processus découvert correspond bien aux logs. Une façon de calculer la Justesse est de déterminer dans quelle mesure les logs s'alignent avec le modèle obtenu (Adriansyah, 2014). Afin de présenter cette notion d'alignement, nous considérons une trace $t =< a, d, b, e >$ et deux modèles de processus M_1 et M_2. Nous considérons un tableau avec deux lignes où la ligne du dessus correspond à la trace de départ t et celle du dessous à la trace générée par le modèle. L'alignement optimal, γ_1, correspond au cas où il existe une trace t qui correspond exactement à une exécution de M_1.

$$\gamma_1 = \begin{array}{|c|c|c|c|} \hline \text{a} & \text{d} & \text{b} & \text{e} \\ \hline \text{a} & \text{d} & \text{b} & \text{e} \\ \hline \end{array}$$

Dans d'autres cas, si nous considérons que les traces générées par le modèle M_2 ne correspondent pas à la trace t de départ, il y aura des alignements multiples. Ainsi, avec cette méthode, nous pouvons distinguer deux sortes de mouvements. Des mouvements sur le modèle (*Moves on the Model*), qui consistent à modifier n éléments de la trace t pour correspondre au modèle M_2. Des mouvements sur la trace (*Moves on the Trace*), consistant à modifier n éléments de la trace générée par le modèle M_2 pour correspondre à la trace t (Adriansyah, 2014).

$$\gamma_{2a} = \begin{array}{|c|c|c|c|c|} \hline \text{a} & \gg & \text{d} & \text{b} & \text{e} \\ \hline \text{a} & \text{b} & \text{d} & \gg & \text{e} \\ \hline \end{array}$$

L'objectif des méthodes d'alignement est de trouver l'alignement optimal avec un coût réduit, c'est pour cette raison qu'un coût positif a été associé à chaque transformation (par exemple le symbole $\gg$). Après avoir appliqué l'algorithme d'alignement, la Justesse sera calculée sur chaque alignement afin de déterminer l'alignement optimal avec le meilleur taux de Justesse.

$$Justesse(t, M_2) = 1 - \frac{\delta(\lambda_{opt}^{M_2}(t))}{\delta(\lambda_{worst}^{M_2}(t))} \tag{3.1}$$

Où δ est la fonction de coût, $\lambda^{M_2}_{worst}(t)$ est le cas le plus défavorable où il n'y a aucune synchronisation possible entre la trace et le modèle. $\lambda^{M_2}_{opt}(t)$ représente les coûts obtenus pour chaque alignement optimal.

3.6.2 Précision (*Precision*)

La **Précision (P)** est une mesure qui indique si le modèle autorise uniquement le comportement observé dans les événements de traces d'exécution. Un modèle découvert peut avoir des problèmes de sur-précision si sa valeur de précision est importante. Dans cet article nous avons utilisé la mesure de précision basée sur le résultat obtenu par l'algorithme d'alignement. Nous considérons que les logs et le modèle sont alignés (Adriansyah, 2014). La précision entre les logs L et le modèle découvert M est donné par :

$$Precision(L, M) = \frac{1}{|E|} \sum_{e \in E} \frac{en_L(e)}{en_M(e)} \tag{3.2}$$

Où E est l'ensemble des événements dans les logs L, A l'ensemble des activités, $en_L(e) \subseteq A$ est l'ensemble des activités présentes dans les traces et $en_M(e) \subseteq A$ l'ensemble des activités présentes dans le modèle.

3.6.3 Généralisation (*Generalization*)

La **Généralisation (G)** est une métrique liée aux comportements non découverts. Le but est de découvrir un modèle de processus capable de généraliser des comportements qui ne sont pas apparus dans les logs. Cette métrique est utilisée pour identifier le problème de sur-précision. Dans le cas le plus favorable, le modèle découvert doit avoir une Généralisation faible et ne doit pas être restreint aux processus présents dans les logs (Adriansyah, 2014).

3.6.4 Mesures d'évaluation propres aux *Fuzzy models*

Les *Fuzzy model*s produits par l'algorithme *Fuzzy Miner* correspondent à une visualisation spécifique d'un processus. La qualité du modèle obtenu décrit le succès relatif de la procédure de simplification et la manière dont l'algorithme a pu extraire les informations les plus significatives. Afin d'évaluer la qualité des modèles découverts, deux mesures sont proposées : le Détail des nœuds et la Conformité (Günther, 2009) :

- Le **Détail des nœuds** décrit l'importance des activités visualisées dans le *Fuzzy model*, relativement aux activités agrégées ou supprimées. Les nœuds des activités visibles sont appelés nœuds explicites alors que les nœuds correspondant à un regroupement d'activité sont des nœuds implicites. La mesure de détail se calcule comme suit :

$$Detail = \frac{\sum v \in V^{s(v)}}{\sum n \in V^{s(n)}} \tag{3.3}$$

  Où N correspond à l'ensemble des nœuds primitifs (activités présentes, agrégées ou supprimées) du *Fuzzy model* F, $V \subseteq N$ est le sous-ensemble de tous les nœuds visibles et s est une fonction qui associe à chaque nœud primitif F une mesure d'importance (Günther, 2009).

- La **Conformité** est une mesure qui décrit l'alignement entre le *Fuzzy model* F et les logs L. Le comportement enregistré dans chaque trace dans les logs est reproduit dans le *Fuzzy model*. Chaque activité dans les logs qui n'existe pas dans le *Fuzzy model* sera comptée comme une déviation. La conformité entre les logs L et le *Fuzzy model* F est définie par :

$$Conformite = \frac{M(L) - d + 1}{M(L) + 1} \tag{3.4}$$

  Où M est le nombre total d'activités présentes dans les logs L et d est le nombre de déviations identifiées (Günther, 2009).

Dans cette section, nous avons présenté les différentes mesures d'évaluation des modèles de processus découverts. Comme nous l'avons indiqué dans l'introduction, nous allons maintenant passer à la deuxième partie de ce chapitre. La deuxième partie est consacré à la description de la comparaison des algorithmes de découverte. La deuxième partie consiste donc en premier lieu, à décrire les données que nous avons utilisé. En deuxième lieu, nous présentons les résultats d'évaluation des modèles de comportements découverts à l'aide des algorithmes de découverte des processus en se servant des mesures d'évaluation des modèles. De plus, une analyse qualitative des modèles de comportements découverts a été fournie à l'aide de l'expert (Suire et al., 2016).

3.7 Données de test

Les données sur lesquelles nous nous appuyons sont issues d'expérimentations sur une bibliothèque numérique menées avec 40 étudiants débutant leur spécialisation dans le champ du patrimoine et de sa mise en valeur (Suire et al., 2016). Leur niveau de connaissance du domaine est homogène et suffisant pour qu'ils soient capables d'adapter leur comportement de recherche d'information en fonction de la tâche à accomplir. Le corpus de documents proposés par la bibliothèque numérique expérimentale contenait des documents hétérogènes, textes généraux, articles scientifiques, documents iconographiques et sources primaires, relatifs au champ de spécialisation des participants.

Lors de cette expérimentation, les participants ont été confrontés à des tâches de recherche d'information dans les deux grandes catégories définies par Marchionini (2006) et conçues par un spécialiste de la question du patrimoine. La catégorie *Lookup* est constituée de tâches de recherche d'information simples relevant des sous catégories *recherche de faits* ou *recherche d'un document connu* des sous-catégories de *Marchionini*. Il s'agît par exemple de trouver une information précise telle qu'une date ou un lieu ou encore à trouver un document sur la base d'information connue. La catégorie *Exploratory* implique quant à elle la sous catégorie *apprentissage et investigation* de *Marchionini*. Elle consiste en une tâche intellectuellement plus complexe visant à trouver l'information nécessaire pour répondre à une question ouverte telle que la construction d'une problématique de recherche et la préparation d'une présentation. Ces activités se traduisent dans les logs par des requêtes, ainsi que des informations relatives aux différents choix de navigation des participants dans les résultats et les pages de la bibliothèque numérique (contenu des requêtes, usage des filtres, sélection d'un document, accès au contenu en plein texte, etc.).

Quantitativement, nous avons utilisé deux jeux de données qui contiennent un ensemble d'activités pour les deux catégories de recherche *Lookup* et *Exploratory*. Le jeu de données *Lookup* contient 102 traces, 1655 événements et 34 utilisateurs alors que le jeu de données *Exploratory* contient 29 traces, 1472 événements et 29 utilisateurs.

Comme mentionné, les logs de la catégorie *Lookup* contiennent 102 instances de processus (traces) de 80 variantes [2]. Six traces apparaissent plus d'une fois alors que 74 sont uniques. Dans les logs de la catégorie *Exploratory*, il y a 29 traces qui sont toutes uniques. C'est une propriété importante des *processus auto-définis* où

2. une variante de processus correspond à la nature de la trace

les traces récurrentes sont rares et peu redondantes contrairement aux processus structurés classiques, qui eux se répètent d'une manière identique. Les algorithmes de découverte de processus ont l'intérêt de trouver des modèles génériques à partir des traces non redondantes. Un autre aspect important des *processus auto-définis* par l'utilisateur vient de la répétition d'une activité au sein d'une même trace. Par conséquent, des traces longues peuvent être rapidement générées. Les boucles proviennent ainsi des hésitations des utilisateurs. Par exemple, trois traces dans les logs *Lookup* contiennent plus de cinquante activités.

3.8 Résultats d'évaluation des modèles de comportements découverts

La découverte des modèles de processus sur la base des traces décrites ci-dessus a été effectué à l'aide de l'outil *ProM* [3]. Afin d'analyser les événements avec *ProM*, nous convertissons les logs en fichier *CSV* (*Comma-Separated Values*). Les logs sont divisés en fonction du thème de recherche (*Lookup* et *Exploratory*) de telle sorte que les pré-analyses sont ignorées. L'outil *ProM* facilite la tâche de transformation des logs en fournissant des *plugins* de conversion qui transforment les fichiers *CSV* en fichier au format *XES* (section 3.3). Dans un fichier *XES*, chaque événement est donné avec son *timestamp*, l'identifiant de l'utilisateur, le nom de l'activité et la catégorie de chaque événement (cf. tableau 3.1).

Suite à la découverte des modèles, nous avons calculé les mesures d'évaluation des modèles découverts décrites dans la section 3.6. La mesure *Fitness* (F) a été calculée avec l'utilisation du *package* « *PNetReplayer* » [4] alors que les mesures *Precision* (P) et *Generalization* (G) ont été calculées à l'aide de *package* « *PNetAlignementAnalysis* ». Le tableau 3.3 présente les résultats d'évaluation de la qualité des modèles découverts à l'aide des algorithmes de découverte des processus cités dans la section 3.5.

Comme on peut le lire dans le tableau 3.3, la *Generalization* atteint toujours une valeur élevée quel que soit l'algorithme de découverte utilisé. Par contre, la *Generalization* n'a pas pu être calculée pour les algorithmes *Heuristic Miner* et α^{++}. De plus la *Precision* est toujours basse sauf pour l'algorithme *Genetic Miner*

3. ProM est un environnement extensible regroupant divers outils dédiés à la fouille de processus (`https://www.promtools.org/doku.php`)

4. `https://svn.win.tue.nl/repos/prom/Packages/`

	Lookup			*Exploratory*		
	F	P	G	F	P	G
Alpha ++	0.00	0.00	0.00	0.00	0.00	0.00
Heuristic Miner	0.00	0.00	0.00	0.00	0.00	0.00
Inductive Miner	**0.9886**	0.2391	0.9994	**0.9315**	0.1437	0.9992
Genetic Miner	0.9992	0.1800	0.9938	0.6232	0.8053	0.9963
Language Based Regions	0.6163	0.3825	0.9793	0.7835	0.1919	0.9622
State Based Regions	0.8995	0.4233	0.9957	0.9560	0.2942	0.9918
Fuzzy Miner	–	–	–	–	–	–

TABLE 3.3 – Résultats des mesures d'évaluation des modèles de processus découverts

dans le cas du jeu de données *Exploratory*. La *Fitness*, d'un autre côté, atteint son meilleur score avec l'algorithme *Inductive Miner* sur les deux jeux de données.

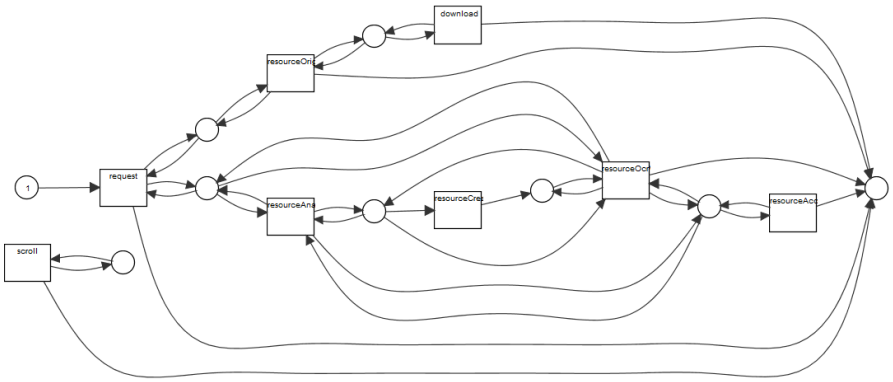


FIGURE 3.8 – Réseau de *Petri* découvert avec l'algorithme *Alpha miner* appliqué à la catégorie *Lookup*

Le premier modèle que nous présentons dans la figure 3.8 est un réseau de *Petri* découvert avec l'algorithme α^{++}. Le réseau de *Petri* découvert contient des parties non vivantes donc il est n'est pas *sound*. Le même problème a été constaté pour l'évaluation des modèles générés par l'algorithme *Heuristic Miner*. Ces derniers ne sont pas non plus des réseaux de *Petri sound* puisqu'ils ne possèdent ni de marquage initial ni de marquage final. De plus, le modèle ne présente pas de point de départ ni d'arrivée. En revanche, comme le montre la figure 3.9, le graphe de dépendance produit au tout début par l'algorithme *Heuristic Miner* est lisible et intéressant pour l'utilisateur final (l'expert de la bibliothèque numérique Suire et al.

(2016). Il est particulièrement utile pour observer les relations existantes entre les différentes étapes du processus. Si l'on tient compte du nombre d'itérations de chacune des étapes il permet d'établir rapidement le processus général et de comprendre l'origine des relations plus rares entre certaines étapes du processus.

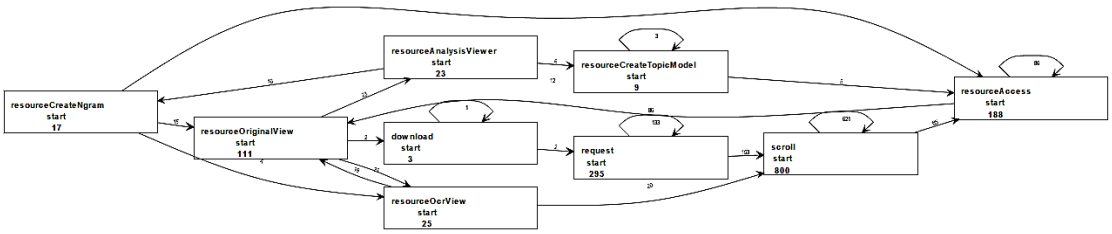


FIGURE 3.9 – Graphe de dépendance découvert avec l'algorithme *Heuristic Miner* appliqué à la catégorie *Exploratory*

Par exemple pour le modèle *Exploratory* dans la figure 3.9, on peut déduire que les utilisateurs lors d'une recherche de type *Exploratory* ont tendance à faire plus de recherche dans la liste des documents disponibles suite à une demande précise (800 fois pour l'activité scroll pour 29 étudiants). La recherche dans la liste se fait dans la plupart des cas d'une manière successive. On remarque aussi que la fréquence de téléchargement des documents suite à l'accès aux ressources est très faible.

La figure 3.10 présente le modèle de processus découvert par l'algorithme *Inductive Miner* sur le jeu de données *Lookup*. La mesure de *Fitness* est excellente sur les deux jeux de données *Lookup* et *Exploratory* (plus de $0,90$). De plus les modèles obtenus ont une haute *Generalization* et une *Precision* basse.

Étant donné que ce modèle (cf. figure 3.10) est capable de détecter les points de départ et de fin du processus et qu'il élimine les arcs les moins importants, il donne une idée très claire du processus globalement suivi par les utilisateurs de la bibliothèque numérique. Les étapes intermédiaires et les arcs qu'il présente permettent de comprendre les différents cheminements des utilisateurs pour accomplir leur tâche. Dans le cadre de la catégorie *Lookup* qui est une tache intellectuellement simple, la totalité des utilisateurs commencent toujours par une requête de recherche. Par la suite, un choix exclusif est mené entre une recherche dans la liste des documents (*scroll*) et un accès directe à une ressource. Il n'est cependant pas possible de mesurer la fréquence de chacune des options de navigation choisies par les utilisateurs et donc leur importance dans le processus.

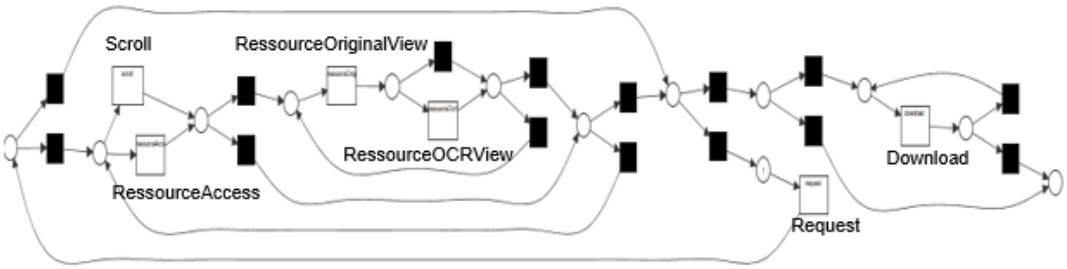


FIGURE 3.10 – Réseau de *Petri* découvert avec l'algorithme *Inductive Miner* appliqué à la catégorie *Lookup*

Quand à l'algorithme *Genetic Miner*, ce dernier découvre deux modèles avec une haute *Generalization*. Le modèle[5] *Lookup* donné par la figure 3.11 est capable de couvrir plus de logs que le modèle de processus *Exploratory*. Malgré un long temps de calcul, la *Fitness* pour le jeu de données *Lookup* atteint $0,9$ alors qu'elle baisse à $0,6$ pour le jeu de données *Exploratory*. La mesure *Precision* a donné un résultat opposé, le modèle découvert sur le jeu de données *Lookup* est moins précis (avec $0,1$) et grimpe jusqu'à $0,8$ pour le jeu de données *Exploratory* .

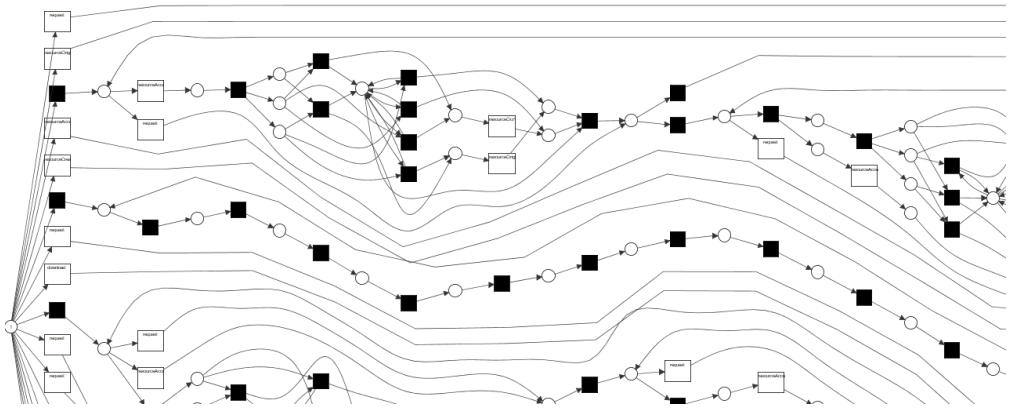


FIGURE 3.11 – Zoom sur le réseau de *Petri* découvert avec l'algorithme *Genetic Miner* appliqué à la catégorie *Lookup*

La représentation produite par ce modèle (cf. figure 3.11) est extrêmement précise et couvre l'intégralité des chemins utilisés par les utilisateurs pour accomplir

5. Zoom sur le réseau de Petri produit par l'algorithme *Genetic Miner*

leur tâche. La contrepartie de cette précision est la grande complexité du graphe produit. Celui-ci ne permet pas une analyse qualitative et ne permet pas de réduire les processus individuels de chaque utilisateur à un ou plusieurs processus généraux. Il ne permet pas de déterminer les relations les plus importantes entre les différentes activités, ni les stratégies les plus globalement adoptées par les utilisateurs. Ce problème de lisibilité est une lacune importante des résultats que nous avons présentés pour ce modèle.

Concernant l'algorithme *Language Based Regions*, chaque transition des modèles minés est accessible lors de l'évaluation de la conformité, alors qu'ils ne possèdent pas de marquage initial et ils adoptent une forme en fleur (c'est-à-dire que toutes les activités sont accessibles à partir d'un état de repos) (Van der Aalst, 2016). Pour les deux jeux de données, l'algorithme *Language Based Regions* produit des modèles avec une haute valeur de *Generalization* et un taux de *Fitness* était dans les deux cas $\simeq 0,7$.

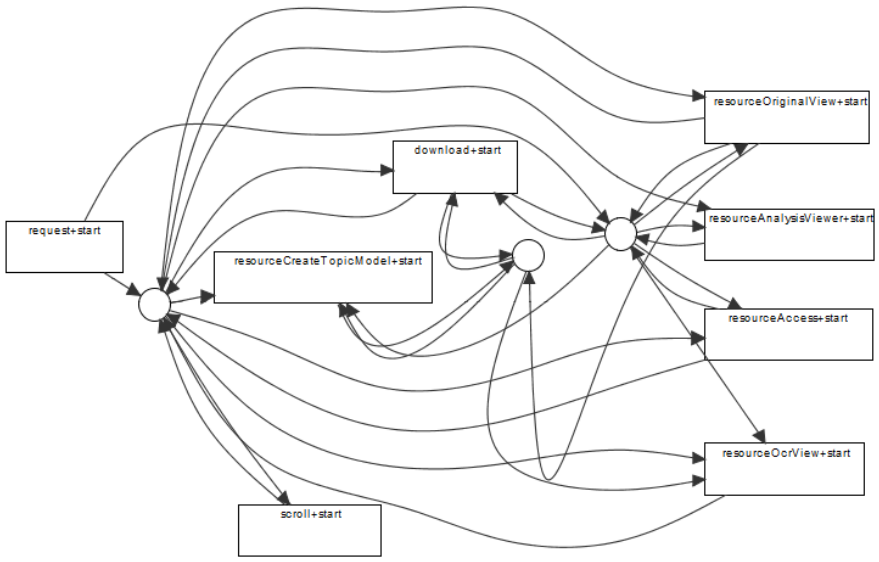


FIGURE 3.12 – Réseau de *Petri* découvert avec l'algorithme *Language Based Regions* appliqué à la catégorie *Lookup*

En accord avec ces résultats, la représentation graphique produite dans la figure 3.12 propose une généralisation intéressante pour la lecture de l'expert. Les différentes activités sont bien représentées et les relations entre elles sont très détaillées. Cette représentation est surtout utile pour identifier les étapes de navigation intermédiaires et les options de navigation qu'elles proposent. Toutefois,

à l'instar de la représentation produite par l'algorithme Inductive Miner, la fréquence des différents choix effectués par les utilisateurs et leur importance dans la résolution de la tâche ne peuvent pas être observées.

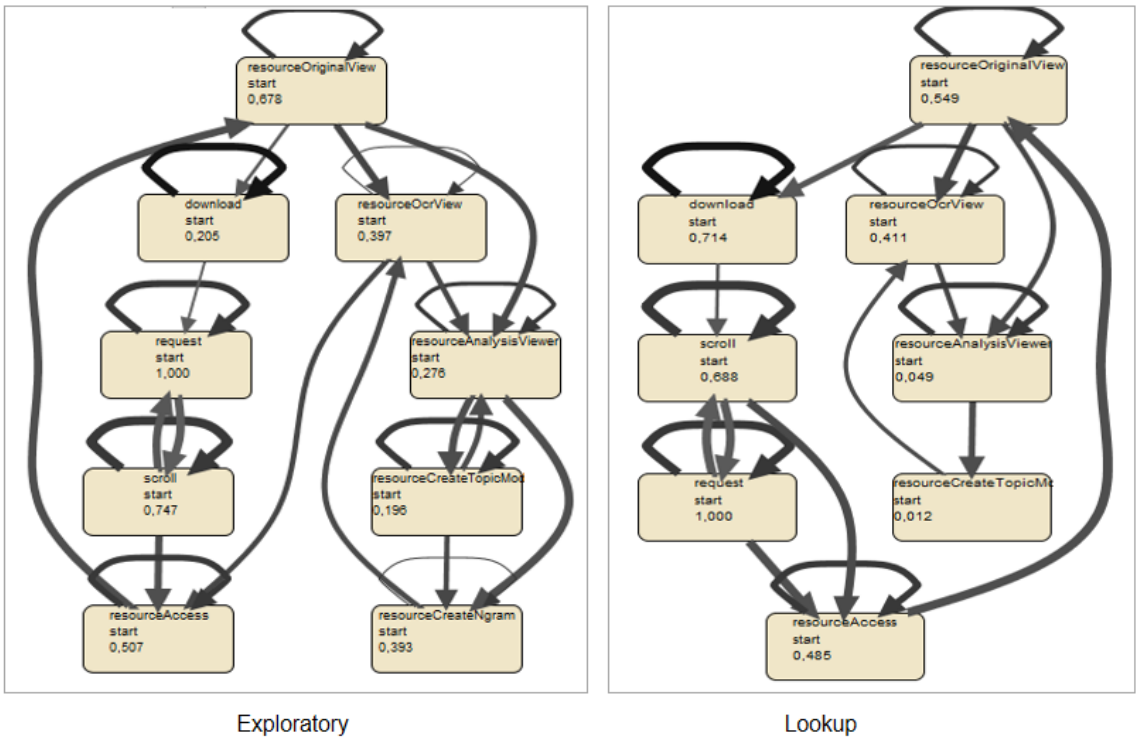


FIGURE 3.13 – Les *Fuzzy models* produits par l'algorithme *Fuzzy Miner* appliqué aux catégories *Lookup* et *Exploratory*

Les modèles obtenus avec l'algorithme *State Based Regions* ont été découverts en se fondant sur les options *Multiset abstraction* et le choix des traces partielles (cf. section 3.5). Les deux réseaux de *Petri* découverts possèdent un marquage initial. L'algorithme *State Based Regions* produit également deux modèles avec un niveau élevé de *Generalization*. De plus, rejouer les logs sur les traces générées par les modèles obtenues produit des bons résultats que ce soit pour le jeu de données *Lookup* que pour le jeu de données *Exploratory* (*Fitness* $\simeq 0.9$). À l'instar de la représentation graphique issue de l'algorithme *Genetic Miner*, la représentation obtenue à l'aide de cet algorithme est très complète. Elle permet à l'expert du domaine de visualiser très finement tous les différents choix opérés par les utilisateurs pour naviguer d'une activité à une autre. Il présente cependant les mêmes défauts, telle que la difficulté à exploiter cette représentation dans un objectif de

généralisation qui constitue une lacune importante dans les résultats des mesures comparatives que nous avons effectuées.

	Détail des nœuds	Conformité
Lookup	1.00	0.79
Exploratory	1.00	0.84

TABLE 3.4 – Mesures obtenues pour l'algorithme *Fuzzy Miner*

Les mesures de qualité des *Fuzzy models* ont été fournies directement par le package *Fuzzy Miner*. Les résultats donnés par le tableau 3.4 montrent que les nœuds primitifs visibles découverts dans les deux graphes sont à la fois importants et significatifs. De plus, rejouer les logs sur les traces issues des deux modèles minés donne de résultats satisfaisants (conformité $\simeq 0.8$). En corrélation avec ces résultats, ces modèles représentent une généralisation claire et intéressante pour les experts (cf. figure 3.13). La fréquence des choix faits par les utilisateurs est bien présentée et permet d'obtenir une vue globale des processus. Cependant, ces modèles ne permettent pas d'identifier les cas plus isolés et les processus les moins fréquents.

3.9 Conclusion

Dans ce chapitre, notre principal objectif était d'étudier la capacité de la fouille de processus dans la modélisation des interactions des utilisateurs de bibliothèque numérique, d'aboutir à des modèles de processus pertinents qui reflètent les pratiques réels des utilisateurs et à même de discriminer différentes pratiques de recherche d'information. Pour atteindre cet objectif, nous nous sommes fondés sur un état de l'art qui a montré l'efficacité de la fouille des processus. Les algorithmes de fouille de processus α^{++}, *Heuristic Miner*, *Inductive Miner*, *Fuzzy Miner*, *Regions Based* et *Genetic Miner* ont été appliqués à deux jeux de données en utilisant l'outil *ProM*. Les données de test sont constituées des traces d'interactions d'une taille maîtrisable générées suite à une expérimentation impliquant plusieurs utilisateurs et plusieurs tâches dans une bibliothèque numérique. Pour chaque algorithme, nous avons évalué les modèles de comportement découverts à l'aide des mesures d'évaluations décrites dans la section 3.6 afin de comparer les modèles minés et nous avons présenté une analyse qualitative des modèles découverts.

La principale conclusion de l'ensemble des mesures effectuées est que l'algorithme *Inductive Miner* donne de très bons résultats pour notre cas d'étude. Nous pouvons également constater que l'algorithme *Fuzzy Miner*, pour lequel il n'est pas possible d'appliquer les mêmes mesures, donne une très bonne vision globale des processus métier, avec une fonctionnalité de *zoom* et de ré-exécution des logs qui est d'un grand intérêt pour appréhender les phénomènes observés. Nous faisons le choix d'utiliser l'algorithme *Inductive Miner* par la suite car il possède le meilleur score de Justesse, donc qu'il correspond le mieux aux traces observées.

Cependant, si nous souhaitons passer à un cas réel, comme les traces de navigation des utilisateurs de *Gallica*, le portail web de la Bibliothèque Nationale de France, découvrir directement des modèles à l'aide des algorithmes de la fouille des processus ne sera pas possible. En effet, comme indiqué dans la section 4.5.1, les algorithmes de découverte de processus s'alimentent généralement d'un journal d'événement organisé d'une façon précise. L'objectif du chapitre 4, est de transformer les traces brutes en un ensemble de traces modélisées, compréhensibles par les algorithmes de découverte des modèles de processus, portant les informations nécessaires et permettant notamment d'identifier les données représentant les activités (événements).

Résumé exécutif

- Nous avons présenté les principes de la fouille de processus (présentation des algorithmes de découvertes de processus, description des formats de données nécessaires, description des mesures d'évaluations des modèles découverts).
- Nous avons comparé les différents algorithmes de découverte sur la base d'une expérimentation d'une complexité maîtrisable réalisée dans une bibliothèque numérique.
- Nous avons montré que la fouille de processus permettait d'extraire des modèles de comportements intéressants et que l'algorithme *Inductive Miner* était le plus performant car il possède le meilleur score de Justesse.

Chapitre 4

D'une trace brute à une trace modélisée : une méthodologie pour l'usage des logs bruts issus des bibliothèques numériques

Sommaire

4.1 Introduction

Dans le chapitre précédent, nous avons présenté nos motivations d'utiliser la fouille de processus pour extraire des informations pertinentes sur les parcours des utilisateurs d'une bibliothèque numérique. L'algorithme *Inductive Miner* (Leemans et al., 2013) était le plus performant (cf. table 3.3) dans notre contexte pour générer des modèles de processus pertinents qui reflètent les pratiques réelles des utilisateurs. Cependant, les données de test que nous avons utilisées proviennent des interactions d'une recherche d'information dans une bibliothèque numérique d'une complexité maîtrisable dont les traces d'événements sont dans un volume faible et facilement exploitable. Il en résulte qu'elles sont directement exploitables par les algorithmes de la fouille des processus.

Dans le but d'analyser des données réelles (les fichiers logs de *Gallica*) à l'aide des algorithmes de fouille de processus, il est tout d'abord nécessaire de structurer les données sous un format bien déterminé avec les informations appropriées. En effet, les logs de connexion constituent une bonne source de données et peuvent être considérés comme des traces brutes. L'analyse des traces nécessite leur modélisation. Nous parlons ici d'un passage de traces brutes en traces modélisées. Ces dernières doivent contenir au moins les trois informations suivantes dans l'ordre : *CaseID*, *Activity*, *Timestamp* (cf. figure 3.1).

L'objectif de ce chapitre est de définir les étapes nécessaires pour le passage d'une trace brute à une trace modélisée exploitable par les techniques de la fouille de processus. Avant de réaliser cette transformation, il est nécessaire de construire des connaissances sur *Gallica* et ses fichiers logs.

La première section 4.2 de ce chapitre est consacrée à la présentation de notre cas d'étude, la Bibliothèque Numérique de France *Gallica*. Les deux sections 4.3 et 4.4 sont dédiées à la description des logs de connexion fournis ainsi que les différents types de traces brutes, particulièrement les multiples requêtes que nous pouvons trouver dans ces traces. La section 4.5 présente, ensuite, notre méthodologie de transformation des traces brutes en traces modélisées. Une description statistique clôture la section.

4.2 Cas d'étude : Gallica

Le portail numérique *Gallica* est mis à disposition des utilisateurs en 1997. À l'origine, *Gallica* était une interface destinée à être consultée sur les terminaux

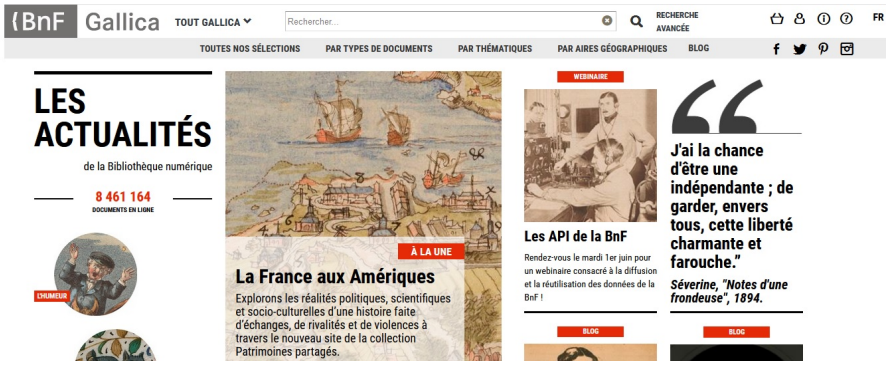


FIGURE 4.1 – Page d'accueil de *Gallica*.

locaux de la bibliothèque nationale, en appoint aux collections classiques. Avec le développement du web, *Gallica* est devenue une plate-forme bien répandue qui offre gratuitement à ses utilisateurs des millions de documents. En effet, plus de 7 millions de documents (images, livres, journaux, etc.) sont accessibles sur Gallica à l'heure de la rédaction de ce manuscrit. La figure 4.1 est une capture de l'interface d'accueil de Gallica. Elle donne accès à toutes les collections de la bibliothèque sans toutefois s'y limiter. Avec la popularisation du web, *Gallica* propose une interface très aboutie qui fournit de très nombreuses fonctionnalités. Elles permettent, parmi d'autres, de réaliser la recherche et l'affichage du plein texte, lorsque les collections s'y prêtent.

Chaque activité sur le site, y compris la consultation de documents est enregistrée. Les fichiers logs contiennent des informations brutes décrivant l'utilisation de la plate-forme : documents consultés, pages de référence, chemins à travers le site, le *timestamp* de chaque activité ainsi que le contexte du navigateur de l'utilisateur et bien d'autres choses encore (Nouvellet et al., 2017; Sumikawa et al., 2019). La figure 4.2 illustre un extrait réel des logs de connexion enregistrés pendant le mois d'avril 2017.

4.3 Description des logs de connexion

Avant de réaliser le passage d'une trace brute à une trace modélisée, nous avons commencé par observer les fichiers logs. Généralement, les logs de connexion à un site web sont enregistrés dans des fichiers qui contiennent toutes les requêtes *HTTP* reçues par les serveurs hébergeant le site. Dans le cas de *Gallica*, les logs n'étaient initialement conservés dans l'archive de la bibliothèque, qu'à des fins de sécurité et

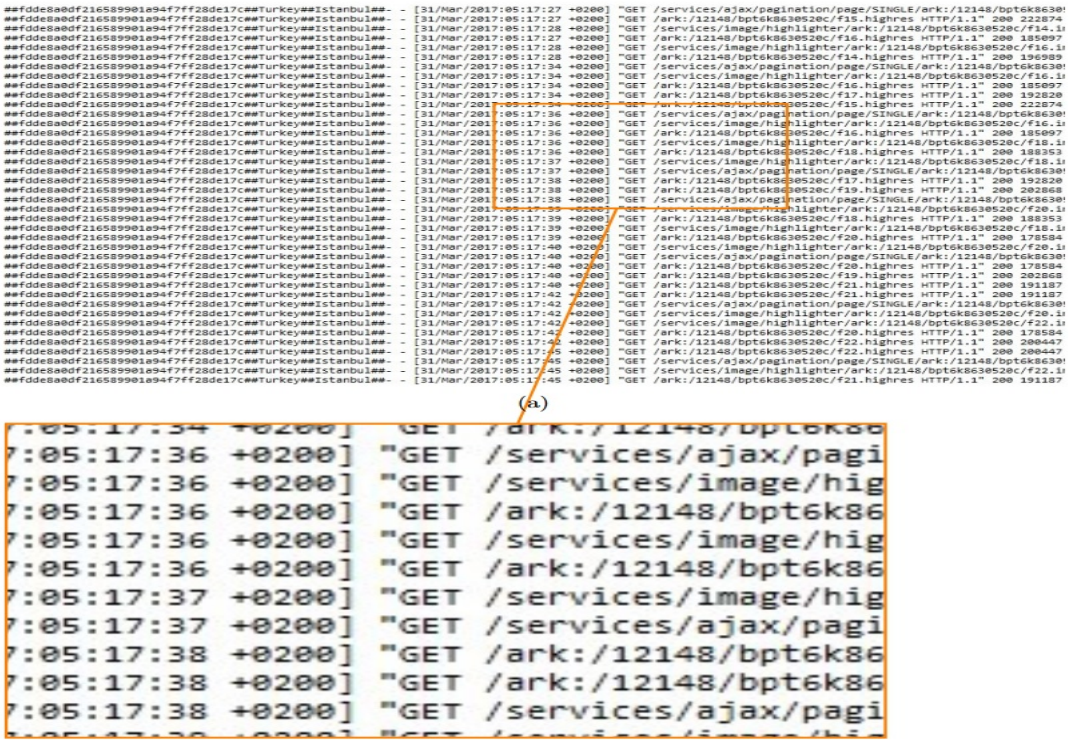


FIGURE 4.2 – Extrait du fichier log suite aux navigations enregistrées durant le mois d'avril 2017 sur *Gallica*

d'évaluation de la qualité de service. Un certain nombre d'ajustements techniques ont ensuite été opérés sur un ensemble de traces enregistrées sur 16 mois pour des fin de recherche. Ces ajustements concernent principalement l'anonymisation des données pour des raisons juridiques et éthiques. Suite à ces manipulations, acceptées et réalisés par le département des systèmes d'information de la bibliothèque, le jeu de données fourni enregistre donc les traces de navigation brutes de janvier 2016 à avril 2017.

Lorsqu'un utilisateur navigue sur le portail, le serveur reçoit une requête *HTTP* lui indiquant quelles informations/pages doivent être renvoyées à l'utilisateur. L'ensemble des requêtes est enregistré sous la forme de logs. Ces logs forment un ensemble de lignes propres à chaque requête *HTTP* et possèdent les informations suivantes :

- **L'adresse *IP*** : identifiant unique d'une connexion interne anonymisé par l'application d'une fonction de hachage préservant l'individualité.
- **La localisation** : la ville et le pays de la connexion.

- **Le *timestamp*** : l'heure, la minute et la seconde de la requête.
- **La requête *HTTP*** : une requête demandant l'accès à une page. Elle encode l'activité effectuée par l'utilisateur. Nous revenons sur les différents types de requêtes *HTTP* dans la section 4.4.
- **La réponse du serveur** : une valeur numérique décrivant l'état de la réponse du serveur (erreur ou succès).
- **Le site réfèrent** : la provenance de l'utilisateur.

La figure 4.3 montre un exemple précis d'une ligne de log d'un utilisateur anonyme connecté depuis le Japon. Le mot clé « *accueil* » trouvé dans sa requête indique que l'utilisateur commence sa navigation par la consultation de la page d'accueil.

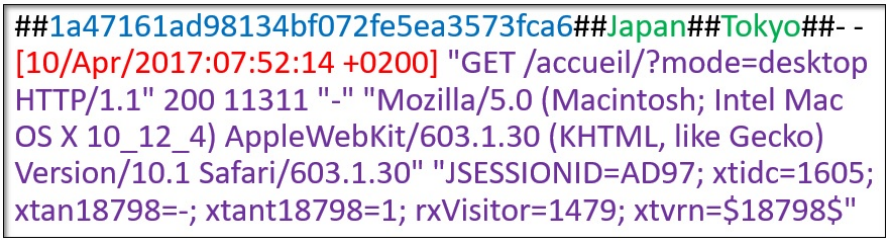


FIGURE 4.3 – Composition d'une ligne de log d'un utilisateur de *Gallica*. Adresse IP La localisation le *timestamp* La requête

4.4 Types de requêtes *HTTP*

Nous venons de décrire les informations mémorisées dans les fichiers logs. De part le fonctionnement sans état des sites web, il n'est pas possible de déduire l'ensemble des actions d'un utilisateur pour une navigation sur le site *Gallica*. Pour cela nous avons besoin de comprendre comment ces logs sont générés, c'est-à-dire que nous devons étudier les requêtes qui ont menées à l'écriture des données dans le fichier log. Nous décrivons, dans cette section, les différents types de requêtes *HTTP* (*Hyper Text Transfer Protocol*), que nous avons rencontrées lors de l'analyse des requêtes.

4.4.1 Requêtes liées au *web-design*

Les requêtes *web-design* sont généralement les requêtes de récupération de données à partir de scripts. Elles ne sont pas souvent porteuses d'informations utiles. Ces requêtes ne sont pas non plus exploitables dans notre travail pour la modélisation des traces. Par conséquent, elles sont toutes filtrées. Voici quelques exemples de requêtes *web-design* que nous avons identifiées dans les logs de *Gallica* :

- des scripts *Javascript* : GET /assets/static/javascripts/vendor/jquery-highlight.js,
- des scripts *CSS* : GET /assets/static/stylesheets/gallica-accueil.css,
- des images propres au design comme le logo : GET html/sites/default/files/visuel-applications.png.

4.4.2 Requêtes *HTML*

Ce type de requêtes est généralement lié aux pages web statiques, dont le contenu ne varie pas en fonction des caractéristiques de la demande, c'est-à-dire qu'à un moment donné, tous les utilisateurs qui demandent la page reçoivent le même contenu. Les pages web statiques de *Gallica* sont principalement les pages qui cataloguent les collections [1].

4.4.3 Requêtes *SRU*

La requête *SRU*[2] (*Search and Retrieve via URL*) assure la communication entre le serveur et l'utilisateur via le protocole *HTTP*. Ce type de requête intervient lorsqu'un utilisateur fait appel au moteur de recherche du *Gallica*. Une requête *SRU* peut utiliser soit une méthode *GET* ou une méthode *POST*. Une requête *GET* est généralement envoyée par le serveur à l'utilisateur pour la représentation de la ressource spécifiée, alors qu'une requête *POST* est envoyée par le client au serveur pour une recherche spécifiée. Une requête *SRU* indique donc l'utilisation du moteur de recherche interne à *Gallica*.

1. `http://gallica.BnF.fr/html/und/livres/livres`
2. `https://www.BnF.fr/fr/protocole-sru-vue-densemble`

4.4.4 Requêtes *ARK*

Les requêtes ARK[3] sont responsables de la consultation de documents. Elles associent des identifiants à la ressource consultée. En effet, le format ARK qui est créé en 2001 par la bibliothèque numérique de Californie, a vocation à identifier des ressources de tous types : physiques (livres papier), numériques (livres numérisés) ou immatériels (concepts). Son but est de fournir des identifiants adaptés aux besoins des producteurs et diffuseurs de données sur le web.

Nous venons de donner une description globale des logs de *Gallica*. Habituellement, une ligne dans le fichier logs est générée suite à une activité ou une interaction par un utilisateur. Les lignes correspondent à des requêtes successives de différents utilisateurs ordonnées par date de requêtes. Afin de former des traces brutes, nous regroupons ces requêtes par utilisateur dans une durée pré-défini (session). Comme nous l'avons signalé dans l'introduction, une trace brute n'est pas adaptée aux algorithmes de fouille de processus. Nous devons transformer ces traces afin d'en générer des modèles. L'objectif de la section suivante sera de décrire les étapes que nous avons définies pour transformer une trace brute en une trace modélisée.

4.5 Passage d'une trace brute à une trace modélisée

Une trace brute (nommée aussi *trace première*) correspond à la trace qui n'a subit aucune transformation ni interprétation. Elle représente la trace telle qu'elle a été collectée ou générée (Marty and Mille, 2009; Ho et al., 2017). La trace brute peut avoir un format défini par le modèle de collecte des traces. Ce type de traces est souvent stocké dans des fichiers textes où les différents champs d'un enregistrement sont séparés par un délimiteur spécifique tel qu'un espace, un dièse ou n'importe quel autre caractère qui peut être aisément identifié.

Afin d'analyser ces traces avec les algorithmes de fouille de processus, il est nécessaire de les organiser sous un modèle particulier comme le montre la figure 4.4. Ce modèle est obtenu suite à quatre étapes de transformation. Le résultat de cette transformation sera sous la forme des journaux d'événements stockés dans des fichiers sous format *CSV*.

3. `https://www.BnF.fr/fr/lidentifiant-ark-archival-resource-key`

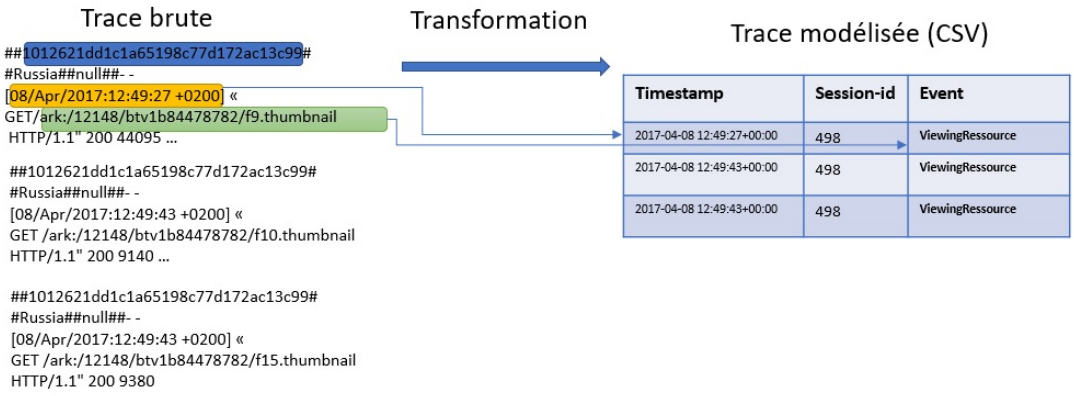


FIGURE 4.4 – Passage d'une trace brute à une trace modélisée

4.5.1 Notation

Les traces brutes contiennent les actions effectuées par les utilisateurs sur un système d'information. Chaque action est enregistrée à l'aide d'un outil de suivi, puis stockée dans une base de données et peut être par la suite interrogée pour effectuer des analyses (Poggi et al., 2013). Nous pouvons alors noter une **trace brute** dans des fichiers logs par $w_i =< l_1, l_2, ..., l_W >$ où chaque l_j représente une **ligne** du fichier logs. La ligne $l_j = (user_{id}; s; r)$ indique que l'**utilisateur** u, désigné par son identifiant $user_{id}$, a exécuté la **requête** r à l'instant indiqué par le ***Timestamp*** s. Nous notons les ensembles $U = \{u_1, u_2, ..., u_{|U|}\}$ et $r = \{r_1, r_2, ..., r_{|R|}\}$ pour désigner respectivement les ensembles des utilisateurs et des requêtes exécutées sur *Gallica*.

La **trace brute** w_i est convertie en une **trace modélisée** t_i d'un journal d'événements (définition 3.2.2) sous le format adéquat de l'entrée des algorithmes de la fouille de processus. Chaque t_i $(1 \leq i \leq k)$ contient un ensemble fini d'**événements** $t_i =< e_{i1}, e_{i2}, ...e_{in_i} >$ et n_i est un nombre fini d'événements. chaque événement de t_i, $e_{i1} = (c_j; a_w; s)$ doit faire référence à une **activité** a_w de l'ensemble d'activités $A = \{a_1, a_2, ..., a_{|A|}\}$, à un *Timestamp* s, et un identifient de trace *CaseId* c_j , qui identifie de manière unique l'utilisateur qui effectue l'activité.

Ainsi, nous obtenons L, le **journal des événements** des traces modélisées $L = \{t_1, t_2, ..., t_k\}$ qui est un ensemble fini de k traces qui correspondent aux différentes interactions des utilisateurs.

La transformation des traces brutes en traces modélisées procède en quatre

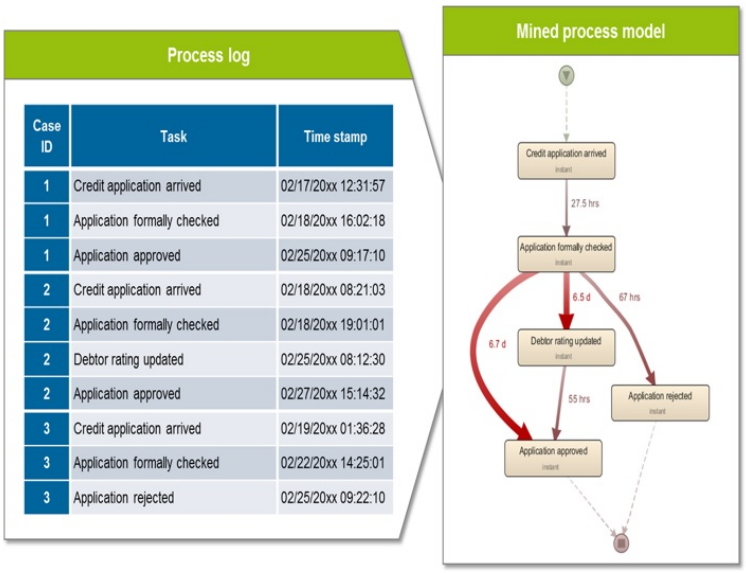


FIGURE 4.5 – Le format des logs adapté aux algorithmes de fouille de processus

étapes inspirées des travaux de Nouvellet et al. (2017); Poggi et al. (2013) : la visualisation, le nettoyage, l'étiquetage et la sessionisation. Dans ce qui suit, nous décrivons en détail chacune de ces étapes.

4.5.2 Visualisation des logs

La première étape consiste à avoir une vision globale des requêtes des utilisateurs et à comprendre leurs finalités. Compte tenu du grand nombre de requêtes, estimé à 500 millions par mois, nous utilisons un sous-ensemble de données pour mener nos expériences. Ce sous-ensemble peut être obtenu en filtrant les requêtes en fonction de leurs localisations, des dates, des adresses *IP* et de tous autres mots clés que nous pouvons identifier.

Dans le but de visualiser les requêtes en fonction des mots clés choisis, nous avons opté pour les les services *ELK*[4]. Pour pouvoir utiliser la pile de service *ELK*, nous avons commencé par l'installation du service *Filebeat*[5] sur notre système. À l'aide du service *Filebeat*, nous lisons chaque fichier de logs, ligne par ligne. Le service *Logstash* reçoit par la suite les logs et les analyses en champs et envoie les logs au service *Elasticsearch*. Ce dernier correspond à une base de données pour nos logs. Enfin, nous pouvons visualiser nos logs avec le tableau de bord *Kibana*

4. `https://www.elastic.co/what-is/elk-stack`
5. `https://www.elastic.co/guide/en/beats/filebeat/current`

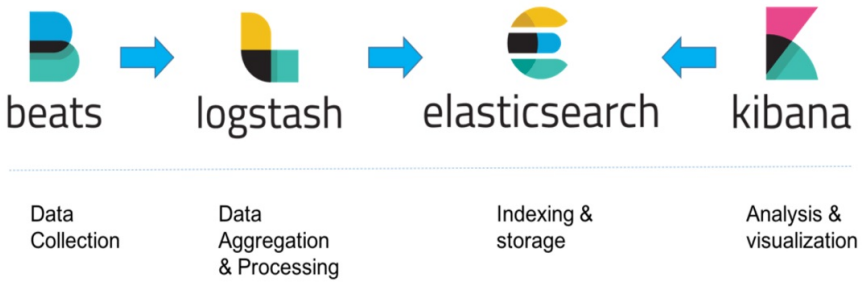


FIGURE 4.6 – Description de la pile de services pour le stockage et la visualisation des requêtes utilisateurs

(cf. figure 4.6).

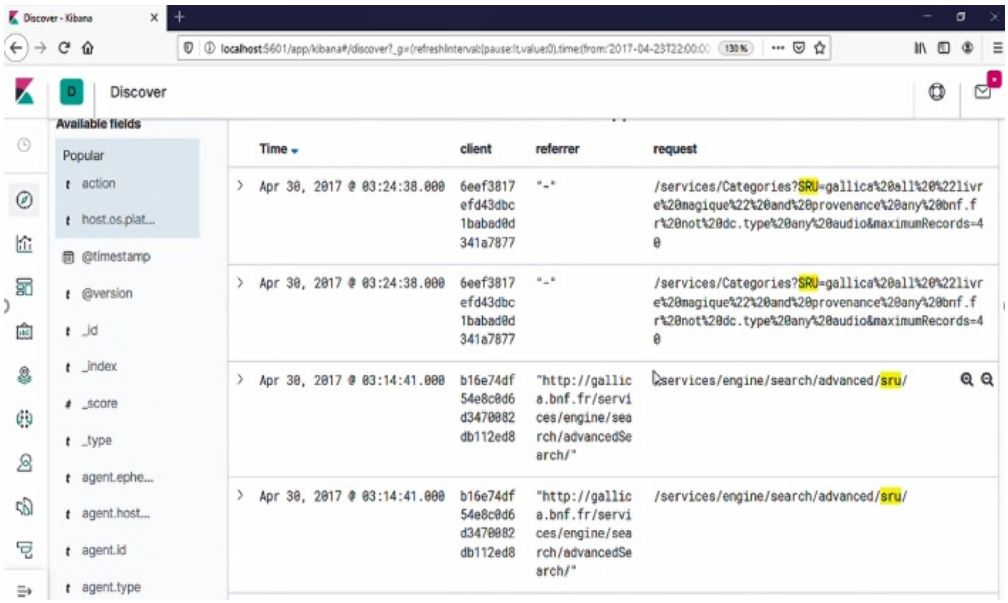


FIGURE 4.7 – Visualisation des requêtes utilisateurs

Suite à la visualisation des logs à l'aide de service *ELK*, notre choix du mot clé pour les logs à traiter doit maintenir la diversité des requêtes sur le jeu de données afin d'obtenir des données génériques. Après avoir analyser toutes les parties filtrées à l'aide de chaque mot clé, nous avons finalement utilisé le *timestamp* pour le choix du jeu de données à traiter. Nous n'avons conservé que les enregistrements sur la période d'un mois (avril 2017) qui a montré une grande variété de requêtes par rapport aux autres *timestamps*. Nous allons maintenant décrire la deuxième étape de pré-traitement de logs, le nettoyage des logs.

4.5.3 Nettoyage des logs

Cette étape consiste à nettoyer les fichiers logs en supprimant les lignes contenant des requêtes non pertinentes principalement liées à la conception de *Gallica* telles que les requêtes *web design*. Nous supprimons également les requêtes envoyées par des robots *bots-crawlers* qui peuvent parfois représenter 50% des requêtes dans les bibliothèques numériques (Nouvellet et al., 2017). À ce stade, nous disposons des requêtes *HTML*, *SRU* et *ARK*. Ces requêtes peuvent être principalement de type *POST* ou de type *GET*. Seules les lignes avec des requêtes *GET* sont conservées car nous nous intéressons aux aux réponses directes des serveurs.

Identifient	**Mot clé**
1	'js'
2	'css'
3	'assets'
4	'png'
5	'Bot'
6	'Spider'

TABLE 4.1 – Exemple des mots clés liés au bruit dans les requêtes

Afin de réaliser cette étape, nous avons construit un dictionnaire de mots clés liés à toute sorte de bruits. Le tableau 4.1 illustre quelques exemples de mots clés que nous avons déterminés pendant l'étape de la visualisation. Les mots clés *Bot*[6] et *Spider*, par exemple indique une recherche via un robot et ce genre de navigation nous ne permet pas d'avoir une idée sur les comportements réels des utilisateurs. Les autres mots clés sont liées aux requêtes *web-design* (cf. section 4.4.1). L'algorithme 1 illustre les étapes nécessaires pour réaliser le nettoyage des fichiers

6. Un robot internet (« bot » en anglais) est un logiciel qui réalise des tâches automatisées sur internet.

logs.

Algorithme 1 : Algorithme de détection du bruit

```
Input : F, dict
Output : F_withoutOutliers
Function OutlierDetection(F, dict) :
    for each item l in F do
        for each item elt in dict do
            if elt ∈ l then
                delete l ;
            end
        end
    end
    return F_withoutOutliers ;
End Function
```

Suite à l'étape de détection des valeurs aberrantes, nous avons supprimé $\sim$ 60% des requêtes. Nous avons finalement obtenu environ 102 millions de requêtes propres prêtes à être étiquetées.

4.5.4 Étiquetage des logs

Suite à la visualisation et au nettoyage de nos logs, chaque requête doit être normalisée en attribuant un nom à l'activité encodée dans la requête. Le nom de l'activité appartient à un ensemble d'étiquettes défini préalablement. En effet, les requêtes r sont peu informatives pour les concepteurs. Nous avons besoin des noms d'activités à la fois simple et compréhensible pour les concepteurs et les utilisateurs si nous voulons leurs faire des recommandations (amélioration du site).

L'étiquetage suit une fonction f qui transforme une requête r en une activité a. L'ensemble des activités $A = \{a_1, a_2, ..., a_{|A|}\}$ peut être dénommé de deux manières différentes : (i) manuellement ou (ii) en utilisant un regroupement non supervisé (Poggi et al., 2013). Plus précisément, en fonction de la méthode choisie pour sélectionner l'ensemble d'activités, f est formée en conséquence. (i) Si l'ensemble d'activités a été créé manuellement, f est un ensemble de règles de cette forme :

$$f(r) = \begin{cases} a_1 & if(Exp_1 \in r) \\ .. & \\ a_{|A|} & if(Exp_{|A|} \in r) \end{cases} \tag{4.1}$$

Où Exp_i peut être n'importe quelle expression, façonnée manuellement par les experts de sites web, en fonction de la structure et du contenu des différentes sections du site. Par exemple, l'activité « *Homepage* » correspond à une requête *HTTP* contenant les mots clés liées la consultation de la page d'accueil de *Gallica*. La figure 4.3, illustre bien notre exemple, la requête contient l'expression *accueil* qui indique que l'utilisateur a commencé sa navigation par la consultation de la page d'accueil de *Gallica*. La requête dans ce cas sera remplacée par l'activité« *Homepage* ».

D'autre part, (ii) l'application du clustering non supervisé à l'ensemble des requêtes r génère un ensemble de $|A|$ *clusters* de requêtes, chaque ensemble est représenté par une activité a. Par conséquent, f sera :

$$m(r) = \begin{cases} a_1 & if(r \in cluster_1) \\ .. & \\ a_{|A|} & if(r \in cluster_{|A|}) \end{cases} \tag{4.2}$$

Durant cette étape, nous avons choisi la première méthode, les noms d'activités ont été définis manuellement à l'aide d'un expert. En effet, le choix des noms des activités a été effectué en deux phases. En premier lieu, nous nous sommes comportés comme des utilisateurs de la bibliothèque numérique *Gallica*. Nous avons navigué sur le site à travers les différentes fonctionnalités proposées en consultant la requête générée suite à chaque activité exécutée. La consultation des requêtes a été réalisée en inspectant l'activité réseau du site web [7]. Cela nous a permis de définir un premier ensemble d'activités. En deuxième lieu, la visualisation des requêtes d'un ensemble d'utilisateurs dans le tableau de bord *Kibana* (cf. partie 4.5.2), nous a permis d'avoir une idée plus claire sur les différentes transitions entre les requêtes (le chemin de navigation). Cette phase nous a aidé à enrichir le premier ensemble par des activités supplémentaires effectuées par des utilisateurs réels.

Pour ce jeu de données, nous avons défini au total 9 noms d'activités que nous jugeons suffisantes pour étiqueter l'ensemble des requêtes réalisées sur *Gallica*. Quelques activités effectuées par les utilisateurs ont été toutes attribuées à une seule activité quand cette dernière peut inclure l'ensemble. L'activité définie pour la consultation des documents de *Gallica*, nommée *ViewingRessource* par exemple englobe d'autre sous activités liées à la consultation, comme le *scrooling*, le zoo-

7. `https://docs.microsoft.com/fr-fr/microsoft-edge/devtools-guide-chromium/network/`

ming, la pagination, etc.

À l'aide de notre fonction d'étiquetage, toutes les requêtes on été étiquetées. La description des différentes activités est fournie dans le tableau 4.2.

Activité	**Description**
HomePage	navigation en page d'accueil
ViewingRessource	consultation, défilement, zoom, etc.
CollectionNavigation	navigation dans les pages de la collection
AccessGallicaBlog	accès au blog du Gallica
DownloadRessource	téléchargement d'un document à partir des collections
GallicaSearchEngine	utilisation du moteur de recherche
AdvancedSearch	demande de l'option de recherche avancée
ResearchReportExtraction	extraction des rapports de recherche
OCRTextExtraction	extraction du texte brut à partir des images

TABLE 4.2 – Étiquetage des logs de connexion

Le tableau 4.3 montre les occurrences des activités triées de la plus fréquente à la moins fréquente. Comme prévu, l'activité *ViewingRessource* domine les autres actions dans les requêtes des utilisateurs puisque l'objectif principal de l'utilisation des bibliothèques numériques est de consulter des ressources. En se basant sur l'occurrence des activités, nous pouvons en déduire, par exemple, que pour certains utilisateurs, l'activité *HomePage* n'est pas essentielle pour consulter leurs ressources. Cela montre que certains utilisateurs peuvent atteindre leurs objectifs sans suivre toutes les étapes définies par les concepteurs.

Activité	**Occurrence**
ViewingRessource	99,050,711
DownloadRessource	1,266,193
GallicaSearchEngine	1,072,016
ResearchReportExtraction	490,068
CollectionNavigation	330,602
HomePage	290,605
OCRTextExtraction	139,655
AccessGallicaBlog	87,773
AdvancedSearch	57,105

TABLE 4.3 – Occurrences des activités dans l'ensemble de données

4.5.5 Sessionisation des logs

La sessionisation est la dernière étape de notre méthodologie de passage vers des traces modélisées. Elle consiste à diviser toutes les requêtes des utilisateurs en sessions. En effet, le comportement de recherche est souvent interprété à l'aide d'une séquence délimitée d'actions de recherche effectuées par un utilisateur. Les sessions ont été aussi étudiées afin de comprendre la recherche dans son contexte et de l'évaluer en termes de succès ou d'échec. Les sessions permettent ainsi de fournir des informations sur les visites répétées, d'examiner la modification des requêtes, d'obtenir des informations sur l'apprentissage dans la recherche, ou d'identifier les modèles dans le comportement de recherche (Walsh et al., 2019).

La sessionisation semble aussi nécessaire, vu qu'on ne peut pas demander à un utilisateur de se connecter et de dire qu'il commence une session de recherche et qu'il finit en disant qu'il a fini et note la qualité de la recherche. En d'autres termes, nous ne disposons pas de requêtes qui marquent le début et la fin de la navigation d'un utilisateur. Nous devons donc faire le découpage nous même avec le problème d'identifier une recherche unique sur une longue durée ou des recherches multiples sur une durée courte.

Pour notre cas d'étude, une session est définie comme une séquence de requêtes effectuées par le même utilisateur pendant un créneau horaire défini de sa visite sur le portail *Gallica*. D'une manière générale, une session débute par la connexion de l'utilisateur à la bibliothèque numérique et se termine lorsqu'il quitte celle-ci. Durant cette session, l'utilisateur effectue un ensemble de recherches. Une recherche est une situation qui débute par l'expression d'un besoin d'information sous la forme d'une requête saisie par l'utilisateur dans la barre de recherche ou bien par la sélection d'une ou plusieurs facettes.

La méthodologie choisie pour cette étape est importante car elle conditionne la cohérence de la trace (Walsh et al., 2019). Comme recommandé par Nouvellet et al. (2017), nous définissons une session comme une navigation partageant la même adresse IP et ne dépassant pas 60 minutes comme durée. Nous effectuons la sessionisation en regroupant les requêtes de la même adresse IP qui prennent moins de 60 minutes, sinon nous les divisons en plusieurs sessions si l'intervalle de durée entre les requêtes est supérieur à 60 minutes.

Suite à l'application des différentes étapes de pré-traitement des traces brutes, nous avons réussi à obtenir des traces modélisées stockées dans des fichiers *CSV*. Comme le montre la figure 4.4, les traces obtenues contiennent des valeurs de champs convenables aux algorithmes de découverte des processus. Chaque ligne

dans les fichiers *CSV* enregistre un événement exécuté qui contient des informations sur l'identifiant la session (*Session-Id*), l'activité ou l'événement (*Event*) exécuté par l'utilisateur, le *Timestamp* et d'autres champs liés à l'événement. Au final, nous avons obtenu plus de $1,7M$ traces modélisées. Le tableau 4.4 fournit quelques statistiques sur le jeu de données obtenu.

Dates de collecte de l'ensemble total de données	Janvier 2016 - Avril 2017
Dates de collecte de l'ensemble de données utilisées	Avril 2017
Nombre de requêtes	476 057 043
Nombre de requêtes nettoyées	102 787 467
Nombre de traces modélisées	1 719 657

TABLE 4.4 – Statistiques de l'ensemble des données réelles

4.6 Conclusion

Nous avons présenté dans ce chapitre notre cas d'étude, des logs de connexion générés suite à la navigation des utilisateurs de la bibliothèque numérique *Gallica*. Nous avons décrit les étapes nécessaires au traitement des traces brutes pour former des traces modélisées. Nous avons enfin présenté des statistiques sur l'ensemble de traces obtenus. Nous avons suivi une méthode de transformation itérative qui se décline sur quatre étapes : la visualisation, le nettoyage, l'étiquetage et la sessionisation. Plus d'$1,7$ million de traces modélisées ont au final été obtenues, réparties sur le mois d'avril 2017.

Pour un tel cas réel avec un grand volume de logs provenant d'une bibliothèque numérique complexe, nous constatons que le volume de traces à analyser reste encore très important. De plus, la diversité des utilisateurs ainsi que les tâches de recherche d'informations (cf. Chapitre 2) fait que les modèles découverts par les algorithmes de fouille de processus risquent de ne pas être en mesure d'extraire des informations pertinentes. De ce fait, nous proposons dans le chapitre suivant une nouvelle méthode de regroupement des traces avant la découverte des modèles de processus.

Résumé exécutif

- Nous avons présenté notre étude de cas. Il s'agit des traces d'interaction des utilisateurs de la Bibliothèques Nationale de France (*Gallica*) sur le mois d'avril 2017, soit plus de 476 M de requêtes.
- Nous avons décrit notre méthodologie, en quatre étapes, de passage des traces brutes à des traces modélisées exploitables par les algorithmes de la fouille de processus : la visualisation, le nettoyage, l'étiquetage et la sessionisation.

Chapitre 5

Une nouvelle méthode pour regrouper les parcours utilisateurs similaires dans les bibliothèques numériques

Sommaire

5.1 Introduction

Notre objectif est de définir une méthodologie pour extraire des modèles des parcours des utilisateurs d'un système d'information. Nous nous sommes concentrés sur les systèmes d'information dans lesquels l'utilisateur n'est pas un expert qui connaît les processus définis à l'avance par l'entreprise. Nous avons choisi de nous concentrer sur les bibliothèques numériques (cf. Chapitre 2) qui correspondent à ces caractéristiques. Notre choix a été appuyé par la nature des traces générées par les utilisateurs lors de leurs navigations. Ces traces constituent un défi particulier vu que, en premier lieu, elles sont à la fois bruitées, incomplètes et forcément non structurées [1]. En deuxième lieu, le volume des traces collectées est très important et en dernier lieu les usages des bibliothèques numériques nous ont appris que les utilisateurs peuvent être répartis en plusieurs familles typiques.

Dans le chapitre 1, nous avons souligné que les concepteurs disposent de nombreuses méthodes en matière d'analyse et de modélisation des comportements des utilisateurs suite à l'utilisation des bibliothèques numériques. En revanche, la fouille de processus n'a jamais été sollicitée dans ce contexte malgré sa capacité à produire des modèles de navigation complets.

Les techniques de la fouille de processus ont prouvé leurs efficacités dans la découverte des modèles de comportement sur une simple quantité de traces issues d'une expérimentation effectuée dans une bibliothèque numérique (cf. Chapitre 3). L'application directe de ces techniques sur un ensemble large de traces issues de *Gallica* se confronte alors à deux problématiques. D'une part, l'application de la fouille de processus sur un ensemble large de traces hétérogènes sur *Gallica* engendre généralement des modèles complexes (dits des *modèles Spaghetti*). D'autre part, les traces nécessitent une étape de pré-traitement pour les rendre compatibles avec les algorithmes de fouille des processus. Bien que nous avons répondu à la deuxième problématique par le nettoyage et la transformation des traces brutes en traces modélisées exploitables (cf. Chapitre 4), nous proposons, dans ce chapitre, une réponse à la première problématique qui s'appuie sur une étape de regroupement des traces avant la génération de modèles. L'hypothèse sous-jacente à cette idée est que la proposition de plusieurs sous modèles de qualité suffisante est meilleure que la génération d'un seul modèle moyen voire mauvais en termes de rapidité et performance. En outre, les sous modèles permettent aux concepteurs de visualiser et analyser des comportements similaires de groupes d'utilisateurs

1. Les utilisateurs appliquent généralement différents chemins d'exécution afin d'atteindre leurs objectifs. Ainsi, chacun d'eux construit son propre parcours.

et ainsi de proposer des solutions spécifiques aux besoins de chaque groupe sans impacter la navigation des autres groupes.

La méthode que nous proposons pour le regroupement des traces doit répondre aux spécificités des traces issues des bibliothèques numériques en terme de volume de données, temps d'exécution et qualité des modèles extraits. L'évaluation de notre méthode se confronte de l'absence de ressources de traces réelles annotées où chaque trace est associée à une classe, nous avons alors simulé un ensemble de traces issues d'une bibliothèque numérique et annoté manuellement ces traces avec les classes appropriées afin de déterminer la pertinence de notre regroupement.

Dans ce chapitre, nous commençons par présenter, dans la section 5.2, un état de l'art sur les principales techniques de regroupement des traces d'exécution ayant pour objectif l'application de la fouille des processus. Nous décrivons, par la suite, notre nouvelle méthode de regroupement basée sur l'extraction et l'encodage des sous-séquences fréquentes à partir des traces d'interactions dans la section 5.3. L'évaluation de notre méthode a été effectuée à deux niveaux, le niveau de regroupement des traces et le niveau de la qualité des modèles de processus en section 5.4.

5.2 État de l'art sur le regroupement des traces

Comme nous l'avons indiqué dans le chapitre 3, la fouille de processus est une discipline analytique relativement jeune qui sert de passerelle entre l'exploration de données et la gestion des processus d'entreprise (Van der Aalst, 2016). La fouille de processus a été utilisée pour l'extraction des informations sur la façon dont les processus fonctionnent dans la vie réelle, sur le nombre de déviations par rapport aux prévisions et sur la façon dont ces processus peuvent être améliorés. Cependant, lorsqu'il s'agit des données volumineuses et complexes avec un comportement capturé par le journal des événements très hétérogène, la prise en compte de l'ensemble des traces peut induire en erreur les algorithmes de fouille de processus. Ils génèrent généralement des modèles *spaghetti*, trop compliqués, ou des modèles généraux, trop légers. Dans de tels cas, de la découverte des modèles *spaghetti*, il n'est pas possible d'extraire des informations ou des caractéristiques pertinentes sur le modèle découvert.

Une solution possible pour éviter ce type de modèles, est d'identifier les parcours similaires ou avec des variations proches. Ces parcours partagent des comportements homogènes et devront être regroupés. La figure 5.1 illustre le processus générale de regroupement des traces d'exécution employé par la communauté de

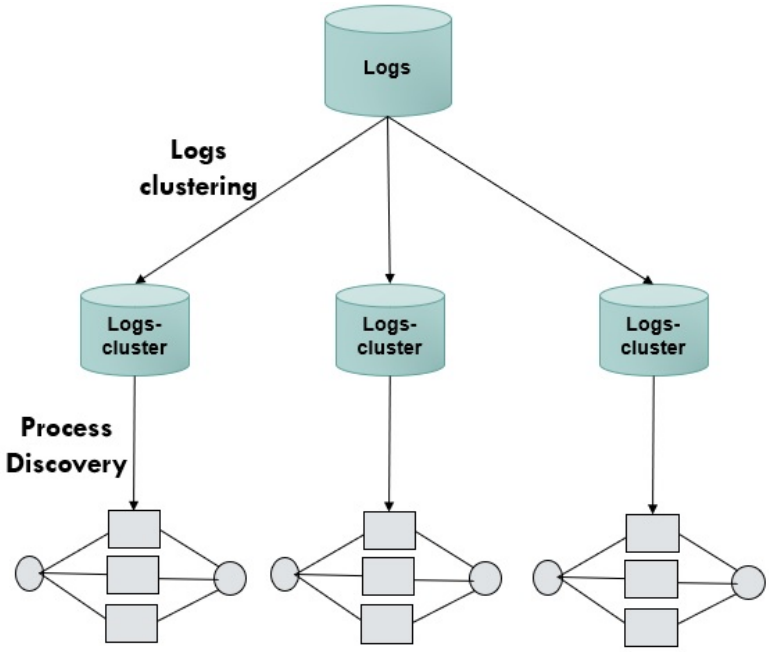


FIGURE 5.1 – Processus global du regroupement des traces d'interactions

la fouille des processus (Van der Aalst, 2016). La découverte des modèles de processus ne sera plus donc effectuée sur l'ensemble total des traces dans le journal d'événement, mais plutôt sur chaque sous-ensemble généré par le regroupement.

Dans notre contexte, pour faire les choix les plus adaptés à nos objectifs, nous nous focaliserons en priorité sur les travaux de regroupement des traces qui ont été spécifiquement développés pour le domaine de la fouille des processus.

Dans cette section, nous examinons les travaux précédents sur le regroupement des traces. Zandkarimi et al. (2020) propose une investigation récente dans ce contexte et décrit 103 travaux de recherche pertinents dans le regroupement de traces entre 2004 et 2020. Dans notre travail, nous catégorisons ces méthodes en quatre approches principales :

- Approche basée sur la similarité entre les traces (***Trace-based***)
- Approche basée sur la similarité entre les descripteurs (***Feature-based***)
- Approche basée sur la qualité des modèles (***Model-based***)
- Approche hybride (***Hybrid clustering***)

.

Dans le reste de cette section, nous décrivons en détail chacune de ces approches.

5.2.1 Approche basée sur la similarité entre les traces

Les travaux de cette approche regroupent les traces en se basant sur leurs similarités syntaxiques (Bose and Van der Aalst, 2009 a; Chatain et al., 2017). Le calcul de la similarité syntaxique entre les traces est inspiré de la distance de *Levenshtein* entre deux chaînes de caractères (Bose and Van der Aalst, 2009 a). Une trace peut être éditée en une autre trace en substituant, ajoutant ou supprimant des événements. Techniquement, la distance d'édition entre deux traces est le nombre minimum d'opérations d'édition nécessaires pour transformer une trace en une autre. Plus la distance d'édition est faible, plus les traces sont similaires. Par exemple, pour deux traces t_1 et t_2, les opérations d'éditions sont calculées selon les activités $A \cup \{-\}$ ou $-$ représente le *gap*. Pour $a, b \in A$, le couple

- (a, a) représente une correspondance d'activités entre t_1 et t_2 sur la même position $t_1(i)$ and $t_2(j)$. Une correspondance peut être considérée comme une substitution d'une activité par elle-même.
- $(a, -)$ représente la suppression de a dans t_1 à une certaine position $t_1(i)$
- $(-, b)$ représente l'insertion de b dans $t_1(i)$
- (a, b) représente le remplacement/substitution de a dans t_1 avec b à une certaine position $t_1(i)$ ou $a \neq b$

Parmi les premiers travaux basés sur la similarité entre les traces, nous pouvons citer le travail de Bose and Van der Aalst (2009 a) qui a proposé une méthode générique basée sur la distance d'édition afin de traiter la sensibilité de la fonction de coût. Cette dernière mesure le nombre d'opérations nécessaires pour passer d'une trace à une autre. Ils ont déterminé le coût en prenant en compte le contexte d'un événement dans une trace. Les traces de coût minimal ont été regroupées. Dans un travail connexe, Di Francescomarino et al. (2016) ont utilisé la distance d'édition afin de générer une matrice de similarité entre les traces. Puis, ils ont utilisé un algorithme de *clustering* pour regrouper les traces en se basant sur la densité des instances de processus similaires. La complexité quadratique du temps pourrait être le principal inconvénient de leurs approches. Quant à Chatain et al. (2017), ils ont supposé qu'un modèle de processus standard est disponible et ils ont

aligné les traces avec les exécutions de ce modèle. Chaque groupe de traces est lié à une variante de trace [2] qui correspond à une exécution complète du modèle, qui sert de centroïde. Les traces dans le *cluster* doivent toutes être suffisamment proches du centroïde (en termes de distance entre les séquences). Ceci permet d'identifier les exécutions du modèle qui reproduisent les traces typiques observées, et aussi d'isoler les traces déviantes qui sont trop éloignées de ce que le modèle décrit. De cette façon, même en cas de déviations, de traces incomplètes ou bruitées, ou même de dérives dans le modèle de processus (par exemple, en traitant le processus des ventes d'hiver, alors que les traces correspondent aux ventes d'été), une explication du processus des traces dans chaque *cluster* est disponible, afin que les parties prenantes puissent les relier de manière plus fiable au processus sous-jacent.

5.2.2 Approche basée sur la similarité entre les descripteurs

La deuxième approche est la similarité basée sur des descripteurs (ou caractéristiques) de traces. Elle consiste à convertir chaque trace en un vecteur qui représente une ou plusieurs caractéristiques quantitatives de cette trace. La similarité entre deux traces est donc déduite de la similarité entre leurs vecteurs correspondants. Le calcul de la similarité entre les traces se basent alors sur des opérations vectorielles. Parmi les descripteurs utilisés dans les méthodes existantes, nous trouvons, par exemple, la fréquence des événements ou la fréquence de la relation de succession directe entre les événements Song et al. (2008); Bose and van der Aalst (2009 b).

Song et al. (2008), ont introduit une nouvelle approche pour diviser le journal en sous-ensembles homogènes de traces en utilisant des phénomènes fréquents trouvés dans les traces tels que les activités, leurs transitions, leurs auteurs et leurs performances. Cette approche est connue par le profilage de trace (*Trace Profiling*). Selon les auteurs, le profilage de traces peut être décrit comme la mesure d'un ensemble de traces avec un certain nombre de profils, résultant un vecteur agrégé (contenant les valeurs pour chaque élément mesuré dans un ordre défini). Ces vecteurs peuvent ensuite être utilisés pour calculer la distance entre deux traces quelconques, en utilisant une métrique de distance. Dans le cadre de leurs études, les auteurs ont transformé des traces provenant d'un système d'information hospitalier en un ensemble de caractéristiques telles que l'occurrence d'événements

2. Une variante de trace est un type de traces, par exemple, si le journal d'événement L est composé de trois traces $t_1 = \{a, b, c\}$, $t_2 = \{a, b\}$, $t_3 = \{a, b, c\}$, la première variante de trace est $\{a, b, c\}$ et la deuxième est $\{a, b\}$.

individuels dans l'ensemble des traces d'exécution ou l'occurrence des successions directes entres les événements. Chaque trace est convertie en un vecteur pour calculer sa distance avec d'autres traces. Les auteurs ont réalisé plusieurs expériences avec trois mesures de distance différentes et quatre techniques de regroupement. Le problème de cette approche pourrait être la malédiction de la dimensionnalité. Trop de dimensions font que la distance entre plusieurs traces semble égale et que les résultats du *clustering* sont imprécis. Ceravolo et al. (2017) ont également adopté le profilage de traces en introduisant la notion du position profile. Cette notion consiste à définir un triplet qui prend en compte l'occurrence d'une activité, sa position et sa fréquence d'apparition dans cette position.

Bose and van der Aalst (2009 b) ont appliqué une technique similaire sur des sous-parties plus larges de traces. Ils ont évalué l'occurrence des motifs plus complexes comme les répétitions (c'est-à-dire les n-grammes observés à différents emplacements de la trace avec des différents longueurs n). Les n-grammes observés sont les différents patterns existants dans les traces tels que les répétitions maximales (*Maximal Repeat Set*), ainsi que les répétitions super maximales (*Super Maximal Repeat Set*) et les répétitions presque super maximales (*Near Super Maximal Repeats Set*) pour créer des ensembles de caractéristiques qui constituent le vecteur d'une trace particulière. La figure 5.2 présente les différents patterns extraits de chaque trace dans le journal d'évènements. Pour la trace T_1, par exemple, l'ensemble des répétitions maximales est $\{a, b, bcd\}$. Puisque la répétition maximale b est compris dans la répétition maximale bcd, b ne remplit pas les conditions requises pour être une répétition super maximale. L'occurrence de la répétition maximale b à la position 6 dans T_1 ne se chevauche avec aucune autre répétition maximale. Par conséquent, b est qualifié pour être une répétition presque super maximale.

Id	Trace	Maximal Repeat Set	Super Maximal Repeat Set	Near Super Maximal Repeat Set
T_1	aabcdbbcda	{a, b, bcd}	{a, bcd}	{a, b, bcd}
T_2	dabcdabcbb	{b, dabc}	{dabc}	{b, dabc}
T_3	bbbcdbbbccaa	{a, b, c, bb, bbbc}	{a, bbbc}	{a, c, bbbc}
T_4	aaadabbccc	{a, b, c, aa, cc}	{b, aa, cc}	{a, b, aa, cc}
T_5	aaacdcdcbedbcc-badbdebdc	{a, b, c,d, e, aa, bd, cb, db, dc, cdc}	{e, aa, bd, cb, db, cdc}	{a, c, e, aa, bd, cb, db, dc, cdc}

FIGURE 5.2 – *Maximal*, *Super maximal*, and *Near Super Maximal repeats* extraites à partir des traces

Pour les ensembles de caractéristiques (patterns) extraites, chaque trace est

transformée en un vecteur dont les valeurs correspondent au nombre d'occurrences de chaque caractéristique dans cette trace. Néanmoins, travailler dans l'espace des n-grammes entraîne une surcharge de calcul. En outre, la sélection d'une valeur appropriée pour n n'est pas triviale.

5.2.3 Approche basée sur la qualité des modèles

Cette approche a mis l'accent directement sur la qualité des modèles découverts et la distribution des traces parmi les groupes de traces. Elle suppose que les modèles de processus précis et pertinents sont découverts à partir de sous-logs homogènes (Veiga and Ferreira, 2009). Le modèle de processus est considéré comme une entrée pour le regroupement des traces. Ces traces sont réutilisées pour extraire des modèles de processus. Les groupes obtenus dépendent fortement des résultats de mesures d'évaluation de la qualité des modèles découverts (De Weerdt et al., 2013; De Koninck and De Weerdt, 2019).

Le travail de Veiga and Ferreira (2009), par exemple, est considéré parmi les premiers travaux sur le regroupement des traces basés sur le *Model-based similarity*. Les auteurs ont combiné le regroupement de traces avec des modèles de *Markov* de premier ordre en utilisant une approche hiérarchique. Initialement, des groupes aléatoires sont construits et les traces sont distribuées entre eux. Par conséquent, les modèles de chaque groupe (la chaîne de *Markov* de chaque *cluster*) sont évalués. Ensuite, de manière itérative, les traces sont réassignées aux groupes et l'évaluation est faite à nouveau jusqu'à ce que l'algorithme converge et que les modèles ne changent plus. Les auteurs ont également ajouté une étape de pré-traitement pour traiter le bruit en éliminant les séquences peu communes et rares qui affectent les modèles probabilistes. Cette étape consiste à éliminer les séquences de traces d'une faible occurrence d'apparition. Suite au regroupement des traces, un algorithme de découverte de processus peut être utilisé pour l'obtention des modèles.

Quant à De Weerdt et al. (2013), leur méthode *ActiTraC* vise à trouver la distribution optimale des traces entre les groupes qui conduit à une qualité meilleure des modèles de processus découverts. En d'autres termes, ils ne cherchent pas à trouver la similarité entre les traces, mais plutôt à regrouper les traces qui se trouvent dans un certain modèle de processus. Les auteurs appellent ce problème la divergence entre le biais d'évaluation et le biais de clustering et proposent une nouvelle approche basée sur l'apprentissage actif qui prend d'abord des cas uniques (des traces) et, sur la base de leur distance ou de leur fréquence, ils sont regroupés en tant que *clusters* primaires. Les groupes n'acceptent pas de nouvelles traces,

sauf si leurs *Fitness* est optimisée, sinon les traces sont allouées à un groupe de destruction ou sont distribuées de manière égale entre les autres groupes. La complexité de calcul pourrait être le principal inconvénient de leur approche.

5.2.4 Approche hybride

En plus des méthodes déjà établies dans la littérature, des nouvelles techniques basées sur la combinaison des différentes types d'approches ont été introduites.

Le travail de Hompes et al. (2015), par exemple, combine les approches *Model-based* et les *Feature-based*. Les traces sont ensuite transformées en vecteurs en se basant sur le profilage de traces et une matrice de similarité est calculée sur les vecteurs en appliquant la mesure de similarité cosinus 5.3.2.1. Finalement, la matrice de similarité est l'entrée de l'algorithme de *clustering* de graphes *MCL*[3] (*Markov Cluster Algorithm*) (vanDongen, 2000). L'algorithme de *clustering* de graphes est capable de trouver les variations et les déviations d'un processus sur la base d'un ensemble de perspectives sélectionnées. À l'aide de cette méthode, il est possible de comprendre comment et pourquoi le comportement a changé en comparant les changements dans les regroupements des différentes partitions des traces. Suite au regroupement des traces, les auteurs ont utilisé l'*Heuristic Miner* 3.5.2 pour la découverte des processus. La figure 5.3 résume le processus complet de cette approche hybride.

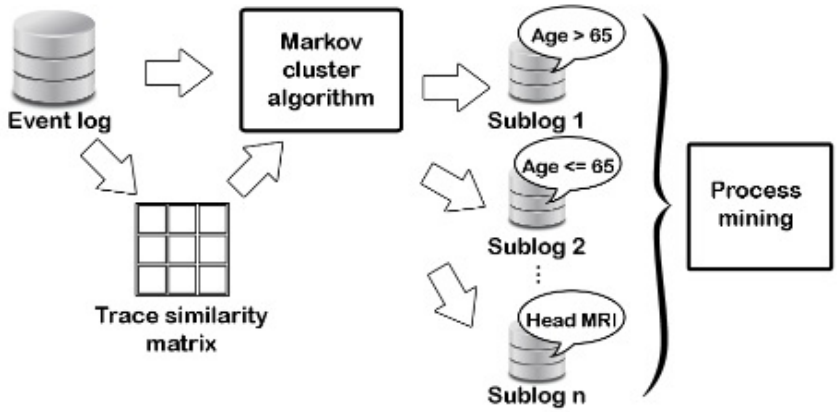


FIGURE 5.3 – Aperçu de l'approche hybride de regroupement des traces (Hompes et al., 2015)

3. https://micans.org/mcl/

De Koninck and De Weerdt (2019) ont présenté une nouvelle méthode de regroupement de traces nommée *TraCluSI* (*Trace Clustering using Super-Instances*) qui combine également à la fois les approches *Model-based* et les *Feature-based*. Dans ce travail, les traces (instances de processus) sont convertis, dans un premier temps, en des vecteurs en se basant sur le profilage de traces (Song et al., 2008). En deuxième lieu, un algorithme de clustering est utilisé pour diviser l'ensemble de traces sur un nombre de *clusters* k. Suite à cette étape, pour chaque groupe de trace, une sélection des traces les plus importantes, dites des super-instances, sera effectuée en se basant sur des stratégies (la fréquence ou la centralité). Si la fréquence est choisie, les super-instances seront sélectionnées sur la base de la fréquence des instances de processus distinctes de toutes les instances de chaque *cluster*. Si la centralité est choisie comme stratégie, la super-instance sera sélectionnée en fonction de sa distance par rapport au centroïde du *cluster* : la super-instance qui est la plus proche du centroïde est sélectionnée. Indépendamment de stratégie choisie, toutes les traces non sélectionnées, sont temporairement stockées séparément. Enfin, la méthode utilisée dans *ActiTraC* (De Weerdt et al., 2013) est appliquée pour distribuer les traces entre les groupes des super-instances.

La table 5.1 présente une vue d'ensemble des méthodes de regroupement de traces que nous venons de présenter ci-dessus. Dans la colonne « Méthode » nous citons les articles respectifs des méthodes, décrites ci-dessus, de regroupement de traces ; la colonne « Approche » indique la catégorie des travaux à laquelle la méthode appartient ; la colonne « Regroupement » indique la stratégie utilisée par les auteurs pour le traitement de traces avant d'appliquer l'algorithme de *clustering* choisi ; la colonne « *Clustering* » fait référence à la technique de *clustering*[4] sous-jacente utilisée, qui est généralement un algorithme de regroupement de données.

5.2.5 Bilan de l'état de l'art

Bien que les approches existantes offrent des résultats prometteurs (Zandkarimi et al., 2020), chacune d'entre elles présente des inconvénients. Elles ont notamment une complexité de calcul élevée et ne produisent pas forcement des modèles de haute qualité pour tous les domaines étudiés. Pour notre cas d'étude, par exemple, nous avons montré dans le chapitre 2 que les traces issues des bibliothèques numériques constituent un défi particulier vu qu'elles sont à la fois hétérogènes et non structurées. Cela est du principalement à la diversité des utilisateurs qui, un bon

4. K-clusters n'indique pas un algorithme spécifique, mais plutôt le nombre de clusters k qui est défini au préalable par l'algorithme de clustering.

Méthode	**Type d'approches**	**Regroupement traces**	**Clustering**
Bose and Van der Aalst (2009 a)	*Trace-based*	Distance d'édition	Hiérarchique
Di Francescomarino et al. (2016)	*Trace-based*	Distance d'édition	*DBSCAN*
Chatain et al. (2017)	*Trace-based*	Distance de *Levenshtein*	*Closeness-centroids*
Song et al. (2008)	*Feature-based*	Caractéristiques fréquentes	Multiples algorithmes
Ceravolo et al. (2017)	*Feature-based*	Caractéristiques fréquentes	Multiples algorithmes
Bose and van der Aalst (2009 b)	*Feature-based*	n-grammes	Hiérarchique
Veiga and Ferreira (2009)	*Model-based*	Les chaînes de *Marcov*	Hiérarchique
De Weerdt et al. (2013)	*Model-based*	Apprentissage actif	*k-clusters*
Hompes et al. (2015)	*Hybrid clustering*	Distance *Cosinus*	*Markov algorithm*
De Koninck and De Weerdt (2019)	*Hybrid clustering*	Caractéristiques fréquentes	*k-means* et Apprentissage actif

TABLE 5.1 – Les différentes méthodes pour chaque type d'approches de regroupement de traces

nombre parmi eux, ne suivent pas le modèle préalablement défini par les concepteurs.

Les approches *Feature-based* permettent aux chercheurs d'appliquer des algorithmes liés à la fouille des données et des différents concepts appliqués sur les événements existants dans les traces. De plus, la conversion des traces en des vecteurs en tenant compte des caractéristiques fréquentes conduit à des résultats encourageants (Song et al., 2008; Zandkarimi et al., 2020). Cependant, l'utilisation d'un espace vectoriel avec trop de caractéristiques entraîne une charge de calcul importante qui se traduit par un temps d'exécution extrêmement long. Par exemple, pour n activités, on peut avoir un nombre n^2 de relations directement suivies et n^3 si nous considérons une séquence de trois activités. Avec des milliers d'activités distinctes, la prise en compte de l'ensemble de l'espace des caractéristiques peut s'avérer coûteuse en terme de calcul. Les approches basées sur la syntaxe (*Trace-based*) sont également coûteuses en terme de temps de calcul. La mesure de la distance entre deux traces avec une simple approche basée sur la syntaxe comme

la distance d'édition générale prend un temps quadratique. En outre, selon des études qui ont incorporé des connaissances d'experts dans le regroupement (Lu et al., 2019; De Koninck et al., 2017), les deux types d'approches précédentes souffrent d'un autre inconvénient du regroupement, à savoir le regroupement non pertinent des traces. Cela est dû au fait que deux traces peuvent être similaires selon les mesures de distance, mais représenter des comportements de recherche ou de navigation différents.

Pour les approches *Model-based*, les traces sont regroupées si elles améliorent ensemble la qualité des modèles. Cependant, leur principal inconvénient est leur grande complexité de calcul. La qualification des traces dans l'appartenance à un groupe est réalisée par l'évaluation de la qualité du modèle de processus qui est une tâche coûteuse en termes de calcul. L'évaluation dépend également de nombreux facteurs, notamment l'algorithme de découverte de processus utilisé, les métriques d'évaluation du modèle et le type d'évaluation lui-même (basée des algorithmes d'alignement ou autres) (Van der Aalst, 2016). Les mêmes problèmes peuvent être diagnostiqués avec les approches *Hybrid-based* mais toujours une complexité de calcul meilleure que les approches *Model-based*.

Dans le cadre de cette thèse, nous considérons que notre nouvelle méthode de regroupement des traces appartient aux approches *Feature-based*, nous convertissons également les traces en vecteurs. Nous appliquons, ensuite, une métrique de similarité pour calculer la distance entre les traces converties pour finalement attribuer le groupe approprié à chaque trace (cf. figure 5.4). Notre choix est motivé par la richesse des traces issues de *Gallica* en termes de caractéristiques. Ces dernières peuvent contribuer fortement à générer des modèles pertinents. Cependant, pour faire face à la complexité de cette approche, nous allons simplifier la représentation des traces qui conduit à accélérer leur traitement.

En effet, notre méthode devrait être capable de gérer le nombre exponentiel d'événements dans les traces d'exécution issues des bibliothèques numériques, d'une part, et d'extraire des caractéristiques plus pertinentes pour le regroupement efficace des traces, d'autre part. Nous comparons notre méthode à deux autres inspirées de l'état de l'art, toutes les deux représentent les traces à l'aide de leurs caractéristiques fréquentes (le profilage de trace). Ces caractéristiques sont la fréquence des événements et les successions directes entre les événements dans l'ensemble des traces. Nous montrerons que ces facteurs ne sont pas suffisants pour encoder les traces et ensuite les regrouper. D'autres paramètres, principalement les sous-séquences fréquentes dans les traces, peuvent en outre contribuer à distinguer les traces et par conséquent les utilisateurs et leurs objectifs. De plus, la conver-

sion des traces en vecteurs et le regroupement des événements avec l'extraction et l'encodage des sous-séquences fréquentes, pourrait alléger de manière significative la complexité des traces et le grand nombre d'événements, et finalement réduire significativement le temps de calcul.

5.3 Méthode de regroupement proposée

Afin de rechercher des similitudes entre les différentes traces dans les portails des bibliothèques numériques, nous proposons une classification non supervisée qui consiste à regrouper les traces sur la base des sous-séquences fréquentes. Nous décrivons donc dans cette section notre méthode et les différents caractéristiques mises en jeu pour réaliser un regroupement performant.

Notre méthode, qui adopte l'approche *Feature-based*, se déroule en trois phases :

- **La préparation des traces** : elle consiste à convertir les traces en vecteurs de caractéristiques (cf. sous-section 5.3.1).
- **La similarité entre les traces** : elle permet de mesurer la distance entre les vecteurs de traces (cf. sous-section 5.3.2).
- **Le regroupement des traces** : elle se base sur un algorithme de clustering pour rassembler les traces similaires (cf. sous-section 5.3.3).

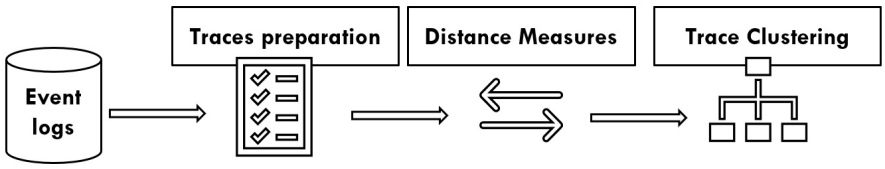


FIGURE 5.4 – Aperçu sur le processus général des approches *Feature-based*

La figure 5.4 illustre le pipeline de notre méthode. Nous détaillons toutes les étapes dans les sous-sections suivantes.

5.3.1 Préparation des traces

Afin de convertir nos traces sous forme de vecteurs de caractéristiques, nous nous sommes basé sur l'analyse des séquences (*Sequential Pattern Mining*). L'analyse des séquences est fréquemment utilisée par la communauté de la fouille de

processus dans la phase de nettoyage des journaux d'événements (Hassani et al., 2019; Sani et al., 2018).

L'idée fondamentale de notre stratégie de regroupement se base sur l'hypothèse suivante : une trace d'exécution contenant une sous-séquence fréquente est une trace significative. Les événements fréquents et les sous-séquences fréquentes ont été utilisés dans les travaux de Lu et al. (2019) pour améliorer le regroupement des traces dans le domaine médical. Ces traces sont définies manuellement par des experts pour simuler des parcours de patients dans un système d'information d'une clinique. À l'aide des échantillons, Lu et al. (2019) ont proposé d'extraire les événements ainsi que les sous-séquences fréquentes qui ont été ensuite utilisées comme critères comportementaux pour déterminer les groupes de traces correspondants (Lu et al., 2019).

Dans le cadre des bibliothèques numériques, les traces des utilisateurs ayant des connaissances avancées ne doivent pas être mélangées avec les traces d'utilisateurs non familiarisés avec les bibliothèques numériques. Nous supposons que les sous-séquences fréquentes d'événements sont pertinentes pour faire cette distinction (cf. Chapitre 2). L'utilisation de la même séquence d'événements pour une tâche particulière indique que cette séquence satisfait l'utilisateur pour atteindre son objectif. Les autres traces sans sous-séquences fréquentes impliquent généralement que leurs utilisateurs sont toujours à la recherche de leurs requêtes pour atteindre leurs objectifs. Nous pensons que l'exécution fréquente d'une sous-séquence par un utilisateur indique que cette sous-séquence satisfait son exigence d'atteindre une tâche particulière. De plus, les retours en arrière dans les traces accentuent généralement des hésitations. Les utilisateurs novices des bibliothèques numériques utilisent fréquemment les retours en arrière, tandis que les utilisateurs avancés ont tendance à suivre des cheminements sans retour en arrière. Notre méthode devrait donc encoder les sous-séquences fréquentes, les retours en arrière et tout autre critère qui peut être déterminant dans la modélisation des parcours utilisateurs et proposer un ensemble de recommandations pour réaliser une tâche spécifique pour un utilisateur particulier.

Concrètement, une sous séquence fréquente $FSS = < e_1, ..., e_n >$ contient un ensemble fini d'événements de longueur n ($n > 1$) où leurs événements sont exécutés dans l'ordre au moins deux fois. Dans le cadre de cette thèse, nous avons utilisé l'algorithme *Prefixspan*[5] (Pel et al., 2001), pour l'extraction des motifs séquentiels FSS des traces d'événements. Pour ce faire, nous avons donc commencé par convertir les journaux d'événements originaux R (cf. tableau 5.3) en un en-

5. `https://pypi.org/project/prefixspan/`

semble de traces L. Ces derniers regroupent les traces d'événements dans l'ordre d'exécution temporelle selon l'identifiant unique *CaseId*. Par exemple, la trace avec le *CaseId* 1 dans la table 5.3 correspond à <home-index, home-languages, home-languages-selection>. *PrefixSpan* découvre tous les motifs séquentiels fréquents qui apparaissent dans une base de données de séquences (l'ensemble de traces L dans notre cas). Il extrait les sous-séquences qui apparaissent dans les séquences de la base de données ayant un support minimal. Le support d'un motif séquentiel est le nombre de séquences où le motif apparaît divisé par le nombre total de séquences dans la base de données (Pel et al., 2001). Par exemple, si on considère un journal d'événements L avec quatre traces : $db = [t_1 = < e_1, e_2, e_4 >, t_2 = < e_3, e_2, e_1 >, t_3 = < e_2, e_1, e_4 >$ et $t_4 = < e_3, e_4 >]$.

L'algorithme *PrefixSpan* considère chaque trace comme une séquence de données textuelles. Par la suite, l'application de l'algorithme PrefixSpan(db) sur l'ensemble des traces avec un support égale à zéro par exemple (toute sorte de sous-séquences fréquentes) produit les sous-séquences fréquentes illustrées dans la table 5.2.

Fréquence	**Sous séquence fréquente**
3	e_1
1	e_1, e_2
1	e_1, e_2, e_4
2	e_1, e_4
3	e_2
2	e_2, e_4
2	e_2, e_1
1	e_2, e_1, e_4
3	e_4
2	e_3
1	e_3, e_2
1	e_3, e_2, e_1
1	e_3, e_1
1	e_3, e_4

TABLE 5.2 – Sous-séquences fréquentes générées par l'algorithme *PrefixSpan*

Dans ce travail, nous considérons une sous-séquence fréquente, chaque partie de la trace contenant au moins deux événements consécutifs. La fréquence d'une sous-séquence fréquente f_{FSS} est le nombre d'occurrences dans l'ensemble des traces d'événements de la séquence d'événements (dans l'ordre). Les sous-séquences fré-

quentes extraites sont triées d'abord en fonction de leurs longueurs et, ensuite, en fonction de leurs fréquences f_{FSS} lorsque deux sous-séquences fréquentes ont la même longueur.

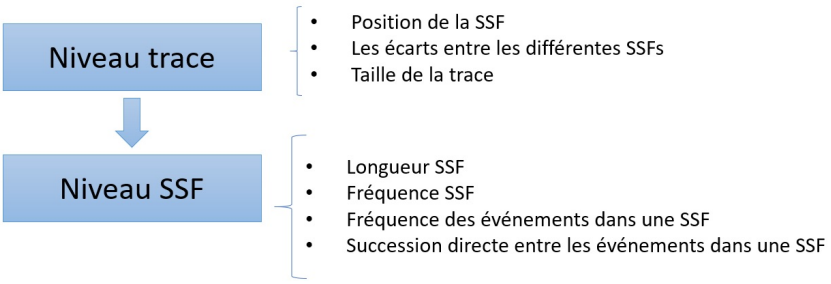


FIGURE 5.5 – Les niveaux de l'encodage

Dans cette thèse, comme le montre la figure 5.5, nous considérons que l'encodage proposé a été établie sur deux niveaux : au **niveau de la trace** et bien évidement au **niveau la sous séquence fréquente** elle même. Au niveau des traces, nous convertissons les traces en appliquant un encodage particulier sur les sous-séquences fréquentes. Chaque sous-séquence fréquente identifiée dans une trace est remplacée par son encodage. Tous les autres événements qui n'appartiennent pas à une sous-séquence fréquente sont considérés comme non pertinents et seules leurs positions sont prises en compte dans le regroupement. Pour cette raison, nous remplaçons ces événements par 1 dans les traces. De cette façon, nous distinguons aussi les traces avec les même sous-séquences fréquentes mais pas dans les même positions et nous obtiendrons un ensemble de vecteurs de tailles variables.

Par exemple, les deux traces $t_1 = < e_1, e_2, e_3, e_4 >$ et $t_2 = < e_0, e_1, e_2, e_3 >$ sont représentées respectivement par les deux vecteurs différents $[FSS_1\ 1]$ et $[1\ FSS_1]$, où FSS_1 est l'encodage de la sous séquence fréquente $< e_1, e_2, e_3 >$. En outre, les événements e_0 et e_4 ne sont pas considérés dans le regroupement. Par ailleurs, le fait de remplacer tous les événements qui ne sont pas dans la sous-séquence fréquente par 1 permet de conserver l'information sur la position de la sous-séquence fréquente, les écarts entre les différentes sous-séquences fréquentes ainsi que la taille de la trace. En plus des sous-séquences fréquentes et leurs positions dans la trace, d'autres facteurs peuvent être impliqués pour profiler les utilisations typiques des bibliothèques numériques.

Au niveau des sous-séquences fréquentes, l'encodage que nous proposons est basé sur quatre caractéristiques qui peuvent enrichir la représentation vectorielle

de nos traces :

- **La longueur d'une sous-séquence fréquente** : c'est le nombre d'événements dans une sous-séquence fréquente. Nous supposons qu'une sous-séquence fréquente de longueur m et de fréquence f_{FSS} est plus importante qu'une sous-séquence fréquente de longueur n et d'une même fréquence f_{FSS} si $m > n$. Plus la longueur est élevée, plus la sous-séquence fréquente est significative.

- **La fréquence d'une sous-séquence fréquente** : est le nombre de fois où la sous-séquence fréquente apparaît dans l'ensemble des journaux d'événements. La sous-séquence fréquente ayant la fréquence f_{FSS} la plus élevée est la sous-séquence fréquente la plus importante.

- **La fréquence des événements dans une sous-séquence fréquente** : ce facteur a été inspiré de Song et al. (2008). La fréquence de chaque événement est le nombre d'occurrences total de cet évènement dans l'ensemble des traces. La différence entre deux sous-séquence fréquentes ayant les mêmes fréquence et longueur est soulignée par la fréquence de leurs événements.

- **La succession directe entre les événements dans une sous-séquence fréquente** : ce facteur a également été inspiré de Song et al. (2008). Notre encodage prend en compte la fréquence des relations directes entre les événements dans une sous-séquence fréquente. Il s'agit du nombre total d'occurrences des successions directes entre chaque paire d'événements dans l'ensemble des traces. Les relations redondantes entre deux événements dans une sous-séquence fréquente peuvent indiquer des critères importants pour le regroupement tels que les *backtracks* ou l'utilisation d'événements pivots (par exemple, la consultation de la page d'accueil de la bibliothèque numérique) qui peuvent déclencher de nombreux autres événements.

Afin d'encoder les sous-séquences fréquentes extraites à partir des traces, nous proposons une nouvelle formule qui prend en compte toutes les caractéristiques mentionnées ci-dessus :

$$Encoding(FSS) = \frac{1}{f_{FSS} \sum_{i=1}^{n-1} f_{e_i} f_{e_{i+1}} f_{r_{i,i+1}}} \tag{5.1}$$

Où, f_{FSS} est la fréquence de la sous-séquence fréquente extraite, n est sa longueur (nombre d'événements dans la sous séquence fréquente), f_{e_i} est la fréquence de l'événement e_i et $f_{r_{i,i+1}}$ est la fréquence de la relation directe entre tous les événements consécutifs de la sous-séquence fréquente dans les journaux d'événements. La valeur de l'encodage de la sous-séquence fréquente est comprise entre 0 et 1 : une valeur d'encodage proche de 0 indique que la sous-séquence fréquente est importante dans l'ensemble des journaux d'événements. Tandis qu'une valeur proche de 1 implique une sous-séquence peu informative, ce qui est en totale cohérence avec la substitution des évènements trous [6] par 1.

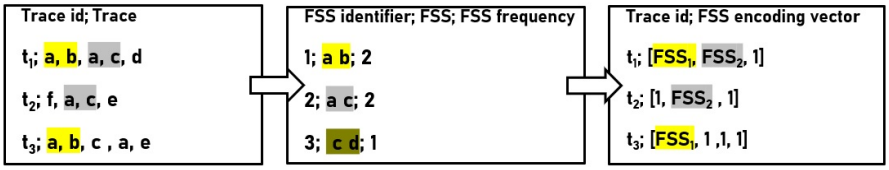


FIGURE 5.6 – Application de l'algorithme d'encodage 2 à un échantillon de traces : l'échantillon de traces (à gauche), les sous-séquences fréquentes extraites (au centre) et la représentation vectorielle des traces (à droite).

Afin de traiter le problème du bruit et les valeurs aberrantes dans les traces (Van der Aalst, 2016), celles qui ne contiennent aucune sous-séquence fréquente ont été supprimées. Ces traces qui se limitent à des séquence de 1 ne peuvent pas servir bien évidement au regroupement des traces. Cette stratégie de nettoyage de traces a été aussi utilisées dans les travaux de Sani et al. (2018) et Hassani et al. (2019). Sani et al. (2018), en effet, a défini un seuil sous lequel les traces n'ayant pas des séquences fréquentes sont supprimées. Selon les auteurs les traces supprimées sont des traces qui ne possèdent pas des relations importantes entre leurs événements telles que le parallélisme et la concurrence. L'algorithme 2 illustre les différentes étapes que nous avons implémentées pour l'encodage des sous-séquences fréquentes.

Suite à l'encodage des sous-séquences fréquentes, toutes les traces dans les journaux d'événements sont converties en des vecteurs (cf. figure 5.6). Ces vecteurs sont ensuite regroupés en fonction de leurs similarités en utilisant des opérations vectorielles. Dans la partie 5.3.2, nous décrivons les mesures de similarité utilisées dans le but de regrouper les vecteurs de traces similaires.

6. Ce sont les événements qui n'appartiennent pas à une sous-séquence fréquente

Algorithme 2 : Algorithme d'encodage des sous-séquences fréquentes (*FSS Encoding algorithm*)

Input : Sequence of traces $\boldsymbol{L}$, Frequent sub-sequences $\boldsymbol{FSS}$, List of frequent sub-sequences length $\boldsymbol{FSS_{Length}}$, List of frequent sub-sequences count $\boldsymbol{FSS_{Count}}$, Events identifiers $\boldsymbol{Event_{Id}}$, Events frequencies $\boldsymbol{Event_{Freq}}$, Matrix of direct succession between events $\boldsymbol{M}$

Output : $\boldsymbol{L_{withFSSencoding}}$

Function `FSSEncoding`(FSS, FSS_{Length}, FSS_{Count}, $Event_{Id}$, $Event_{Freq}$, M) **:**

- **for** *each line* l *in* L **do**
  - **for** *each item* fss *in* FSS *and* $(FSS_{Length}(fss) > 1)$ **do**
    - **if** $fss \in l$ **then**
      - $freq_{fss} = FSS_{Count}(fss)$// frequency of the fss
      - $events = FSS(fss).split()$ // browse events in the fss
      - **for** *each item* i *in* $len(events) - 1$ **do**
        - $e1 = events[i]$
        - $e2 = events[i + 1]$
        - $id1 = Event_{Id}[e1]$
        - $id2 = Event_{Id}[e2]$
        - $f1 = Event_{Freq}[e1]$
        - $f2 = Event_{Freq}[e2]$
        - $r = M[id1][id2]$
        - $sum+ = f1 * f2 * r$
      - **end**
      - $Enco_{fss} = 1/(int(freq_{fss}) * sum)$
      - $ligne = ligne.replace(FSS(fss), str(Enco_{fss}))$
    - **end**
  - **end**
- **end**
- **return** $L_{withFSSencoding}$;

End Function

5.3.2 Similarité entre les traces

Afin d'appliquer des mesures de similarités entre les vecteurs de traces générés par l'algorithme 2, nous avons défini une dimension fixe de l'espace vectoriel. Cette dimension correspond à la taille n de la plus grande trace vectorielle. Pour cela, nous avons ajouté des zéros à la fin des vecteurs de taille inférieur à n. Nous pensons que l'ajout de zéros à la fin des vecteurs ne modifie pas les traces originales à partir desquelles les vecteurs sont calculés. Un zéro peut faire référence à un événement neutre qui correspond à $\#rien\_faire$ à la fin d'une session. Une trace avec seulement deux événements peut être représentée par ces deux événements et une séquence de $\#rien\_faire$ à la fin. Une fois les vecteurs sont normalisés et de taille fixe, nous utilisons les mesures *Jaccard* et *Cosinus*[7] pour calculer les distances entre les vecteurs. Ces mesures ont été recommandées par la communauté de la fouille des processus pour les méthodes *Feature-based*.

Étant donné un ensemble de séquences de traces, la distance entre une paire de traces nous aide à trouver leur similarité (similarité $= 1 - distance$) et à dériver une relation structurelle entre elles.

5.3.2.1 La similarité *Cosinus*

Après avoir converti les traces en leurs représentations correspondantes, leurs similarités peuvent être calculées efficacement. La mesure *Cosinus* calcule la similarité entre deux vecteurs à n dimensions en déterminant le cosinus de leur angle. Ainsi, nous calculons la similarité entre chaque vecteur de trace et toutes les autres traces en utilisant la formule suivante :

$$cosine\_sim(\overrightarrow{A}, \overrightarrow{B}) = \frac{\overrightarrow{A} \times \overrightarrow{B}}{||\overrightarrow{A}|| \times ||\overrightarrow{B}||} \tag{5.2}$$

où $\overrightarrow{A}$ et $\overrightarrow{B}$ sont respectivement les représentations vectorielles des traces.

5.3.2.2 L'indice de*Jaccard*

L'indice de *Jaccard* est une statistique utilisée pour comprendre les similitudes entre des ensembles d'échantillons. Pour deux ensembles A et B, la représentation

7. `https://docs.scipy.org/doc/scipy/reference/spatial.distance.html`

mathématique de l'indice s'écrit comme suit :

$$Jaccard\_Index(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{5.3}$$

où A et B sont respectivement les représentations vectorielles des traces.

Dans le cadre de cette thèse, nous utilisons pour nos expérimentions les deux mesures à la fois, nous comparons les résultats et nous reportons les meilleurs résultats obtenus par l'une des mesures. Suite aux calculs des distances entre les traces, une matrice de distance est produite. Les vecteurs de traces ainsi que les matrices de similarité seront l'entrée pour les algorithmes de regroupement que nous allons décrire dans la partie 5.3.3.

5.3.3 Regroupement des traces

Dans le cadre de cette thèse, également, nous avons fait le choix d'utiliser deux algorithmes de clustering : *Density-Based Spatial Clustering of Applications with Noise* connu par *DBSCAN* (Ester et al., 1996) et *Meanshift* (Comaniciu and Meer, 2002). Les deux algorithmes ne nécessitent pas de spécifier au préalable le nombre de *clusters* ce qui est en compatibilité avec notre cas. Nous ne connaissons pas préalablement le nombre de clusters pour regrouper les traces des systèmes d'informations.

Le premier algorithme de clustering que nous avons utilisé est ***DBSCAN*** (Ester et al., 1996). Il prend en entrée deux paramètres, n_{min} qui est le nombre minimum de voisins et *eps* qui correspond à la distance *epsilon* ou l'epsilon-voisinage entres les points à regrouper. *DBSCAN* itère sur les points du jeu de données. Pour les points analysés, il construit l'ensemble des points atteignables par densité à partir de ce point : il calcule l'epsilon-voisinage *eps* de ce point, puis, si ce voisinage contient plus de n_{min} points, il poursuit avec les epsilon-voisinages de chacun d'entre eux, et ainsi de suite, jusqu'à ce qu'on ne puisse plus étendre le *cluster*. Si le point n'a pas assez de voisins, alors il sera étiqueté comme bruit. Cela permet à *DBSCAN* d'être robuste aux points aberrants puisque son mécanisme les isole.

Quant à l'algorithme **Meanshift**, il commence par l'affectation des points de données aux *clusters* d'une manière itérative en déplaçant les points vers la valeur dominante (Comaniciu and Meer, 2002). Cette dernière est la valeur la plus représentée d'une variable quelconque dans une population donnée (dans la région, dans le contexte du *Meanshift*). Plus concrêtement, étant donné un ensemble de

points de données, l'algorithme assigne d'une manière itérative chaque point de données vers le centroïde de *cluster* le plus proche. La direction vers le centroïde de *cluster* le plus proche est déterminé par l'endroit où se trouvent la plupart des points proches. Ainsi, à chaque itération, chaque trace se rapprochera de l'endroit où se trouvent la plupart des traces similaires. Lorsque l'algorithme s'arrête, chaque trace est affectée à un *cluster*.

Comme le montre le pseudo code 3, suite à l'application des algorithmes de *clustering*, nous obtiendrons un ensemble de groupes de traces enregistrés dans des fichiers *CSV*. Les fichiers obtenus suivent le format original des traces (des journaux d'événements dans la table 5.3). Les groupes de traces obtenus permettent de distinguer les types d'utilisateurs des bibliothèques numériques ainsi que leurs tâches de recherche d'informations et seront ensuite modélisés séparément à l'aide des algorithmes de fouille des processus.

Algorithme 3 : Pseudo code de la méthode de regroupement des traces basé sur le *FSS Encoding*

Data : Original logs file $\boldsymbol{R}$
Result : Logs Files $\boldsymbol{F}$ generated according the found clusters
begin
Convert the original event logs $\boldsymbol{R}$ to sequence of traces $\boldsymbol{L}$;
From $\boldsymbol{L}$, find frequent sub-sequences $\boldsymbol{FSS}$ based on defined features;
For each element in $\boldsymbol{L}$, find the most frequent $\boldsymbol{FSS}$ and replace found $\boldsymbol{FSS}$ by their encoding;
Remove elements in $\boldsymbol{L}$ where no $\boldsymbol{FSS}$ found;
For each element in $\boldsymbol{L}$, Gaps between $\boldsymbol{FSS}$ encoding will be replaced by "1";
Do distance measurement between trace vectors $\boldsymbol{L}$;
Do clustering of $\boldsymbol{L}$;
Generate Logs Files $\boldsymbol{F}$ according to the found clusters;
Return $\boldsymbol{F}$;

5.3.4 Modélisation des traces

À ce stade, nous avons obtenus plusieurs clusters de traces. Nous appliquons à ces clusters l'*Inductive Miner* comme algorithme de découverte de processus (Leemans et al., 2013) afin de découvrir des modèles de processus. Suite à une comparaison que nous avons effectuée des différents algorithmes de découverte de processus

dans le chapitre 3, l'algorithme *Inductive Miner* (cf. section 3.5.4) a montré une capacité de traiter les traces des bibliothèques numériques et à surpasser toutes les autres techniques (Trabelsi et al., 2019). En plus, cet algorithme a la particularité de produire des modèles de comportement sans anomalies à partir des traces complexes, bruitées et non complètes, ce qui nous permet de découvrir les modèles et de vérifier la conformité des modèles découverts par rapport des logs de départ.

5.4 Validation de la méthode proposée

Notre évaluation est menée à deux niveaux : **le niveau de regroupement des traces** et **le niveau de qualité des modèles de processus** (cf. figure 5.7). Pour le niveau de regroupement, un ensemble de données étiquetées assignant une classe à chaque trace d'utilisateur est nécessaire. Malheureusement, de telles données ne sont pas ouvertement disponibles. Nous avons donc simulé cette ressource. Ces données ont été également utilisées pour évaluer la qualité des modèles de processus. Pour des raisons de comparaison, nous définissons deux méthodes de regroupement inspirées de l'approche *Feature-based* (cf. partie 5.4.3). Les deux méthodes sont respectivement basées sur la fréquence des événements dans les journaux d'événements (***freq-based***) et sur la succession directe entre les événements (***relation-based***). Désormais, nous désignons notre méthode basée sur l'encodage des sous-séquences fréquentes par ***FSS-based***.

Dans cette section, nous décrivons d'abord le jeu de données simulé. Nous présentons ensuite les métriques d'évaluation et nous présentons enfin les résultats obtenus pour les deux niveaux d'évaluation. Nous tenons à préciser que, concernant l'évaluation des modèles de processus, la découverte des modèles de processus ainsi que l'évaluation des modèles découverts à l'aide des mesures d'évaluation de la qualité des modèles découverts a été réalisé avec l'outil *PROM* (Van Dongen et al., 2005).

5.4.1 Données simulées

Nous avons simulé les comportements de recherche des utilisateurs dans une bibliothèque numérique [8] en reproduisant les caractéristiques des principales catégories décrites par Marchionini (2006). Cette stratégie expérimentale est simplifiée

8. basée sur `https://projectblacklight.org/`

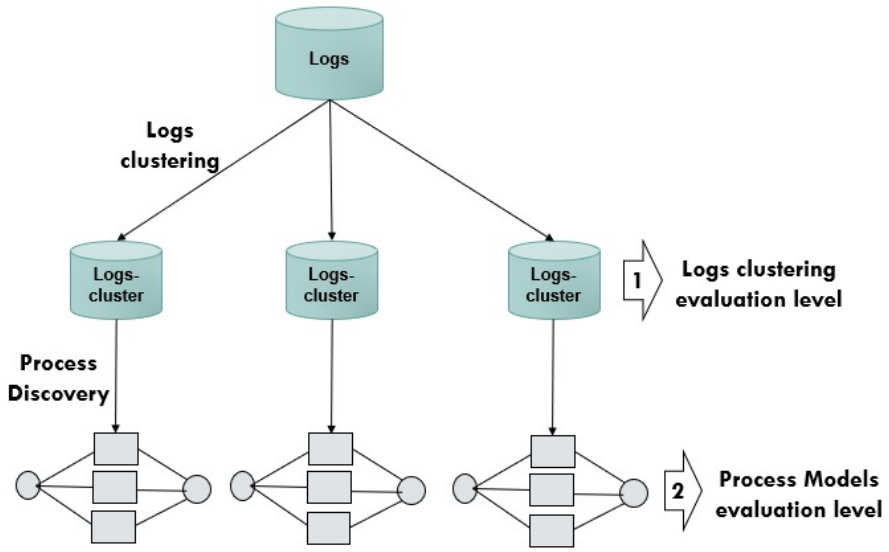


FIGURE 5.7 – Les niveaux d'évaluation de la méthode *FSS encoding*

par rapport à l'utilisation de données réelles et se limite à tester la validité de l'approche et la pertinence des métriques que nous utilisons. Notre objectif principal est d'évaluer la capacité de l'approche proposée à modéliser les comportements des utilisateurs.

L'ensemble de données simulées définit trois classes de traces :

- **Des traces *Lookup*** où les utilisateurs peuvent accéder à des documents précisément identifiés avec peu de manipulations en utilisant le moteur de recherche ou en parcourant les différentes catégories de documents (tâches de réponse à des questions ou de navigation) ;

- **Des traces *Borderline*** où les utilisateurs peuvent accéder à des documents dans un domaine bien défini, mais utilisent différentes méthodes de recherche et de filtrage des résultats (tâches d'acquisition de connaissances ou de compréhension) ;

- **Des traces *Exploratory*** où les utilisateurs peuvent accéder à un large éventail de documents dans différents domaines et de différents types, ils utilisent des fonctions de recherche et de filtrage plus avancées (tâches de planification, d'évaluation ou d'analyse) (Marchionini, 2006), (Suire et al., 2016).

CaseId	User	Date	Activity label
1	$user_1$	2016-01-12T10 :34 :25	home index
2	$user_2$	2016-01-12T10 :36 :25	home index
1	$user_1$	2016-01-12T10 :34 :26	home languages
1	$user_1$	2016-01-12T10 :34 :28	language selection
3	$user_3$	2016-01-12T10 :36 :26	home index
3	$user_3$	2016-01-12T10 :36 :27	catalog show

TABLE 5.3 – Exemple de traces simulées dans la bibliothèque numérique

Quantitativement, l'ensemble de données simulées contient 100 traces réparties en 40 traces de type *Lookup*, 30 traces de type *Borderline* et 30 traces de type *Exploratory*. Chaque trace est liée à un utilisateur. Évidemment, les traces *Exploratory* sont les traces les plus complexes et qui couvrent plus de 10 types d'événements pour un total de 310 occurrences d'événements. Les traces *Borderline* sont constituées de 6 types d'événements pour un total de 170 événements et les traces *Lookup* sont limitées à 5 de types d'événements pour un total de 140 occurrences d'événements.

Le modèle de processus que nous présentons dans la figure 5.8 est un graphe de succession directe entre l'ensemble des événements (*Directly Follows Graph* – DFG) [9] découvert sur l'ensemble de données simulées. L'algorithme de découverte de processus utilisé prend en compte les événements et leur fréquence dans les journaux d'événements, ainsi la fréquence de la succession directe entre l'ensemble des événements (Leemans et al., 2019). Les journaux d'événements seront mappés à un graphe de successions directes dont les sommets sont les événements et les arêtes les relations directes. La case verte représente l'événement de départ dans toutes les traces tandis que la case rouge représente l'événement final. Cependant, comme nous l'avons mentionné précédemment, il est loin d'être facile pour les concepteurs de comprendre un graphe de successions directes combinant plusieurs types de recherches. Il est toujours plus simple de découvrir chaque type dans un graphe séparé plutôt que dans un graphe complet (cf. figure 5.8).

9. Dans ce chapitre, nous utilisons le graphe de successions directes pour présenter nos modèles au lieu du réseau de *Petri* pour des raisons de simplicité.

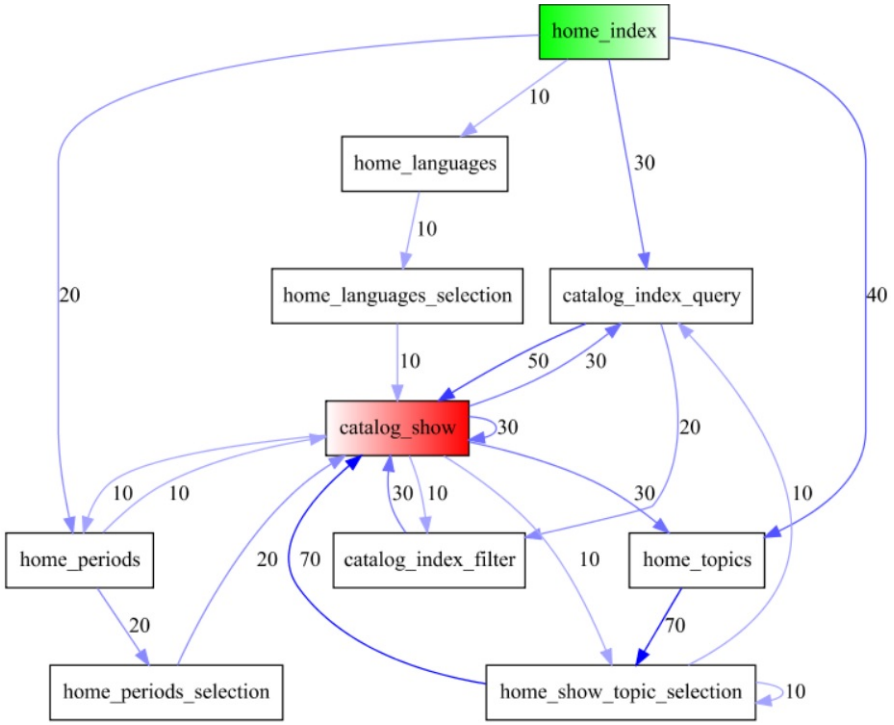


FIGURE 5.8 – Graphe de successions directes découvert pour l'ensemble des journaux d'événements simulés

5.4.2 Mesures de validation

Comme mentionnée dans la section 5.4, l'évaluation de notre méthode basée sur l'encodage des sous-séquences fréquentes est menée à deux niveaux : **le niveau de regroupement des traces** et **le niveau de qualité des modèles de processus**. Le figure 5.9, résume les différentes mesures d'évaluation que nous avons employées pour l'évaluation de notre méthode *FSS-based*.

5.4.2.1 Niveau de la qualité des groupes de traces

Comme le montre la figure 5.9, suite au regroupement des traces, la qualité des *clusters* de traces est évaluée à l'aide de deux métriques. La première métrique est le **Silhouette** score , qui décrit la distance de séparation entre les *clusters* générés. Il indique la proximité de chaque point d'un *cluster* par rapport aux points des

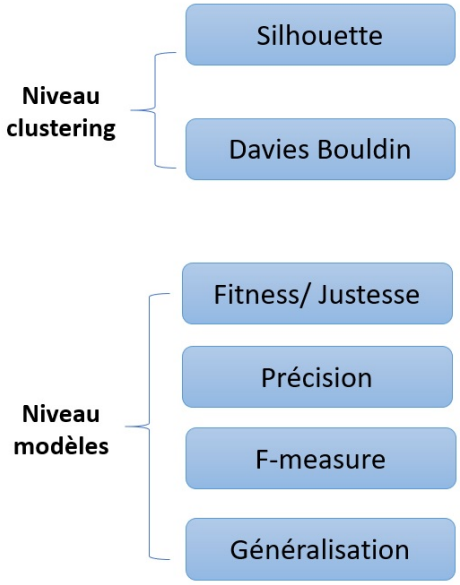


FIGURE 5.9 – Les mesures utilisées pour chaque niveau de d'évaluation de la méthode proposée

clusters voisins (cf. figure 5.10). Le Silhouette score est calculé en utilisant :

$$Silhouette(x, y) = \frac{(x - y)}{max(x, y)} \tag{5.4}$$

Où, y est la distance moyenne entre les points d'un même *cluster* et x est la distance moyenne au *cluster* le plus proche. Le Silhouette score indique que l'échantillon est très éloigné des *clusters* voisins lorsque $Silhouette =\sim 1$. Alors que $Silhouette =\sim 0$ indique que les limites entre les *clusters* voisins sont très proches. Enfin, une valeur négative de la Silhouette indique que plusieurs traces ont pu être assignées à des *clusters* erronés Rousseeuw (1987).

La deuxième métrique est l'indice **Davies-Bouldin**. Elle évalue la similarité *intra-cluster* et les différences *inter-cluster*. Cet indice traduit la « similarité » moyenne entre les *clusters*, où la similarité est une mesure qui compare la distance entre les *clusters* avec la taille des *clusters* eux-mêmes. Pour une meilleure qualité de regroupement, l'indice *Davies-Bouldin* donne une valeur plus faible [10].

10. `https://scikit-learn.org/stable/modules/clustering.html`

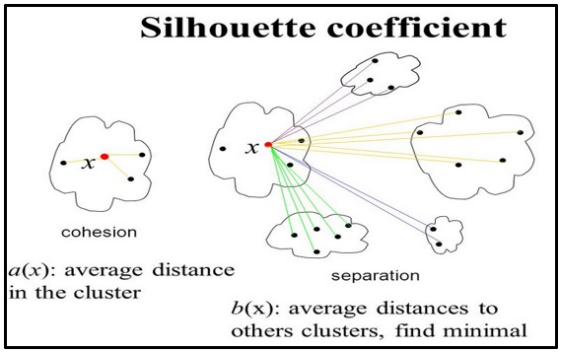


FIGURE 5.10 – Évaluation des *clusters* avec le Silhouette score (Rousseeuw, 1987)

5.4.2.2 Niveau de la qualité des modèles de processus

Afin d'évaluer la qualité des modèles de processus découverts en appliquant la méthode proposée (*FSS-based*), nous avons utilisé quatre métriques [11] très connues par la communauté de la fouille de processus (Adriansyah, 2014).

La **Fitness** qui détermine dans quelle mesure le modèle de processus couvre les journaux d'événements. Nous avons aussi utilisé la **Precision** qui correspond au taux d'activités dans les journaux d'événements par rapport au total des activités activées dans le modèle de processus. La **Generalization** a été aussi employée et elle est généralement liée aux comportements non vus. Ce critère vise à mesurer la capacité du modèle à généraliser le comportement observé dans les journaux d'événements. Un modèle de processus approprié doit trouver un équilibre entre la *Precision* et la *Generalization* (Buijs et al., 2012). Ainsi, les modèles de processus doivent aussi prendre en compte un compromis entre la *Fitness* et la *Precision*. Il est presque démontré qu'une petite quantité de comportements (journaux d'événements) conduit à une diminution de la *Fitness* et une augmentation de *Precision* Sani et al. (2019). Pour ces raisons, une **F-measure** pour les évaluations de modèles a également été proposée :

$$F - measure = \frac{2 * F * P}{F + P} \tag{5.5}$$

11. les mesures d'évaluation des modèles de processus, la *Fitness*, *Precision* et *Generalization* ont été détaillé dans le chapitre 3

5.4.3 Résultats sur les données simulées

Dans cette sous-section, nous fournissons les résultats expérimentaux sur les journaux d'événements simulés à partir d'une bibliothèque numérique. Notre objectif ici est de retrouver les trois groupes d'utilisateurs de la bibliothèque numérique décrits dans les données simulées (cf. sous-section 5.4.1).

Comme nous l'avons mentionné au début de cette section, dans le but de comparer notre méthode de regroupement avec les méthodes de travaux connexes de l'état de l'art, nous nous sommes inspirés des méthodes *Feature-based* de Song et al. (2008). Nous avons implémenté deux méthodes de regroupement des traces basées sur les caractéristiques, respectivement basées sur la fréquence des événements (***freq-based***) et la relation de succession directe entre les événements (***relation-based***). Concernant la méthode basée sur la fréquence, pendant l'étape de préparation des traces (cf. sous-section 5.3.1), chaque événement e dans les traces sera remplacé par sa fréquence f_e dans tous les journaux d'événements. Alors que, dans la méthode basée sur les relations qui consiste à compter les relations de succession directe entre les événements r dans tous les journaux d'événements (cf. section 4.5.1). Les traces seront donc remplacées par le comptage des relations de succession directe entre tous les événements. Par exemple, si on considère un journal d'événements avec deux traces $t_1 = < e_1, e_2, e_3 >$ et $t_2 = < e_1, e_2 >$; les fréquences des événements e_1, e_2 et e_3 sont respectivement 2, 2 et 1 ; l'événement e_1 est toujours suivi de e_2 tandis que e_2 est directement suivi par e_3 une fois.

Ainsi, les traces sont converties en :

- méthode basée sur la fréquence (***freq-based***)
  - $t_1 = [2, 2, 1]$
  - $t_2 = [2, 1]$
- méthode basée sur la relation (***relation-based***)
  - $t_1 = [2, 1]$
  - $t_2 = [2]$

Comme notre méthode *FSS-based*, suite à la conversion des traces en des vecteurs soit avec la méthode *freq-based* ou la méthode *relation-based*, la distance entre les vecteurs des traces a été calculée et le regroupement des traces a été effectué de la même manière que la nôtre.

5.4.3.1 Niveau de regroupement des traces

La table 5.4 présente les résultats obtenus du regroupement sur les traces simulées. Comme mentionné dans la section 5.3.3, nous utilisons deux algorithmes de *clustering* : *DBSCAN* et *Meanshift*.

		freq-based	**relation-based**	**FSS-based**
Meanshift	**Silhouette**	0,731	0,473	**0,817**
	Davies Bouldin	0,282	1,178	**0,238**
	NB clusters	5 clusters	3 clusters	**3 clusters**
DBSCAN	**Silhouette**	0,730	0,423	0
	Davies Bouldin	0,412	1,178	0
	NB clusters	2	2 (+50 noises)	1

TABLE 5.4 – Les résultats du *clustering* des traces simulées

Ces résultats montrent une meilleure performance de l'algorithme *Meanshift* par rapport à *DBSCAN*. *Meanshift* est aussi meilleur de point de vue utilisation puisque, contrairement à *DBSCAN*, il ne nécessite aucun paramètre d'entrée saisi par l'utilisateur. Les résultats montrent également que la méthode basée sur les relations (***relation-based***) et la méthode basée sur l'encodage des sous-séquences fréquentes (***FSS-based***) nous permettent d'obtenir les trois *clusters* qui sont en accord avec le nombre de clusters dans la base d'évaluation 5.4.1. Cependant, la méthode basée sur l'encodage sous-séquence fréquente offre une meilleure qualité de *clusters* que la méthode basée sur les relations. La méthode basée sur l'encodage des sous-séquences fréquentes obtient de meilleurs résultats sur les scores *Davies bouldin* et *Silhouette*.

En outre, dans le cadre de la validation du regroupement en exploitant la connaissance de la classe réelle de chaque trace, nous avons calculé les métriques de classification supervisée (Davis and Goadrich, 2006) : pour chaque classe C_i (où $1 <= i <= j$ et j est le nombre de classes réelles), le *Recall* (Rappel en français) est le taux de classes prédites correctes C_i sur le total des classes C_i dans la référence ; $R_{ci} = \frac{\sum C_i corrects}{\sum C_i references}$. La *Precision* est le nombre de classes prédites correctes C_i sur le nombre total de classes prédites C_i $P_{ci} = \frac{\sum C_i corrects}{\sum C_i predicted}$. Le *F1-score* $F1_{ci} = \frac{2 * P_{ci} * R_{ci}}{P_{ci} + R_{ci}}$ qui est la valeur harmonique de P et R et le *F1-macro* qui est la moyenne des *F1-Scores*. Nous avons également utilisé la précision totale (*Total accuracy* en anglais) qui est le nombre total de classes prédites correctement sur

le nombre total de traces.

	FSS-based			**relation-based**		
	C1	C2	C3	C1	C2	C3
Recall	1	0.33	0.66	0.52	0.3	0.66
Precision	0.58	1	1	0.42	0.3	1
F1-score	0.74	0.5	0.82	0.46	0.3	0.79
F1-macro	**0.69**			0.52		
Total accuracy	**0.75**			0.5		

TABLE 5.5 – Les résultats de la comparaison sur les données simulées

La table 5.5 montre clairement que la méthode basée sur l'encodage des sous-séquences fréquentes surpasse celle basée sur la relation. Pour chaque classe, la méthode basée sur l'encodage des sous-séquences fréquentes atteint un *F1-score* supérieur à $0,5$. Dans l'ensemble, la Précision totale et la *F1-macro* de la méthode basée sur l'encodage des sous-séquences fréquentes dépasse considérablement celle générée par la méthode basée sur les relations.

5.4.3.2 Niveau de la qualité des modèles de processus

Pour le niveau de l'évaluation des modèles de processus, nous présentons des résultats sur les modèles obtenus à partir des *clusters Meanshift*. La table 5.6 présente la qualité des modèles de processus découverts à partir des trois *clusters* des méthodes basées sur les relations et sur l'encodage des sous-séquences fréquentes, ainsi que des *clusters* d'or de la référence. La *Fitness*, la *Precision*, la *F-measure* et la *Generalization* des trois *clusters* ont été moyennées.

	Real clusters	**relation-based**	**FSS-based**
Fitness	1	1	1
Precision	0,5360	0,5965	**0,6345**
F-measure	0.7000	0.7500	**0.7800**
Generalization	0.6719	0.2697	0.4533

TABLE 5.6 – Les performances des modèles de processus découverts à partir les journaux d'événements simulés

Selon les résultats, l'approche basée sur l'encodage des sous-séquences fréquentes (*FSS-based*) améliore les valeurs de la *Precision* et de la *F-measure*.

De plus, pour la valeur de *Generalization*, les résultats montrent que les modèles obtenus en utilisant l'approche basée sur les sous-séquences fréquentes sont plus robustes en raison de la faible différence entre la *Precision* et la *Generalization* Van der Aalst (2016).

Les expériences que nous avons menées sur les données simulées montrent que notre méthode (***FSS-based***) semble efficace dans l'extraction des connaissances à partir des journaux d'événements. Les trois comportements typiques des utilisateurs correspondent parfaitement aux trois groupes identifiés. De plus, l'encodage des sous-séquences fréquentes permet de réduire fortement la taille des traces avant le *regroupement*, ce qui est essentiel pour faire face aux cas réels (traces de navigations réelles).

5.5 Conclusion

Du fait de l'utilisation croissante des bibliothèques numériques, un nombre de plus en plus élevé de traces d'exécution est généré. La fouille des processus dans ce cas conduit à des modèles complexes qui ne peuvent pas être exploités pour analyser les parcours des utilisateurs. Dans ce chapitre, nous avons exploré une nouvelle méthode afin de générer des groupes de traces disjoints pertinents pour différentes classes d'utilisateurs ou tâches de recherche d'information (*FSS-encoding*). Nous avons commencé par transformer les traces en vecteurs de caractéristiques fréquentes, puis nous avons appliqué une métrique de similarité afin de calculer la distance entre ces vecteurs et nous finissons en attribuant chaque trace au groupe approprié.

Nous avons, ensuite, utilisé des techniques de la fouille de processus pour extraire les modèles des parcours des utilisateurs. Les modèles de processus découverts sont utiles aux concepteurs des bibliothèques numériques pour répondre aux exigences des utilisateurs et de leur proposer un ensemble de recommandations. Nous avons comparé notre méthode à deux autres méthodes de regroupement, inspirées de l'état de l'art, sur un ensemble de données simulées où les utilisateurs sont classés manuellement en trois classes selon leurs tâches (objectif de la recherche) en reproduisant les caractéristiques des principales catégories décrites par *Marchionini* (Marchionini, 2006). Les résultats ont montré l'efficacité de notre méthode à la fois pour regrouper les utilisateurs et modéliser leurs parcours.

Le chapitre suivant décrit l'utilisation de notre méthode de regroupement sur les données réelles fournies par *Gallica* avant de passer à l'extraction des modèles

de parcours des utilisateurs.

Résumé exécutif

- Nous avons présenté un état de l'art sur les différentes méthodes de regroupement des traces utilisées dans le contexte de la fouille de processus.
- Nous avons présenté notre nouvelle méthode de regroupement des traces des utilisateurs des bibliothèques numériques (*FSS-encoding*) basée sur trois étapes principales : extraction des caractéristiques des traces basée sur l'identification de sous-séquences et leur fréquence ; la similarité entre les traces et le regroupement des traces.
- Nous avons évalué note méthode de regroupement des traces en la comparant avec des méthodes inspirées de l'état de l'art et nous l'avons validé à l'aide d'une simulation que nous avons effectué sur une bibliothèque numérique .

Chapitre 6

Application sur des données réelles : traces issues de Gallica

Sommaire

6.1 Introduction

Suite au regroupement des traces (cf. Chapitre 5), nous sommes passés à leur modélisation à l'aide de l'algorithme de découverte des processus *Inductive Miner*. L'objectif de ce chapitre est d'évaluer la capacité de notre méthode à identifier des parcours de recherche d'information similaires sur des données réelles issues du système d'information de la Bibliothèque Nationale de France. Comme nous l'avons mentionné dans le chapitre 4, les traces récupérées sont brutes et, suite à un pré-traitement, elles sont converties en traces modélisées exploitables par les algorithmes de la fouille de processus. Néanmoins, pour un tel cas réel l'application de notre méthode soulève un défi important. L'énorme quantité de traces issues de *Gallica* rend leurs regroupement et modélisation très coûteux en terme de temps et n'aboutit potentiellement pas à des résultats exploitables. À titre d'exemple, le nombre mensuel de traces issues de *Gallica* dépasse approximativement $1,5M$. Le regroupement d'une telle quantité de traces proposerait un nombre élevé de *clusters* ce qui est loin d'être pratique dans un monde réel et ne sont par conséquent pas exploités par les concepteurs de *Gallica*. Un échantillonage est ainsi indispensable afin de proposer une modélisation pertinente et avantageuse aux concepteurs et utilisateurs. Notre objectif est donc, en premier lieu, de mettre en place une méthodologie d'expérimentation afin de déterminer un échantillon générique de traces qui peut modéliser les comportements de différents utilisateurs de *Gallica*. En deuxième lieu, notre travail évalue l'applicabilité de notre méthode, d'une part, sa capacité à aider les concepteurs de *Gallica* à comprendre les comportements des utilisateurs et, d'une autre part, les éventuelles limites des modèles.

Tout comme dans le chapitre 5 (cf. section 5.4), nos expérimentations ont été menées sur deux niveaux, **la qualité des clusters** et **la qualité des modèles**. L'évaluation de la qualité des *clusters* de traces réelles est décrite dans la section 6.2 tandis que la qualité des modèles générés à partir de ces *clusters* est discutée dans la section 6.3.

6.2 Évaluation du regroupement

Comme nous l'avons indiqué dans le chapitre 4, nous travaillons sur les données de *Gallica* du mois d'avril 2017. Le nombre total des traces modélisées est 1 719 657, elles comportaient 102 787 467 événements (*cf.* table 4.4). Afin de mener à bien notre évaluation, nous avons extrait, à partir de ces traces, de nombreux

échantillons. Dans un premier temps, nous avons utilisé des traces de dates chronologiques pour construire plusieurs échantillons de dimension variable de 1 000 à 50 000 traces. Pour chaque échantillon, nous avons extrait des sous-échantillons avec des tailles maximales de traces allant de 1 000 à 5 000 évènements. Un sous-échantillon est également construit sans contrainte sur la longueur des traces. Dans un deuxième temps, nous avons effectué un échantillonage aléatoire de traces ayant des dates d'exécution non consécutives.

6.2.1 Traces de dates chronologiques

Le premier échantillonage respecte la chronologie de l'exécution des traces. Nous nous servons des n premières traces exécutées avec un nombre n variable de 1 000 à 50 000. L'objectif est de déterminer le nombre de traces raisonnable qui engendre des modèles génériques couvrant les différents comportements des utilisateurs. Nous avons défini cinq échantillons de dimensions respectives 1 000, 50 00, 10 000, 20 000 et 50 000. Nous nous sommes arrêté au seuil de 50 000 traces parce que les méthodes de l'état de l'art auxquelles nous nous comparons sont incapables de traiter ce nombre de traces.

La table 6.1, décrit la qualité des *clusters* obtenus à partir des trois méthodes **freq-based**, **relation-based** et notre **FSS-based** sur des différents échantillons. Afin d'évaluer le regroupement de trace, nous avons utilisé deux métriques, la *Silhouette* et le *Davies-Bouldin* score, (cf. section 5.4.2). Une meilleure qualité de regroupement est traduite par un faible *Davies Bouldin* et une *Silhouette* proche de 1. Nous indiquons également le nombre de *clusters* obtenus et le temps d'exécution du regroupement afin de juger l'ordre de réalisation du regroupement dans chaque échantillon. Le temps de calcul rapporté est la durée d'exécution de l'algorithme de regroupement à l'aide d'une machine qui dispose de 56 CPUs, 128Go de mémoire, et une carte graphique de type *NVIDIA TITAN X*.

Comme l'indique la table 6.1, hormis la méthode *FSS-based* qui a été exécutée sur tous les échantillons, les méthodes *relation-based* et *freq-based* ont échoué, pour des raisons techniques, de traiter 50 000 traces. Toutes les méthodes ont généré un nombre raisonnable de *clusters* sur tous les échantillons. Trois *clusters* ont été générés par la méthode *freq-based* sur l'échantillon de 5 000 traces alors que seulement deux sont générés avec les autres méthodes sur tous les échantillons. La table montre aussi que les trois méthodes proposent des valeurs similaires de *Silhouette* ($\sim 0,5$). En revanche, notre méthode *FSS-based* propose des scores *Davies-Bouldin* nettement plus bas, particulièrement avec un nombre de traces important. Les ré-

		1 000	5 000	10 000	20 000	50 000
freq-based	NB Clusters	2	3	2	2	–
	Silouhette	**0,564**	**0,547**	**0,547**	**0,551**	–
	Davies-bouldin	**0,495**	0,756	0.753	0.746	–
	Execution time (h-m-s)	11m32s	15h20m48s	87h51m58s	258h34m40s	–
relation-based	NB Clusters	2	2	3	2	–
	Silouhette	0,459	0,449	0,446	0,444	–
	Davies-bouldin	1,013	1,042	1,046	1,053	–
	Execution time (h-m-s)	5m47s	4h34m44s	22h18m19s	116h53m	–
FSS-based	NB Clusters	2	2	2	2	2
	Silouhette	0,467	0,461	0,459	0,575	0,459
	Davies-bouldin	0,560	**0,573**	**0,579**	**0,582**	**0,581**
	Execution time (h-m-s)	6m16s	4h10m21s	18h24m48s	77h13m41s	439h38m19s

TABLE 6.1 – Évaluation du *clustering* sur les n premières traces

sultats de la méthode *FSS-based* montrent que jusqu'à 20 000 traces, plus nous avons de données plus le regroupement est performant. En revanche, en comparant les résultats sur les échantillons de 20 000 et 50 000, nous constatons que le score de *Davies-Bouldin* reste quasi constant alors que la *Silhouette* diminue. En termes de temps d'exécution, notre méthode semble la plus efficace à manipuler un nombre important de traces. Sur 20 000 traces le temps d'exécution de la méthode *FSS-based* est presque la moitié de la durée d'exécution de la méthode *relation-based* et moins que le tiers de temps de la méthode *freq-based*.

6.2.2 Échantillonnage aléatoire

Outre que les n premières traces, nous avons échantillonné aléatoirement des traces ayant des dates d'exécution non consécutives. Les expériences précédentes ont montré que les méthodes de l'état de l'art ne sont pas capables de traiter 50 000 traces, nous avons ainsi généré deux échantillons de 20 000 et 40 000 traces d'exécution. Notre objectif ici consiste à tester la validité de notre méthode sur des traces qui ne sont pas ordonnées chronologiquement. Comme le montre la table 6.2, les résultats sur 20 000 traces sont semblables à ceux obtenus sur l'échantillon de 40 000. Le traitement est néanmoins plus rapide. Les résultats montrent également que les meilleures valeurs de *Silhouette* sont obtenues avec la méthode *freq-based*

mais avec un temps d'exécution beaucoup plus lent. Notre méthode surpasse, par contre, les autres sur le score de *Davies-bouldin* en temps plus réduit d'exécution.

		20 000	40 000
freq-based	NB Clusters	4	2
	Silouhette	**0,518**	**0,528**
	Davies-bouldin	0,889	0,865
	Execution time (h-m-s)	230h40m28s	753h43m19s
relation-based	NB Clusters	2	2
	Silouhette	0,434	0,454
	Davies-bouldin	1,081	1,029
	Execution time (h-m-s)	122h20m33s	440h36m46s
FSS-based	NB Clusters	2	2
	Silouhette	0,453	0,470
	Davies-bouldin	**0,597**	**0,570**
	Execution time (h-m-s)	71h30m23s	258h46m20s

TABLE 6.2 – Évaluation du regroupement sur les échantillons aléatoires de traces

6.3 Évaluation de la modélisation

Suite au regroupement des traces et la génération des *clusters*, nous avons procédé à l'évaluation des modèles de processus découverts sur la base des *clusters* obtenus. Notre objectif est d'observer qualitativement et quantitativement l'impact de notre méthode de regroupement dans la modélisation des données réelles.

À l'instar de notre traitement des données synthétiques (cf. section 5.4), nous nous sommes servi de l'algorithme *Inductive Miner* (Leemans et al., 2013) pour découvrir les modèles des traces réelles et de l'outil *PROM* dans l'évaluation. La modélisation est réalisée sur les *clusters* issus du regroupement des échantillons décrits dans la section précédente. Nous commençons par l'évaluation des traces de dates chronologiques puis les traces échantillonnées aléatoirement. La métrique de conformité *F-measure*, qui représente le compromis entre la *Fitness* et la *Precision* a été utilisée pour évaluer chaque modèle découvert sur chaque échantillon.

6.3.1 Traces de dates chronologiques

Nous avons généré des modèles à partir de tous les *clusters* obtenus sur les échantillons de traces chronologiques de dimension 1 000, 5 000, 10 000, 20 000 par les trois méthodes de regroupement et 50 000 uniquement par notre méthode *FSS-based*. La figure 6.1 décrit l'évolution des *F-measures* de chaque méthode en fonction de la taille des données (expérience réalisée sur les n premières traces). Les scores indiqués correspondent à la moyenne des scores des modèles obtenus pour l'ensemble des *clusters*.

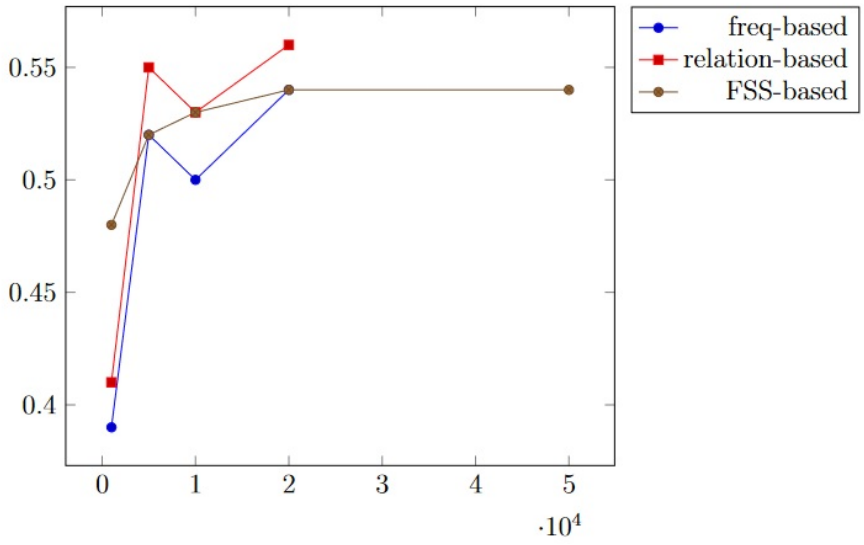


FIGURE 6.1 – Courbes des *F-measures* en fonction de la dimension des échantillons de traces chronologiques

La figure 6.1 montre que, quelle que soit la méthode, le comportement des *F-measures* sur les différents ensembles de données est similaire. En effet, toutes les méthodes génèrent des *F-measures* entre 0.5 et 0.55 sur des échantillons de plus que 5 000 traces. Seule, la méthode *FSS-based* est la plus performante sur peu de données (la *F-measure* varie entre 0.48 et 0.49 sur moins de 5 000 traces). En outre, les courbes issues des méthodes *freq-based* et *relation-based* contiennent des oscillations ce qui complique la définition d'une dimension générique. En revanche, pour la méthode *FSS-based* le score est proportionnel à la taille de données et a tendance de converger à partir de 20 000 traces. Pour les trois méthodes, les meilleures *F-measures* ont été accomplies sur l'échantillon de 20 000 traces.

Les résultats obtenus pour l'évaluation du *clustering* ne reflètent parfois pas la qualité des modèles de processus découverts sur la base des groupes de traces trouvés. L'expérimentation sur l'échantillon de 1 000 traces avec la méthode *freq-based*, par exemple, a montré des bonnes performances de regroupement cependant ses modèles découverts ne sont pas satisfaisants. La figure 6.2 illustre les deux modèles découverts sur les deux *clusters* générés par la méthode *freq-based*.

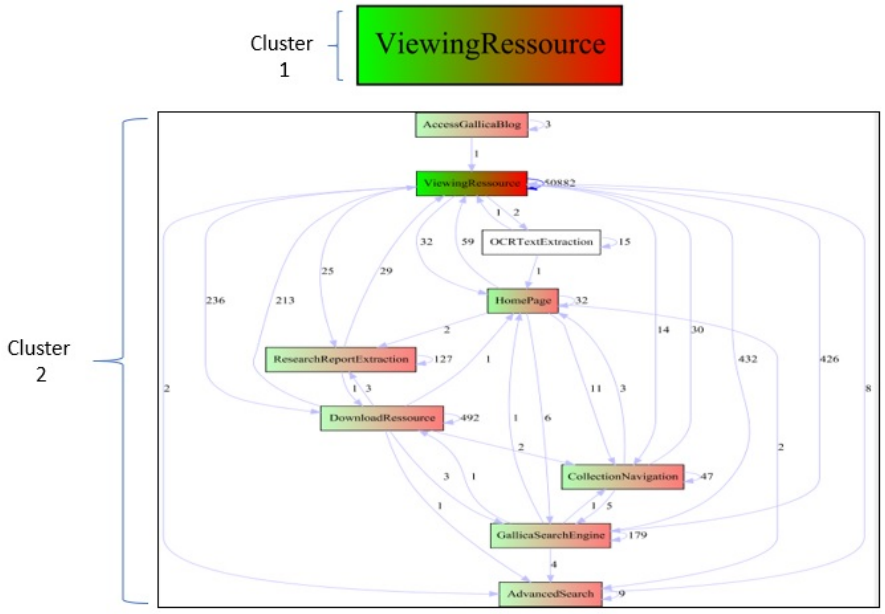


FIGURE 6.2 – Les modèles de processus découverts sur la base des *clusters* trouvés avec la méthode *freq-based* sur les 1 000 premières traces

Comme le montre la figure 6.2, le modèle issu du premier *cluster*, qui avait une bonne *Silhouette*, ne couvre qu'une seule variante de trace avec un type de recherche unique. Ce type de modèle n'est pas assez informatif, il ne peut aider ni les concepteurs dans la recommandation et l'amélioration du système ni les utilisateurs sur les fonctionnalités potentielles qu'ils peuvent effectuer pour atteindre leurs objectifs. Pour cette raison, nous soulignons qu'une méthode performante doit atteindre des bons résultats sur les deux phases d'évaluation que ce soit regroupement ou modélisation.

6.3.2 Échantillonnage aléatoire

Nous avons généré et évalué les modèles issus des *clusters* des échantillons aléatoires. La figure 6.3 décrit l'évolution des *F-measures* de chaque méthode en fonction des deux dimensions d'échantillons 20 000 et 40 000.

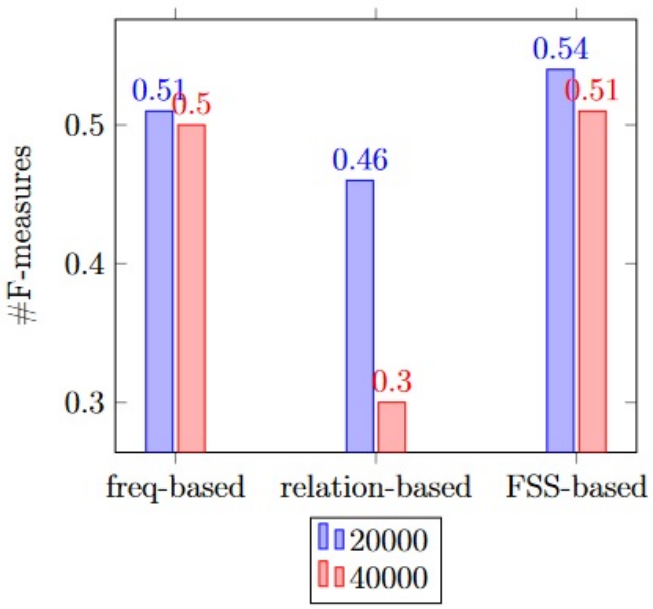


FIGURE 6.3 – Courbes des *F-measures* en fonction de la dimension des échantillons aléatoires

Les *F-measures* obtenues confirment qu'un échantillon de dimension 20 000 représente le mieux l'ensemble de traces. Toutes les méthodes obtiennent des meilleurs scores sur 20 000 traces. Les résultats montrent également que notre méthode *FSS-based* découvre des modèles plus performants que ceux générés par les autres méthodes.

6.3.3 Échantillon générique : évaluation détaillée

Suite aux deux d'expérimentations réalisées sur les traces réelles issues de *Gallica*, nous avons constaté que les performances sont assez similaires à partir de 20 000 traces. Nous avons alors choisi cet échantillon afin de mener des évaluations plus détaillées. Notre choix a été appuyé aussi par le fait que les meilleurs

F-measures données par les trois méthodes sont atteintes en moins de temps d'exécution sur des échantillons de 20 000 traces. En outre, entre l'échantillon aléatoire et l'échantillon de traces chronologiques, nous nous sommes servi de ce dernier. En effet, dans un cas réel, les fichiers logs des systèmes d'information contient des requêtes (qui engendrent des traces) avec des dates ordonnées. Dans cette section nous décrivons en détail les résultats obtenus à partir d'une série d'expériences sur des échantillons variés de 20 000 traces. Ces échantillons sont obtenus en définissant un seuil pour la longueur des traces de chaque échantillon.

La longueur des traces dans l'ensemble total de données varie de 1 à $1,5M$ avec une nette domination des traces de moins de 5 000. Toutes les traces de plus de 5 000 apparaissent une seule fois dans les journaux d'événements. La longueur moyenne de toutes les traces est de 60. Les statistiques montrent également que environ 60% de traces sont composées d'au plus 4 événements. Cela indique potentiellement que la plupart des utilisateurs sont soit rapidement bloqués, soit ils ont un niveau avancé avec les bibliothèques numériques qui leur permet d'atteindre vite leurs objectifs. La répartition du nombre de traces dans l'échantillon de 20 000 traces selon leur longueur indique que $0,8\%$ de traces contiennent plus de 5 000 évènements. Nous avons alors défini deux sous-ensembles de cet échantillon avec des longueur maximale de traces de 1 000 et 5 000 évènements. La table 6.3 présente les performances de chaque méthode dans la modélisation des deux sous-ensembles ainsi que de l'échantillon total.

Comme le montre la table 6.3, la méthode (*FSS-based*) et la méthode *relation-based* sont plus performantes que la méthode basée sur la fréquence. De plus, pour les traces comportant moins de 1 000 événements (le cas le plus réaliste), qui sont les plus fréquentes dans l'ensemble total des traces d'exécution, notre méthode (*FSS-based*) génère les meilleurs compromis entre la *Fitness* et la *Precision*. La *F-measure* obtenue avec notre méthode sur des traces de moins de 1 000 évènement surpasse les scores atteints par les deux autres méthodes sur les trois ensembles. Nous constatons que les traces peu fréquentes, qui représentent généralement un blocage de l'utilisateur ou potentiellement un oubli de session ouverte, peuvent perturber notre méthode pour bien modéliser les comportements des utilisateurs des bibliothèques numériques.

D'un point de vue qualitatif, la figure 6.4 montre que les deux modèles des deux *clusters* générés par notre méthode sont à la fois justes et précis et correspondent à deux catégories différentes d'utilisateurs. Le modèle découvert du premier *cluster* regroupe potentiellement les utilisateurs qui consultent directement les documents. Ces sessions correspondent principalement à la consultation d'un ensemble de do-

		1 000	5 000	total
freq-based	NB clusters	2	2	2
	Fitness	0,9181	0,9176	0,9176
	Precision	0,3906	0,3802	0,3802
	Generalization	0,7198	0,7199	0,7199
	F-measure	0,5481	0,5376	0,5376
relation-based	NB clusters	2	2	2
	Fitness	**0,9984**	0,9567	0,9567
	Precision	0,4172	**0,4012**	**0,4012**
	Generalization	**0,9967**	**0,9995**	**0,9995**
	F-measure	0,5884	**0,5653**	**0,5653**
FSS-based	NB clusters	2	2	2
	Fitness	0,9949	**0,9960**	**0,9965**
	Precision	**0,4244**	0,3687	0,3688
	Generalization	0,9556	0,9981	0,9981
	F-measure	**0,5950**	0,5381	0,5380

TABLE 6.3 – Résultats de la modélisation des traces sur les sous-ensembles de l'échantillon 20 000

cuments sans passer par la page d'accueil. Ils terminent ensuite leur recherche soit par une dernière consultation soit par le téléchargement du document souhaité. Dans ce *cluster*, environ 33% des utilisateurs effectuent l'action « *ViewingRessource* » au maximum deux fois par trace. D'après ce modèle, la plupart des utilisateurs n'explorent pas les collections proposées dans la bibliothèque numérique mais se connectent plutôt pour une recherche rapide et personnalisée d'un document spécifique. Enfin, les sessions de ce groupe d'utilisateurs sont courtes. Ils ne font que de brèves visites du portail *Gallica*.

Le deuxième *cluster*, par contre, est clairement destiné aux utilisateurs qui recherchent plus de ressources, ils utilisent des fonctionnalités plus avancées comme l'extraction de textes et la discussion dans le blog de *Gallica*. Le deuxième modèle, présenté dans la figure 6.4, couvre tous les chemins utilisés pour accomplir une tâche d'exploration avancée. L'une des limites de ce modèle est la complexité de son graphe généré et la possibilité d'intégration d'autre types de recherches directes (*Lookup*), qui ne commencent pas forcément par la consultation des documents (*Viewing Ressource*).

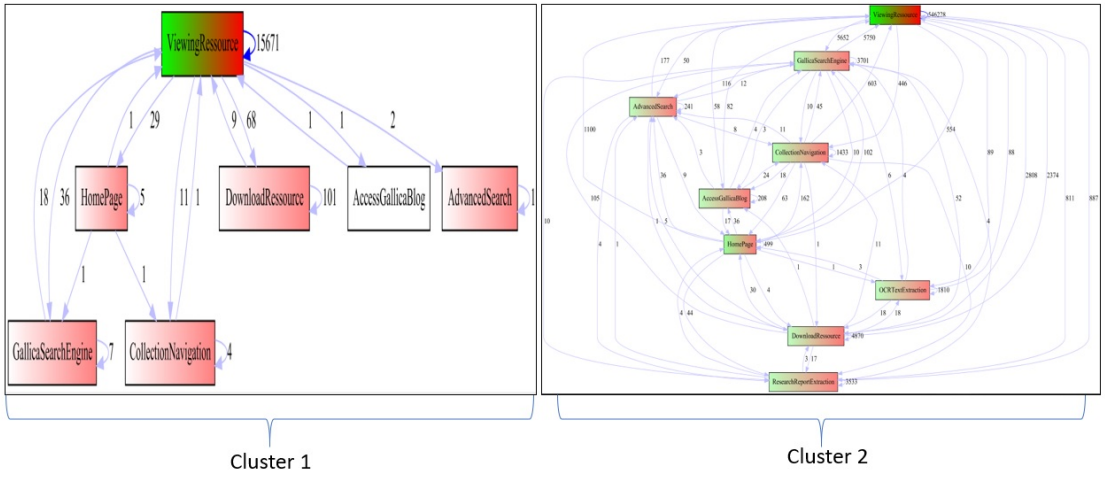


FIGURE 6.4 – Modèles de processus découverts à l'aide de la méthode *FSS-based* pour 20 000 traces comportant jusqu'à 1 000 événements.

Les figures 6.5 et 6.6 présentent les modèles découverts sur les *clusters* obtenus respectivement par les méthodes *relation-based* et *freq-based*.

Tout comme la méthode *FSS-based*, le regroupement à l'aide des méthodes *relation-based* et *freq-based* a engendré deux *clusters*. Bien que les performances des modèles de la méthode *FSS-based* sont les plus élevés, les modèles donnés par la méthode *relation-based* semblent satisfaisants qualitativement. Ils regroupent clairement deux types différents d'utilisateurs avec des navigations simples et avancées. En revanche, la méthode *freq-based* propose un modèle très basique et un modèle très complexe. La fusion du premier modèle avec le deuxième modèle semble ne pas être perturbant pour ce dernier. En effet, le modèle complexe décrit tout type de recherche et tout type d'utilisateurs. Le modèle basique se limite aux utilisateurs qui réalisent une série de consultations de documents suivie d'une activité potentielle pas forcément très fréquente. Ce modèle est clairement non pertinent et ne couvre qu'une minorité d'utilisateurs en comparant avec ceux représentés par le deuxième modèle. Le premier modèle est peu utile aux concepteurs. En effet, il ne découvre pas un type particulier d'utilisateur ou de recherche d'informations dans une bibliothèque numérique. Ce modèle a regroupé 5 934 traces composées de la seule activité « *ViewingRessource* » sur un ensemble total de 7 914 traces. La table 6.4 illustre les différents types de traces (variantes) dans le 1er *cluster*. Si on considère que le 1er modèle de processus concerne les utilisateurs qui font une recherche rapide (comme le 1er modèle du *FSS-based*), les traces de ce modèle ne

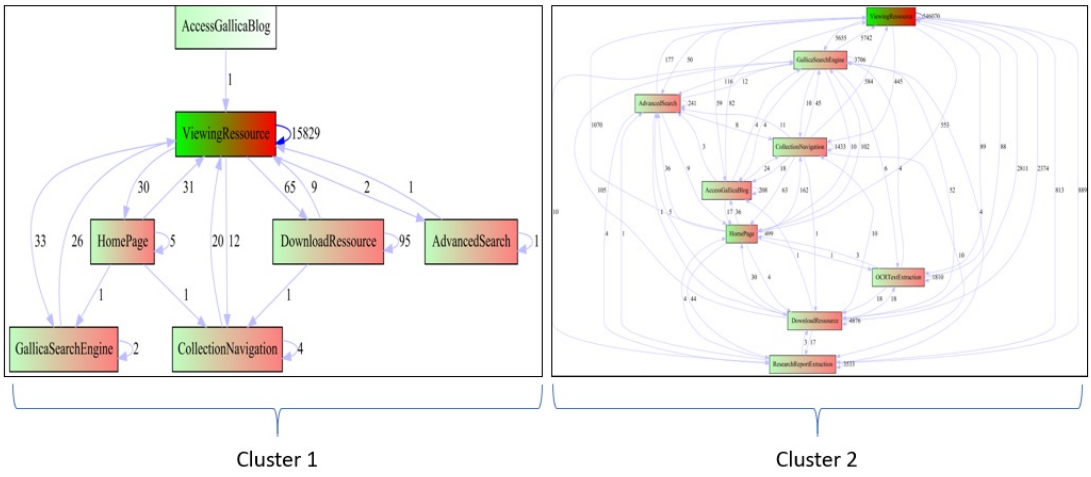


FIGURE 6.5 – Modèles de processus découverts à l'aide de la méthode *relation-based* pour 20 000 traces comportant jusqu'à 1 000 événements.

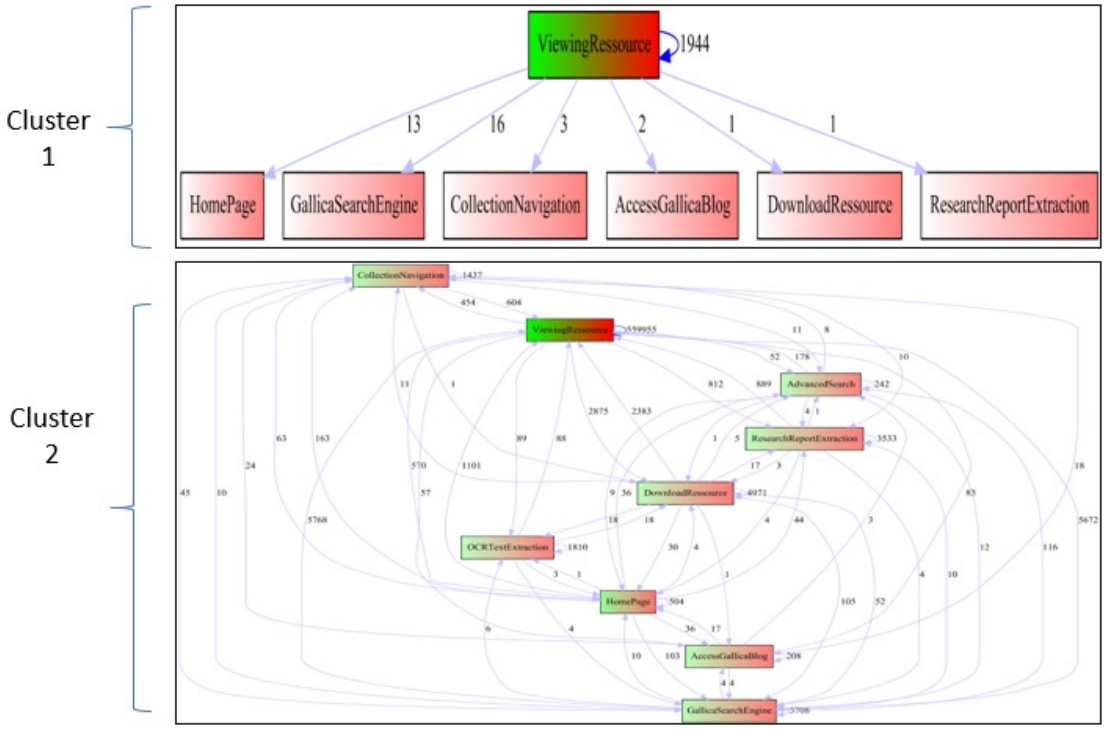


FIGURE 6.6 – Modèles de processus découverts à l'aide de la méthode *freq-based* pour 20 000 traces comportant jusqu'à 1 000 événements.

donnent pas une information complète sur les parcours utilisateurs (la majorité des traces ne sont composées que d'un seul événement). De ce fait, ce genre de modèle ne peut pas être bénéfique pour les concepteurs de *Gallica*, particulièrement pour la recommandation.

Type de trace (Variante)	**Fréquence d'apparition**
ViewingRessource	5934
ViewingRessource ViewingRessource	1944
ViewingRessource GallicaSearchEngine	16
ViewingRessource HomePage	13
ViewingRessource CollectionNavigation	3
ViewingRessource AccessGallicaBlog	2
ViewingRessource DownloadRessource	1
ViewingRessource ResearchReportExtraction	1

TABLE 6.4 – Répartition des traces regroupées dans le 1er *cluster* généré par la méthode *freq-based*

Les expériences ont montré que la modélisation des parcours des utilisateurs en utilisant l'encodage des sous-séquences fréquentes nous permet de bien séparer les groupes d'utilisateurs et de modéliser d'une manière efficace les différents comportements de chaque groupe d'utilisateurs. Nous pouvons conclure que la méthode *FSS-based* est robuste à la fois pour le clustering et la modélisation des journaux d'événements dans les bibliothèques numériques. Elle a découvert avec succès des groupes réels d'interactions d'utilisateurs avec peu et beaucoup de données d'événements, ainsi que des traces de différentes longueurs.

6.4 Conclusion

Dans ce chapitre, nous venons de présenter une série d'expérimentations que nous avons menées sur des données réelles. Notre jeu de données correspond à des traces de navigation des utilisateurs sur le portail *Gallica* de la Bibliothèque Nationale de France. Ce chapitre a permis de valider les résultats obtenus avec les traces synthétiques. Dans un cas réel, notre méthode était également robuste sur le regroupement et la modélisation. Ses performances surpassent celles obtenues par deux méthodes existantes inspirées de l'état de l'art. Notre méthode a soulevé le défi du traitement d'un grand volume de journaux provenant d'une bibliothèque numérique complexe et fréquemment consultées. Elle a permis de ressortir des

parcours typiques des utilisateurs. La qualité de regroupement des traces a été déterminée à l'aide des mesures qui étudient la compacité ou la séparation des *clusters*. La qualité des modèles de processus découverts a été mesurée en se servant des mesures qui calculent le taux de présence ou plutôt la correspondance des traces d'exécution par apport au modèle découvert.

Les modèles générés permettent d'améliorer la conception de la bibliothèque numérique en améliorant les fonctionnalités utiles liées aux événements fréquents et en découvrant les fonctionnalités peu utilisées qui nécessitent potentiellement plus de documentation. De plus, en se basant sur les chemins dans les modèles, les concepteurs peuvent proposer des recommandations pertinentes pour aider les utilisateurs novices à comprendre le modèle de navigation et à atteindre leurs objectifs en suivant une séquence d'événement inspirées des parcours réalisées par les utilisateurs expérimentés.

Résumé exécutif

- Nous avons validé notre méthode de regroupement sur un cas réel, les traces issues de *Gallica*.
- Étant le grand défi posé par la quantité énorme des traces réelles, nous avons proposé une méthodologie pour valider la qualité du modèle obtenu.

Conclusion et perspectives

Conclusion

Nous avons tous, en tant qu'êtres humains, un schéma mental qui nous permet d'appréhender les informations que l'on rencontre. Ce schéma est susceptible d'évoluer selon la situation dans laquelle nous nous trouvons. Les systèmes d'information ont, pour leur part, un schéma fixe déterminé par leurs concepteurs. Les moyens mis en œuvre par les utilisateurs et par le système d'information pour appréhender de l'information ne sont pas les mêmes. Cette différence crée un fossé sémantique qui, pour être franchi, nécessite un effort d'adaptation de la part des concepteurs. De ce fait, un système d'information est construit pour permettre à une personne d'effectuer un traitement. Il doit répondre aux besoins des utilisateurs, sa définition doit donc s'appuyer sur une connaissance précise de ce qu'ils font, de la diversité de leurs situations, etc.

Dans cette thèse, nous avons cherché à modéliser les comportements des utilisateurs de systèmes d'information. Nous avons choisi de nous placer dans des systèmes d'information pour lesquels l'utilisateur est extérieur à l'entreprise qui offre des services. C'est le cas par exemple pour une collectivité territoriale qui offre un service à un usager. Ces systèmes d'information possèdent généralement des processus métier faiblement structurés qui engendrent des traces de navigation à la fois homogènes et hétérogènes, complètes et incomplètes, on parle ici des processus d'usage faiblement structurés. La modélisation de l'historique des interactions des utilisateurs est utile à la fois aux utilisateurs afin de suivre des interactions similaires et atteindre rapidement leurs objectifs, ainsi qu'aux concepteurs pour offrir des recommandations appropriées et pour l'optimisation de l'utilisation des différentes fonctionnalités, et pour faciliter l'accès aux ressources stockées.

Dans le chapitre 1, nous avons introduit les notions de base des comportements des utilisateurs des systèmes d'information. Nous avons ainsi montré que les comportements des ces utilisateurs ont un impact important sur leurs parcours

de navigation. Nous avons présenté, par la suite, trois catégories de méthodes pour l'analyse et la modélisation des comportements des utilisateurs dans les systèmes d'information. Ce travail d'étude nous a permis de déterminer les limites actuelles de ces méthodes. Nous avons ainsi écarté les deux premières catégories le *Web usage mining* et *Web analytics* étant donné qu'ils ne produisent pas des schémas de navigation complets et nous avons retenu la fouille de processus comme catégorie de modélisation.

Dans le cadre de cette thèse, nous avons fait le choix d'étudier les parcours des utilisateurs dans les bibliothèques numériques. Le choix de ces systèmes a été basé sur plusieurs facteurs dont le plus important est la compatibilité des bibliothèques numériques avec notre problématique sur l'analyse des processus faiblement ou non structurés et dont leur modèle n'est pas (complètement) défini. Le taux croissant d'utilisation des bibliothèques numériques est aussi un facteur appréciable pour notre étude, ce qui rend la modélisation des comportements de ces utilisateurs une forte nécessité. Le chapitre 2, a été l'occasion pour nous de mieux connaître les bibliothèques numériques. L'usage des bibliothèques numériques a été étudié et il a été mis en évidence plusieurs familles d'utilisateurs typiques.

La première question que nous nous sommes posés dans cette thèse, portait sur la capacité des techniques de fouille de processus à analyser des traces de navigation issues des systèmes d'information faiblement structurés, précisément les bibliothèques numériques, et à en déterminer la structure et les variations se pose. Pour répondre à cette question, nous avons réalisé une étude exhaustive sur les différentes techniques de la fouille de processus dans le chapitre 3. Nous avons ainsi montré que la fouille de processus permettait d'extraire des modèles de comportements complets et que l'algorithme *Inductive Miner* était le plus performant (Trabelsi et al., 2019).

Cependant pour passer à un cas d'étude réel, le portail web de la Bibliothèque Nationale de France, *Gallica*, l'application immédiate des techniques de la fouille de processus nécessitent un format de données précis. Les traces brutes de navigation doivent être organisées sous la forme de sessions de recherche de chaque utilisateur. Or les données collectées (les traces brutes) ne sont pas organisées de cette manière. Elles ne reflètent que les requêtes faites. Nous avons donc proposé, dans le chapitre 4, une méthode afin de transformer les traces brutes en traces modélisées. Pour cela nous avons défini une méthode itérative qui se décline sur quatre étapes : la visualisation, le nettoyage, l'étiquetage et la sessionisation des traces brutes. Suite à ce processus de transformation, nous avons obtenu plus d'1.7 million de traces modélisées pour le mois d'avril 2017.

En outre, même si les données réelles sont devenues compatibles avec les algorithmes de fouille de processus, ces derniers vont découvrir des modèles de comportements complexes et non compréhensibles. Nous avons donc proposé une nouvelle méthode de regroupement des traces modélisées volumineuses, complexes et non structurées, en tenant compte de la qualité des groupes obtenues, le temps d'exécution ainsi que la qualité des modèles de comportements découverts. À cet effet, nous avons consacré la première partie du chapitre 5 à un état de l'art sur les différentes méthodes de regroupement des traces dans le domaine de la fouille de processus, en examinant les avantages et les limites de chaque méthodes. Nous avons décrit par la suite notre méthode de regroupement. Notre méthode se déroule en trois principales étapes : nous convertissons les traces en vecteurs de caractéristiques fréquentes. Nous appliquons ensuite une métrique de similarité pour calculer la distance entre les traces converties pour finalement attribuer le groupe approprié à chaque trace à l'aide d'un algorithme de *clustering* existant. Pendant la première étape, Notre méthode de regroupement nommée *FSS-encoding*, se sert des sous-séquences fréquentes existantes dans les traces. Nous avons donc défini une nouvelle mesure qui réalise l'encodage des sous-séquences extraites. L'originalité de cette nouvelle mesure est de tenir compte de quatre caractéristiques tirées de l'ensemble des traces ainsi que la sous séquence elle même. Nous avons ensuite comparé et validé notre méthode avec deux autres méthodes que nous avons implémenté inspirées de l'état de l'art. La validation est basée sur des traces que nous avons simulé dans une bibliothèque numérique. Les expérimentations montrent que notre méthode surpasse les méthodes existantes à la fois pour le regroupement et la modélisation des traces simulées (Trabelsi et al., 2021 a).

Le chapitre 6, a été l'occasion pour nous d'appliquer notre méthode de regroupement sur les données réelles de *Gallica*. Nous avons constaté que, pour l'application de la méthode, nous avons rencontré des difficultés liées au volume de données. Le regroupement et la modélisation d'une telle quantité de traces est très coûteuse en terme de temps. Le regroupement proposerait un nombre élevé de *clusters* ce qui est loin d'être facile à être exploité par les concepteurs de *Gallica*. Nous avons donc proposé une méthodologie d'expérimentation afin de déterminer quelle quantité et quel longueur de traces sont nécessaires afin d'obtenir une information pertinente (Trabelsi et al., 2021 a).

Nous avons donc défini une méthode pour extraire le modèle des parcours utilisateurs d'une bibliothèque numérique en nous basant sur la fouille de processus. Pour appliquer cette méthode à la bibliothèque numérique *Gallica* nous avons défini une méthode d'encodage des traces afin de regrouper celles qui correspondent

à des comportements proches, puis une autre pour gérer le volume de données à analyser.

À partir des réflexions et travaux de cette thèse nous avons développé un outil qui intègre les résultats obtenus. Cet outil, décrit en annexe A, permet d'effectuer la visualisation, le regroupement et la modélisation de parcours utilisateurs dans une bibliothèque numérique (Trabelsi et al., 2021 b).

Perspectives

Pour l'élaboration de ce travail, nous avons bien entendu fait des choix théoriques et aussi avons été conduit à limiter notre méthode sur un cas d'étude. Cependant, le coeur de la proposition demeure adaptable à d'autres cas. La méthode que nous avons développée peut être appliquée à d'autres catégories de systèmes d'information. Les résultats obtenus peuvent être généralisés à d'autres domaines tels que les sites web marchands, les services externalisés ou les environnements d'apprentissage numérique.

Plusieurs perspectives s'ouvrent à nous à l'issue de ce travail. Nous en développons trois dans les paragraphes suivants.

Affinement de l'étiquetage des traces brutes Le premier axe de perspective concerne le pré-traitement des traces brutes issues de *Gallica* et plus précisément l'étape d'étiquetage des traces brutes. En effet, nous nous sommes limités, dans ce travail, à neuf activités prédéfinis pour étiqueter chaque ligne du logs. Nous suggérons à cet effet d'affiner d'avantage ces activités afin de générer des modèles plus précis.

Validation de la méthode Nous avons validé notre méthode sur des traces issues d'une bibliothèque numérique qui a le défis d'être à la fois complexes et potentiellement incomplètes. Dans le but de mieux valider notre méthode, nous envisageons de recourir à d'autres données issus des parcours utilisateurs dans d'autres systèmes d'information. Si les résultats sur cette tâche (ou d'autres) montrent aussi de bonnes performances, notre méthode prendra alors toute sa justification car l'utilisation de nouveaux systèmes d'information ne nécessitera aucune modification à notre méthode.

Accompagnement des utilisateurs La troisième perspective que nous souhaitons explorer concerne le développement des outils d'accompagnement des utilisateurs. La caractérisation des parcours utilisateurs et le regroupement permet d'envisager des outils permettant d'accompagner l'utilisateur au cours de son parcours avec la détection des parcours typiques et atypiques. Les parcours typiques aide à mettre en place des systèmes de recommandation tandis que l'analyse des parcours atypiques peut aider à repérer des failles de sécurité ou de phénomènes émergeants. Cette perspective, à la lumière de nos travaux, fait apparaître plusieurs axes de recherche tels que :

- le développement des méthodes de regroupements en fonctions des domaines ciblés.
- le développement des méthodes de recommandation qui sont capables de bien exploiter les regroupements des traces afin de personnaliser l'expérience utilisateur et améliorer la recherche d'information. Toutefois, dans un tel contexte, la question qui se pose est de savoir comment lui prescrire le chemin à suivre et que faire en cas de remise en cause ?

Annexe A

Outil développé pour la visualisation des parcours utilisateurs

A.1 Introduction

Dans le cadre de la thèse, nous avons développé un outil pour modéliser les traces de navigation des utilisateurs des bibliothèques numériques. Notre outil permet de visualiser, regrouper et modéliser les parcours des utilisateurs. Une démonstration de notre outil a été fournie dans le lien en bas de page [1]. Nous avons implémenté quatre méthodes pour regrouper les utilisateurs similaires selon leurs traces et nous avons intégré trois algorithmes de découverte de processus pour générer des modèles de processus en fonction des types d'utilisateurs ainsi que leurs tâches de recherche d'information. La présentation de notre outil a été acceptée dans le cadre d'un poster à la conférence internationale *JCDL* (*ACM/IEEE Joint Conference on Digital Libraries*) (Trabelsi et al., 2021 b).

Notre outil a pour but de générer des modèles de comportement qui couvrent les différents parcours des utilisateurs et à fournir des modèles de processus de bout en bout qui montrent et expliquent les choix et les boucles des utilisateurs. Cependant, comme nous avons évoqué dans le chapitre 4, l'utilisation croissante des bibliothèques numériques conduit à un ensemble de données volumineuses et complexes. Compte tenu du nombre exponentiel de requêtes (Sumikawa et al., 2019) et lorsqu'elles sont appliquées aux traces de navigation des utilisateurs d'une biblio-

1. https://drive.google.com/file/d/1mggDeAo7zRqz3sJl9a9j3B7oPhxj0clx/view?usp=sharing

thèque numérique, les techniques de la fouille de processus retournent généralement des modèles de processus complexes, non génériques pour tous les utilisateurs. De ce fait, les objectifs de notre outil devraient également être basés sur la visualisation et le regroupement de traces partageant des comportements similaires des utilisateurs.

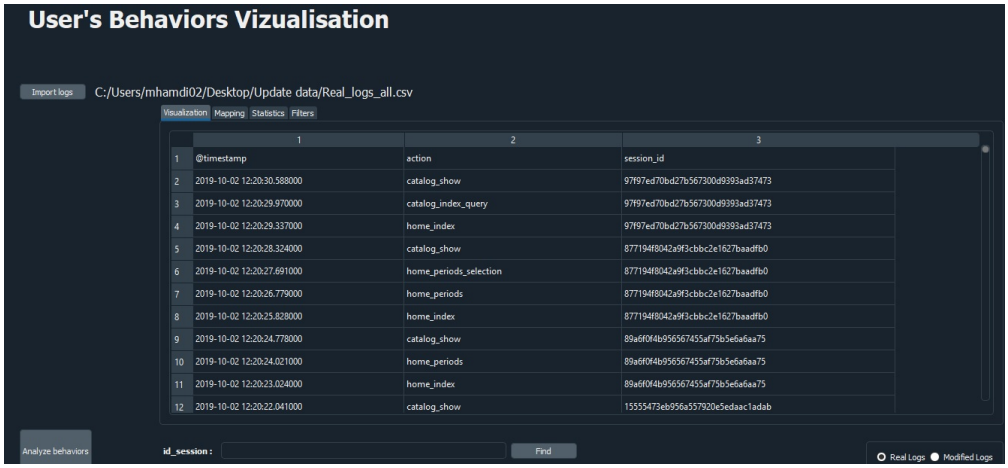
Notre outil dispose de trois principales interfaces : la **visualisation des traces de navigation** (cf. section A.2), le **regroupement des traces similaires** (cf. section A.3) et la **modélisation des traces** respectivement (cf. section A.4). Nous les décrivons en se basant sur un échantillon de traces simulées dans une bibliothèque numérique (cf. section 5.4.1).

A.2 Visualisation des traces

Notre outil traite les traces de navigation suivant le modèle d'événements que nous avons mentionné dans la section 3.2 enregistrées dans des fichiers *CSV*. Ces traces peuvent être directement obtenues à partir des requêtes des utilisateurs réels dans les bibliothèques numériques. Comme le montre la figure A.1, les fichiers *CSV* doivent définir au moins l'action performée par l'utilisateur (un événement), son *Timestamp* et l'identifiant de la session de recherche (*SessionId*). Les fichiers peuvent aussi inclure d'autres informations comme les lieux, les adresses *IP* et tout autre mot clé trouvé dans les requêtes. Les utilisateurs de l'application ont la possibilité de visualiser les traces selon le format tabulaire des fichiers *CSV* comme le montre la figure A.1 ou sous un autre format en regroupant les événements liés à un même utilisateur ou bien évidement dans notre cas le même identifiant de session (cf. figure A.2).

En outre, notre outil permet d'effectuer le *mapping* des nouveaux fichiers logs afin de reconnaître les différents champs nécessaires pour les algorithmes de la fouille de processus. De ce fait, pour les nouveaux fichiers, c'est à l'utilisateur de créer son nouveau *mapping* de traces, en indiquant quel est le champ qui correspond à l'identifiant des traces (Caseid), les événements réalisés ainsi que le *timestamp* de chaque événement exécuté (cf. figure A.3). L'utilisateur peut indiquer également le format de son *timestamp* vu que ça peut changer selon le système.

Dans le but de faire une analyse statistique sur les traces, l'outil fournit ainsi une représentation statistique, comprenant un histogramme qui représente le nombre d'événements réalisés. Nous indiquons également le nombre des lignes totales dans le fichier, le nombre totale de traces ainsi la moyenne des événements par utilisa-



	1	2	3
1	@timestamp	action	session_id
2	2019-10-02 12:20:30.588000	catalog_show	97f97ed70bd27b567300d9393ad37473
3	2019-10-02 12:20:29.970000	catalog_index_query	97f97ed70bd27b567300d9393ad37473
4	2019-10-02 12:20:29.337000	home_index	97f97ed70bd27b567300d9393ad37473
5	2019-10-02 12:20:28.324000	catalog_show	877194f8042a9f3cbbc2e1627baadfb0
6	2019-10-02 12:20:27.691000	home_periods_selection	877194f8042a9f3cbbc2e1627baadfb0
7	2019-10-02 12:20:26.779000	home_periods	877194f8042a9f3cbbc2e1627baadfb0
8	2019-10-02 12:20:25.828000	home_index	877194f8042a9f3cbbc2e1627baadfb0
9	2019-10-02 12:20:24.778000	catalog_show	89a6f0f4b956567455af75b5e6a6aa75
10	2019-10-02 12:20:24.021000	home_periods	89a6f0f4b956567455af75b5e6a6aa75
11	2019-10-02 12:20:23.024000	home_index	89a6f0f4b956567455af75b5e6a6aa75
12	2019-10-02 12:20:22.041000	catalog_show	15555473eb956a557920e5edaac1adab

FIGURE A.1 – Visualisation des traces sous leur format original

	1	2
1	session_id	session_actions
2	ebae14317af361ecb2903bd625cfe982	home_index home_topics home_show_topic_selection home_show_topic_selection catalog_show home_periods home_periods_selection catalog_show catalog_
3	7b8416f56f48784d0cbb068c784c75d1	home_index catalog_index_query catalog_show home_topics home_show_topic_selection catalog_show home_topics home_show_topic_selection catalog_show
4	0c96a032b9245655cacb76f8a76c28d7	home_index home_languages home_languages_selection catalog_show catalog_show catalog_index_query catalog_index_filter catalog_show catalog_index_qu
5	ade0f13d6b4212242e00acf726952efa	home_index home_topics home_show_topic_selection home_show_topic_selection catalog_show home_periods home_periods_selection catalog_show catalog_
6	7a62fee235fecdcbaf329c8f249b5fbb	home_index catalog_index_query catalog_show home_topics home_show_topic_selection catalog_show home_topics home_show_topic_selection catalog_show
7	6312fc1e4c54d73797f2713ceb7d2af9	home_index home_languages home_languages_selection catalog_show catalog_show catalog_index_query catalog_index_filter catalog_show catalog_index_qu
8	3d596f570d3f208f1598348f4e59a5b3	home_index home_topics home_show_topic_selection home_show_topic_selection catalog_show home_periods home_periods_selection catalog_show catalog_
9	d0f3d5a8c20865a68873f339cff31905	home_index catalog_index_query catalog_show home_topics home_show_topic_selection catalog_show home_topics home_show_topic_selection catalog_show
10	a34f6dc9c0efbdb07782ec2f04d779ff	home_index home_languages home_languages_selection catalog_show catalog_show catalog_index_query catalog_index_filter catalog_show catalog_index_qu
11	1bcde1d91980f84dda0739497144f077	home_index home_topics home_show_topic_selection home_show_topic_selection catalog_show home_periods home_periods_selection catalog_show catalog_

FIGURE A.2 – Visualisation des traces sous leurs format modifié

teur.

Les statistiques ont montré, par exemple, sur d'autres échantillons de traces, que les utilisateurs de *Gallica* utilisent rarement la reconnaissance optique de caractères (*OCR*) pour extraire du texte à partir des images et que, lorsque cette fonctionnalité est utilisée, elle est suivie de nombreux événements. Cela signifie potentiellement que la fonctionnalité n'est pas simple à réaliser ou que la plupart des utilisateurs ne la connaissent pas ou ne savent pas l'utiliser.

En outre, l'outil propose de nettoyer le bruit dans l'ensemble des traces selon des critères spécifiques en se basant sur des filtres. Comme le montre la figure A.5), trois types de filtres ont été intégrés dans l'outil. Ces filtres permettent à l'utilisateur de supprimer une activité précise de l'ensemble de traces, supprimer des traces complètes et de supprimer des traces selon une plage horaire. Ceci est particuliè-

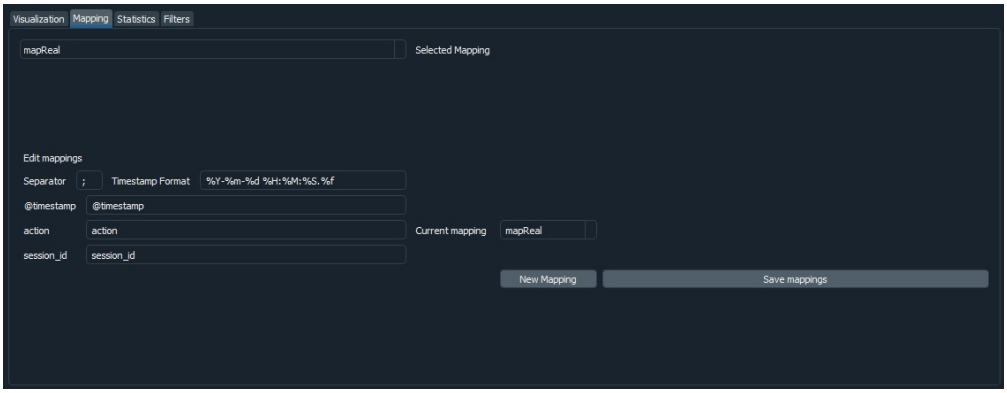


FIGURE A.3 – Le *mapping* des fichiers log

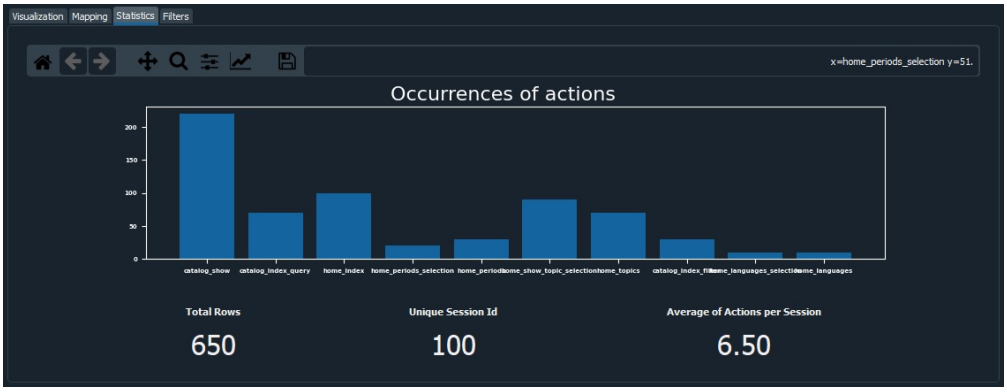


FIGURE A.4 – Statistiques effectuées sur un échantillon de traces

rement utile pour supprimer les traces liées à des *bots-crawlers*, des événements non pertinents qui peuvent perturber la modélisation des parcours des utilisateurs et/ou les retours en arrière qui accentuent généralement les hésitations.

Une fois que l'utilisateur a terminé la visualisation et le nettoyage des traces, il peut passer à l'étape du regroupement.

A.3 Regroupement des traces similaires

Notre outil offre la possibilité de découvrir directement un modèle de comportement à partir de l'ensemble des traces. Il permet ainsi de regrouper les traces similaires avant de passer à la modélisation. Le regroupement des traces d'une manière significative nous permet de générer des modèles clairs et exploitables.

Comme nous l'avons mentionné dans le chapitre 5, le regroupement des traces

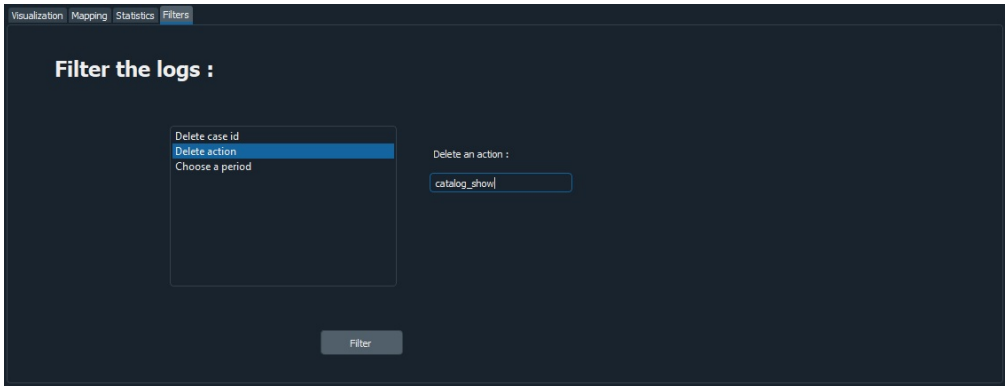


FIGURE A.5 – Nettoyage du bruit dans les traces à l'aide des filtres

doit être effectué sur trois étapes successives. La préparation des traces qui consiste à convertir les traces en des vecteurs de caractéristiques. La similarité entre les traces qui consiste à mesurer la distance entre les vecteurs de traces et le regroupement des traces qui consiste à utiliser un algorithme de *clustering* afin de regrouper les vecteurs en tenant compte des distances. L'outil propose ainsi quatre méthodes pour la préparation des traces : les deux premières méthodes sont inspirées de l'état de l'art. Elles sont basées sur la fréquence des événements (*freq-based*) ou la fréquence de la succession directe entre eux (*relation-based*) Van der Aalst (2016). La (*freq-based*) méthode considère les traces comme similaires lorsqu'elles ont les mêmes événements fréquents. Alors que la méthode *relation-based* regroupe les traces ayant le même comptage des successions directes entre les événements dans l'ensemble des traces. Nous pensons qu'en plus de la fréquence et de la succession, les sous-séquences fréquentes dans les traces sont pertinentes pour regrouper les traces. Nous proposons donc deux autres méthodes au niveau des sous-séquences. La première (*FSS identifier*) regroupe les traces partageant des sous-séquences fréquentes tandis que la deuxième (*FSS encoding*) prend en compte plus de détails sur les sous séquences fréquentes tels que leurs longueurs, leurs fréquences, positions et la succession entre les événements dans les sous séquences fréquentes. La figure A.6 montre les différentes méthodes de préparation des traces. Par la suite, l'utilisateur a la possibilité de choisir la mesure de similarité qui le convient ainsi que l'algorithme de clustering. Suite au regroupement des traces, l'utilisateur de l'application peut passer à l'étape de modélisation ou la découverte des modèles de processus.

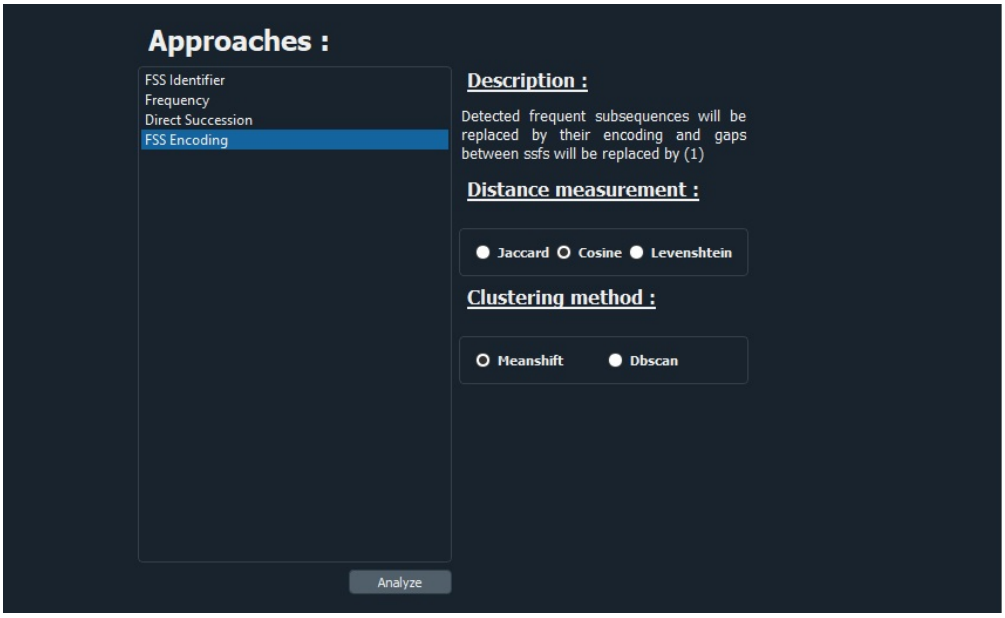


FIGURE A.6 – Regroupement des traces similaires

A.4 Modélisation des comportements

La découverte et la visualisation des modèles de comportements des utilisateurs est considérée parmi les fonctionnalités les plus importantes de notre outil. Elle permet à l'utilisateur de générer des modèles de processus à partir de l'ensemble des traces d'une part et de découvrir les modèles liées aux groupes de traces de l'autre part. Afin d'avoir une idée sur la qualité du regroupement, l'outil fournit également les scores liées à la qualité des *clusters* générés. De ce fait, les scores *Silhouette* ainsi que le *Davies Bouldin* ont été intégrés dans notre outil. La figure A.7 montre par exemple les scores obtenus sur un échantillon de traces. Cette page est dédiée à la fois pour l'affichage des résultats de regroupement ainsi que le choix des algorithmes de découverte des processus et la génération des modèles.

Comme l'indique la figure A.7, deux algorithmes de découverte de processus [2] ont été intégrés à l'outil : l'*Inductive Miner* et le *Directly Follows Miner*. L'algorithme *Inductive Miner* a montré une capacité à extraire des modèles de comportements pertinents et complets des utilisateurs des bibliothèques numériques Trabelsi et al. (2019). Le deuxième algorithme que nous avons intégré est le *Directly Follows Miner*. Ce dernier qui est très utilisé par les commerciaux qui souhaitent montrer des modèles de processus simples et compréhensibles à leurs clients. Comme nous

2. `https://pm4py.fit.fraunhofer.de/getting-started-page#discovery`

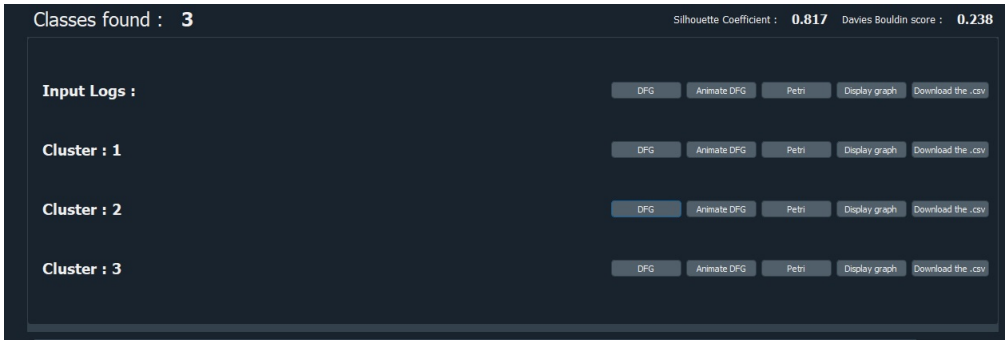


FIGURE A.7 – La page d'accueil pour la modélisation des comportements

l'avons indiqué dans le chapitre 3, l'*Inductive Miner* génère des modèles sous la forme des réseaux de *Petri* (cf. figure A.8). Suite à la génération du réseau de *Petri*, L'utilisateur aura aussi les mesures liées à la qualité du modèle découvert. Les mesures que nous avons intégré ont été également décrites dans la section 3.6.

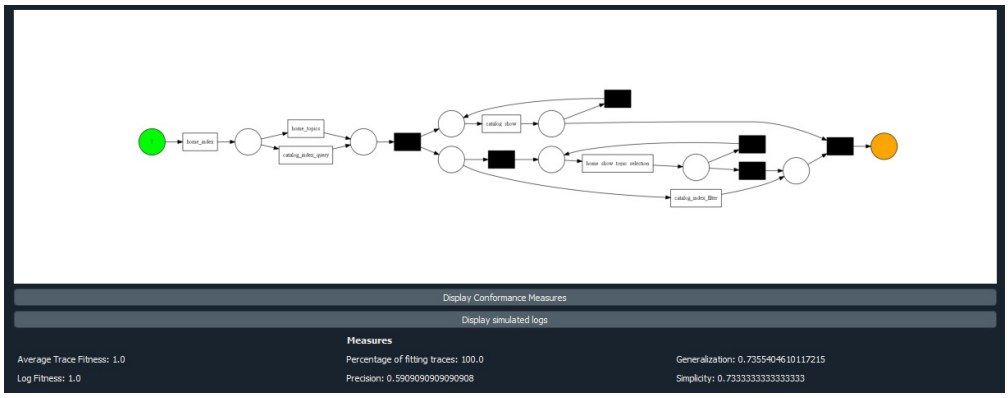


FIGURE A.8 – Le réseau de *Petri* découvert à partir d'un groupe de traces

Quant à l'algorithme *Directly Follows Miner*, ce dernier découvre des modèles sous la forme de graphes de successions directes, dits des *Directly Follows Graphs* (DFG), codant les événements et les relations entre eux (cf. figure A.9).

En outre, nous avons implémenté l'animation des graphes de successions directes découverts (cf. figure. A.9). L'idée consiste à rejouer les traces de départ sur les modèles obtenus. Nous considérons que l'animation fournie peut aider les concepteurs ainsi qu'un simple utilisateur à comprendre le modèle d'une part, et de voir la chronologie des traces exécutées par les utilisateurs d'un autre part. Finalement, notre outil permet également de récupérer les fichiers correspondant aux *clusters* de traces résultantes. Ces fichier sous le format *CSV* peuvent être par

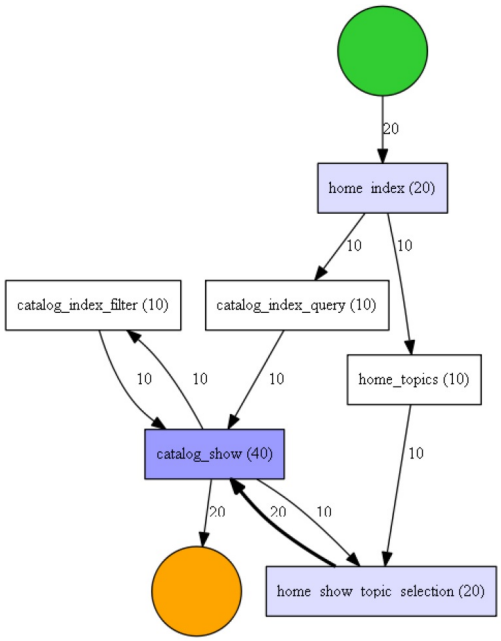


FIGURE A.9 – Le graphe de succession directe découvert à partir d'un groupe de traces

la suite réutilisés par d'autres outils connus par la communauté de la fouille de processus comme *PROM* et *Disco*[3].

A.5 Conclusion

Nous avons présenté dans cette annexe un nouvel outil qui découvre automatiquement des modèles de comportements des utilisateurs des bibliothèques numériques à partir des journaux d'événements en suivant une série d'étapes. L'outil propose trois méthodes efficaces de modélisation. Le principal objectif de notre application est de mettre en évidence les différents types d'utilisateurs d'une bibliothèque numérique en proposant différents modèles disjoints pertinents pour une classe particulière d'utilisateurs et/ou de tâches de recherche d'information. Dans les travaux futurs, nous prévoyons d'enrichir l'outil afin de traiter d'avantage de types de logs. Nous prévoyons également d'alimenter l'application par d'autres algorithmes de clustering et des techniques supplémentaires de la fouille de processus afin de cibler d'autres systèmes d'information.

3. `https://fluxicon.com/disco/`

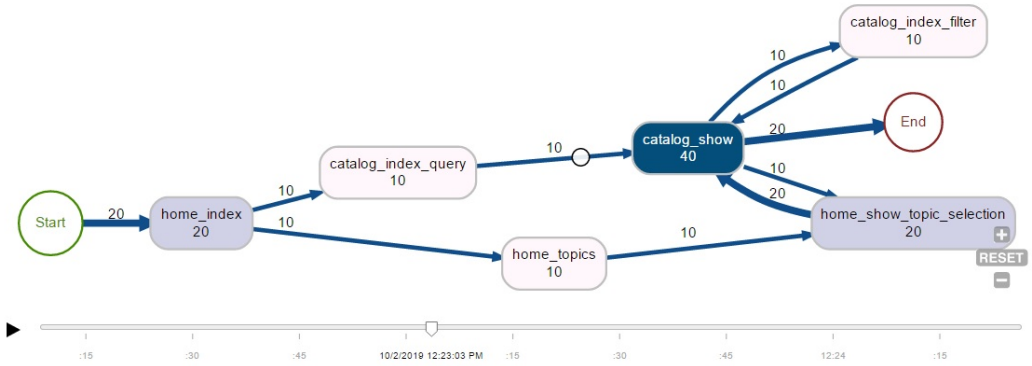


FIGURE A.10 – Le rejoue des traces sur le graphe des successions directes découvert à partir d'un groupe de traces

Annexe B

Publications scientifiques

Conférences Internationales

- **Marwa Trabelsi**, Cyrille Suire, Jacques Morcos, and Ronan Champagnat, A New Methodology to Bring Out Typical Users Interactions in Digital Libraries, The ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL), 2021
- **Marwa Trabelsi**, Cyrille Suire, Jacques Morcos, and Ronan Champagnat, User-Centred Application for Modeling Journeys in Digital Libraries, The ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL), 2021
- **Marwa Trabelsi**, Cyrille Suire, Jacques Morcos, and Ronan Champagnat, User's Behavior in Digital Libraries : Process Mining Exploration, 23rd International Conference on Theory and Practice of Digital Libraries TPDL, 2019 . **Best poster award**.

Conférences Nationales

- **Marwa Trabelsi**, Cyrille Suire, Jacques Morcos, and Ronan Champagnat, Fouille de processus auto-définis : cas d'étude d'un moteur de recherche d'une bibliothèque numérique, INFormatique des ORganisations et Systèmes d'Information et de Décision INFORSID, 2019.

Bibliographie

Adriansyah, A. (2014), Aligning observed and modeled behavior, thèse de Doctorat, Technische Universiteit Eindhoven.

Alves de Medeiros, A., Van Dongen, B., Van Der Aalst, W. and Weijters, A. (2004), « Process mining : Extending the alpha-algorithm to mine short loops », Univ. of Technology, Eindhoven **113**, 145–180.

Beaudouin, V., Garron, I. and Rollet, N. (2016), Je pars d'un sujet, je rebondis sur un autre, thèse de Doctorat, Télécom ParisTech ; Bibliothèque nationale de France ; Labex Obvil.

Bernard, G. and Andritsos, P. (2017), A process mining based model for customer journey mapping, dans « Forum and Doctoral Consortium Papers Presented at the 29th International Conference on Advanced Information Systems Engineering (CAiSE 2017) », Vol. 1848, CEUR Workshop Proceedings, 49–56.

Blandford, A. (2004), Understanding user's experiences : evaluation of digital libraries, dans « DELOS workshop on evaluation of digital libraries Padova, Italy », 77–98.

Blandford, A. (2006), « Interacting with information resources : digital libraries for education », International journal of learning technology **2**(2-3), 185–202.

Bogaard, T., Hollink, L., Wielemaker, J., Hardman, L. and Van Ossenbruggen, J. (2019), Searching for old news : User interests and behavior within a national collection, dans « Proceedings of the 2019 Conference on Human Information Interaction and Retrieval », 113–121.

Bogaard, T., Hollink, L., Wielemaker, J., van Ossenbruggen, J. and Hardman, L. (2018), « Metadata categorization for identifying search patterns in a digital library », Journal of Documentation .

Bose, R. J. C. and Van der Aalst, W. M. (2009 a), Context aware trace clustering : Towards improving process mining results, dans « Proceedings of the 2009 SIAM International Conference on Data Mining », SIAM, 401–412.

Bose, R. J. C. and van der Aalst, W. M. (2009 b), Trace clustering based on conserved patterns : Towards achieving better process models, dans « International Conference on Business Process Management », Springer, 170–181.

Buijs, J. C., Van Dongen, B. F. and van Der Aalst, W. M. (2012), On the role of fitness, precision, generalization and simplicity in process discovery, dans « OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" », Springer, 305–322.

Cadez, I., Heckerman, D., Meek, C., Smyth, P. and White, S. (2000), Visualization of navigation patterns on a web site using model-based clustering, dans « Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining », 280–284.

Ceravolo, P., Damiani, E., Torabi, M. and Barbon, S. (2017), Toward a new generation of log pre-processing methods for process mining, dans « International Conference on Business Process Management », Springer, 55–70.

Chatain, T., Carmona, J. and Van Dongen, B. (2017), Alignment-based trace clustering, dans « International Conference on Conceptual Modeling », Springer, 295–308.

Cifter, A. S. and Dong, H. (2009), « User characteristics : Professional vs. lay users ».

Cole, M. J., Hendahewa, C., Belkin, N. J. and Shah, C. (2015), « User Activity Patterns During Information Search », ACM Trans. Inf. Syst. **33**(1), 1 :1–1 :39.

Comaniciu, D. and Meer, P. (2002), « Mean shift : A robust approach toward feature space analysis », IEEE Transactions on pattern analysis and machine intelligence **24**(5), 603–619.

Davis, J. and Goadrich, M. (2006), The relationship between precision-recall and roc curves, dans « Proceedings of the 23rd international conference on Machine learning », 233–240.

De Koninck, P. and De Weerdt, J. (2019), Scalable mixed-paradigm trace clustering using super-instances, dans « 2019 International Conference on Process Mining (ICPM) », IEEE, 17–24.

De Koninck, P., Nelissen, K., Baesens, B., vanden Broucke, S., Snoeck, M. and De Weerdt, J. (2017), An approach for incorporating expert knowledge in trace clustering, dans « International Conference on Advanced Information Systems Engineering », Springer, 561–576.

De Weerdt, J., Vanden Broucke, S., Vanthienen, J. and Baesens, B. (2013), « Active trace clustering for improved process discovery », IEEE Transactions on Knowledge and Data Engineering **25**(12), 2708–2720.

Di Francescomarino, C., Dumas, M., Maggi, F. M. and Teinemaa, I. (2016), « Clustering-based predictive process monitoring », IEEE transactions on services computing **12**(6), 896–909.

Dinet, J. (2009), « Pour une conception centrée-utilisateurs des bibliothèques numériques », Communication langages (3), 59–74.

Djouad, T., Settouti, L. S., Mille, A., Reffay, C. and Prié, Y. (2010), « Sbt-im : Un système à base de traces pour le calcul des' indicateurs d'interaction dans moodle », Rapport de recherche RR-LIRIS-2010-002 .

Ellis, D. and Haugan, M. (1997), « Modelling the information seeking patterns of engineers and research scientists in an industrial environment », Journal of documentation .

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. et al. (1996), A density-based algorithm for discovering clusters in large spatial databases with noise., dans « Kdd », Vol. 96, 226–231.

Følstad, A. and Kvale, K. (2018), « Customer journeys : a systematic literature review », Journal of Service Theory and Practice .

Günther, C. and Verbeek, E. (2016), « Ieee standard for extensible event stream (xes) for achieving interoperability in event logs and event streams ».

Günther, C. W. (2009), Process mining in flexible environments, thèse de Doctorat, Technische Universiteit Eindhoven.

Hansen, P. and Järvelin, K. (2005), « Collaborative information retrieval in an information-intensive domain », Information Processing & Management **41**(5), 1101–1119.

Hassani, M., van Zelst, S. J. and van der Aalst, W. M. (2019), « On the application of sequential pattern mining primitives to process discovery : Overview, outlook and opportunity identification », Wiley Interdisciplinary Reviews : Data Mining and Knowledge Discovery e1315.

Ho, H. N., Rabah, M., Nowakowski, S. and Estraillier, P. (2017), « Approche de présélection multicritère à base de traces pour la prise de décision dans les applications interactives de type jeux », Revue d'Intelligence Artificielle (RIA) **31**(3), 311–335.

Hompes, B., Buijs, J., Van der Aalst, W., Dixit, P. and Buurman, J. (2015), Discovering deviating cases and process variants using trace clustering, dans « Proceedings of the 27th Benelux Conference on Artificial Intelligence (BNAIC), November », 5–6.

Hu, X. and Cercone, N. (2004), « A data warehouse/online analytic processing framework for web usage mining and business intelligence reporting », International Journal of Intelligent Systems **19**(7), 585–606.

Huurnink, B., Hollink, L., Van Den Heuvel, W. and De Rijke, M. (2010), « Search behavior of media professionals at an audiovisual archive : A transaction log analysis », Journal of the American society for information science and technology **61**(6), 1180–1197.

Jalali, M., Mustapha, N., Sulaiman, M. N. and Mamat, A. (2010), « Webpum : A web-based recommendation system to predict user future movements », Expert Systems with Applications **37**(9), 6201–6212.

Jans, M., Alles, M. and Vasarhelyi, M. (2013), « The case for process mining in auditing : Sources of value added and areas of application », International Journal of Accounting Information Systems **14**(1), 1–20.

Johnson, A. (2008), « Users, use and context : supporting interaction between users and digital archives », What Are Archives ? : Cultural and Theoretical Perspectives : A Reader 145–64.

Kani-Zabihi, E., Ghinea, G. and Chen, S. Y. (2006), « Digital libraries : what do users want ? », Online Information Review .

Kani-Zabihi, E., Ghinea, G. and Chen, S. Y. (2008), « User perceptions of online public library catalogues », International journal of information management **28**(6), 492–502.

Kumar, B. S. and Rukmani, K. (2010), « Implementation of web usage mining using apriori and fp growth algorithms », Int. J. of Advanced networking and Applications **1**(06), 400–404.

Kumar, L., Singh, H. and Kaur, R. (2012), Web analytics and metrics : a survey, dans « Proceedings of the International Conference on Advances in Computing, Communications and Informatics », 966–971.

Leemans, S. J., Fahland, D. and van der Aalst, W. M. (2013), Discovering block-structured process models from event logs containing infrequent behaviour, dans « International conference on business process management », Springer, 66–78.

Leemans, S. J., Poppe, E. and Wynn, M. T. (2019), Directly follows-based process mining : Exploration & a case study, dans « 2019 International Conference on Process Mining (ICPM) », IEEE, 25–32.

Liang, S. and Leng, Y. (2020), Search topic analysis of acm digital library, dans « Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020 », 487–488.

Lu, X., Tabatabaei, S. A., Hoogendoorn, M. and Reijers, H. A. (2019), Trace clustering on very large event data in healthcare using frequent sequence patterns, dans « International Conference on Business Process Management », Springer, 198–215.

Mans, R. S., Schonenberg, M., Song, M., van der Aalst, W. M. and Bakker, P. J. (2008), Application of process mining in healthcare–a case study in a dutch hospital, dans « International joint conference on biomedical engineering systems and technologies », Springer, 425–438.

Marchionini, G. (2006), « Exploratory Search : From Finding to Understanding », Commun. ACM **49**(4), 41–46.

Margolin, V. (1997), « Getting to know the user », Design studies **18**(3), 227–236.

Marty, J.-C. and Mille, A. (2009), Traces, traces d'interactions, traces d'apprentissages : définitions, modèles informatiques, structurations, traitements et usages, Hermes Sciences Publications.

Masseglia, F., Poncelet, P., Teisseire, M. and Marascu, A. (2006), Web usage mining : extraction de périodes denses à partir des logs, dans « EGC : Extraction et Gestion des Connaissances », Cépaduès-Éditions, 403–408.

Meij, E., Bron, M., Hollink, L., Huurnink, B. and de Rijke, M. (2011), « Mapping queries to the linking open data cloud : A case study using dbpedia », Journal of Web Semantics **9**(4), 418–433.

Michel, C. (2002), Le web usage mining, méthodes et expérimentation pour l'extraction d'information et de connaissance sur les données du web., dans « Actes de la 7ème conférence AIM :" Affaire Électronique et Société de Savoir : Opportunités et Défis". 30 mai-1er juin 2002-Hammamet, Tunisie ».

Mughal, M. J. H. (2018), « Data mining : Web data mining techniques, tools and algorithms : An overview », Information Retrieval **9**(6).

Murata, T. (1989), « Petri nets : Properties, analysis and applications », Proceedings of the IEEE **77**(4), 541–580.

Nouvellet, A., Beaudoin, V., D'alché-Buc, F., Prieur, C. and Roueff, F. (2017), Analysis of gallica usage traces, Research report, Télécom ParisTech ; Bibliothèque nationale de France.
URL: *https ://hal.archives-ouvertes.fr/hal-01709264*

Papy, F. and Jakubowicz, C. (2021), Bibliothèque numérique et innovation, Vol. 2, ISTE Group.

Paskali, L., Ivanovic, L., Kapitsaki, G., Ivanovic, D., Surla, B. D. and Surla, D. (2021), « Personalization of search results representation of a digital library », Information Technology and Libraries **40**(1).

Patil, U. M. and Patil, J. B. (2012), Web data mining trends and techniques, dans « Proceedings of the International Conference on Advances in Computing, Communications and Informatics », 961–965.

Pel, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U. and Hsu, M. (2001), Prefixspan : Mining sequential patterns by prefix-projected growth, dans

« Proc. 17th IEEE International Conference on Data Engineering (ICDE). Heidelberg, Germany », 215–224.

Piccoli, G. (2007), Information systems for managers : texts and cases, Wiley Publishing.

Poggi, N., Muthusamy, V., Carrera, D. and Khalaf, R. (2013), Business process mining from e-commerce web logs, dans « Business process management », Springer, 65–80.

Rousseeuw, P. J. (1987), « Silhouettes : a graphical aid to the interpretation and validation of cluster analysis », Journal of computational and applied mathematics **20**, 53–65.

Rozinat, A., de Medeiros, A. K. A., Günther, C. W., Weijters, A. and van der Aalst, W. M. (2007), The need for a process mining evaluation framework in research and practice, dans « International Conference on Business Process Management », Springer, 84–89.

Sani, M. F., van Zelst, S. J. and van der Aalst, W. M. (2018), Applying sequence mining for outlier detection in process mining, dans « OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" », Springer, 98–116.

Sani, M. F., van Zelst, S. J. and van der Aalst, W. M. (2019), The impact of event log subset selection on the performance of process discovery algorithms, dans « European Conference on Advances in Databases and Information Systems », Springer, 391–404.

Santra, A. and Jayasudha, S. (2012), « Classification of web log data to identify interested users using naïve bayesian classification », International Journal of Computer Science Issues (IJCSI) **9**(1), 381.

Saura, J. R., Palos-Sánchez, P. and Cerdá Suárez, L. M. (2017), « Understanding the digital marketing environment with kpis and web analytics », Future Internet **9**(4), 76.

Smeaton, A. F. and Callan, J. (2005), « Personalisation and recommender systems in digital libraries », International Journal on digital libraries **5**(4), 299–308.

Song, M., Günther, C. W. and Van der Aalst, W. M. (2008), Trace clustering in process mining, dans « International Conference on Business Process Management », Springer, 109–120.

Srivastava, J., Cooley, R., Deshpande, M. and Tan, P.-N. (2000), « Web usage mining : Discovery and applications of usage patterns from web data », Acm Sigkdd Explorations Newsletter **1**(2), 12–23.

Suire, C. (2018), Recherche d'information et humanités numériques : une approche et des outils pour l'historien, thèse de Doctorat, Université de La Rochelle.

Suire, C., Jean-Caurant, A., Courboulay, V., Burie, J.-C. and Estraillier, P. (2016), User Activity Characterization in a Cultural Heritage Digital Library System, dans « Proc. of the 16th ACM/IEEE-CS on Joint Conf. on Digital Libraries », JCDL '16, ACM, New York, USA, 257–258.

Sumikawa, Y., Jatowt, A., Doucet, A. and Moreux, J.-P. (2019), Large scale analysis of semantic and temporal aspects in cultural heritage collection's search, dans « 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL) », IEEE, 77–86.

Talakokkula, A. (2015), « A survey on web usage mining, applications and tools », Computer Engineering and Intelligent Systems **6**(2), 22–29.

Trabelsi, M., Suire, C., Morcos, J. and Champagnat, R. (2019), User's behavior in digital libraries : Process mining exploration, dans « International Conference on Theory and Practice of Digital Libraries », Springer, 388–392.

Trabelsi, M., Suire, C., Morcos, J. and Champagnat, R. (2021 a), A new methodology to bring out typical users interactions in digital libraries, dans « 2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL) », 11–20.

Trabelsi, M., Suire, C., Morcos, J. and Champagnat, R. (2021 b), User-centred application for modeling journeys in digital libraries, dans « 2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL) », 332–333.

Van der Aalst, W. (2012), « Process mining : Overview and opportunities », ACM Trans. on Management Information Systems **3**(2), 7.

Van der Aalst, W. (2016), Process mining : Data Science in Action, Springer.

Van der Aalst, W., Adriansyah, A. and Van Dongen, B. (2011), Causal nets : a modeling language tailored towards process discovery, dans « Int. conf. on concurrency theory », Springer, 28–42.

Van der Aalst, W. M. (1996), « Structural characterizations of sound workflow nets », Computing science reports **96**(23), 18–22.

Van der Aalst, W. M., De Medeiros, A. A. and Weijters, A. (2005), Genetic process mining, dans « ICATPN », Springer, 48–69.

Van der Aalst, W. M., Rubin, V., Verbeek, H., van Dongen, B. F., Kindler, E. and Günther, C. W. (2010), « Process mining : a two-step approach to balance between underfitting and overfitting », Software & Systems Modeling **9**(1), 87.

Van Der Aalst, W. M., Van Hee, K. M., Ter Hofstede, A. H., Sidorova, N., Verbeek, H., Voorhoeve, M. and Wynn, M. T. (2011), « Soundness of workflow nets : classification, decidability, and analysis », Formal aspects of computing **23**(3), 333–363.

Van der Aalst, W., Weijters, T. and Maruster, L. (2004), « Workflow mining : Discovering process models from event logs », IEEE Trans. on Knowledge & Data Eng. (9), 1128–1142.

Van der Werf, J. M. E., van Dongen, B. F., Hurkens, C. A. and Serebrenik, A. (2008), Process discovery using integer linear programming, dans « ICATPN », Springer, 368–387.

Van Dongen, B. F., De Medeiros, A. A. and Wen, L. (2009), Process mining : Overview and outlook of petri net discovery algorithms, dans « Transactions on Petri Nets and Other Models of Concurrency II », Springer, 225–242.

Van Dongen, B. F., de Medeiros, A. K. A., Verbeek, H., Weijters, A. and Van Der Aalst, W. M. (2005), The prom framework : A new era in process mining tool support, dans « ICATPN », Springer, 444–454.

vanDongen, S. (2000), « A cluster algorithm for graphs », Information Systems [INS] (R 0010).

Veiga, G. M. and Ferreira, D. R. (2009), Understanding spaghetti models with sequence clustering for prom, dans « International conference on business process management », Springer, 92–103.

Waisberg, D. and Kaushik, A. (2009), « Web analytics 2.0 : empowering customer centricity », The original Search Engine Marketing Journal **2**(1), 5–11.

Walsh, D., Clough, P., Hall, M. M., Hopfgartner, F., Foster, J. and Kontonatsios, G. (2019), Analysis of transaction logs from national museums liverpool, dans « International Conference on Theory and Practice of Digital Libraries », Springer, 84–98.

Weijters, A. and Ribeiro, J. (2011), Flexible heuristics miner (fhm), dans « Computational Intelligence and Data Mining », IEEE, 310–317.

Whyte, G. and Bytheway, A. (1996), « Factors affecting information systems' success », International journal of service industry management .

Wilson, L. A. (2003), « If we build it, will they come ? library users in a digital world », Journal of library administration **39**(4), 19–28.

Wilson, T. D. (1981), « On user studies and information needs », Journal of documentation .

Witten, I. H., Bainbridge, D. and Boddie, S. J. (2001), Power to the people : end-user building of digital library collections, dans « Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries », 94–103.

Witten, I. H., Bainbridge, D. and Nichols, D. M. (2009), How to build a digital library, Morgan Kaufmann.

Xie, H. I. (2008), « Users' evaluation of digital libraries (dls) : Their uses, their criteria, and their assessment », Information processing & management **44**(3), 1346–1373.

Xie, I., Babu, R., Lee, T. H., Castillo, M. D., You, S. and Hanlon, A. M. (2020), « Enhancing usability of digital libraries : Designing help features to support blind and visually impaired users », Information Processing & Management **57**(3), 102110.

Zandkarimi, F., Rehse, J.-R., Soudmand, P. and Hoehle, H. (2020), A generic framework for trace clustering in process mining, dans « 2020 2nd International Conference on Process Mining (ICPM) », IEEE, 177–184.

Zheng, G. and Peltsverger, S. (2015), Web analytics overview, dans « Encyclopedia of Information Science and Technology, Third Edition », IGI Global, 7674–7683.

Modélisation des processus utilisateurs à partir des traces d'exécution, application aux systèmes d'information faiblement structurés.

Résumé : Ce travail porte sur l'analyse des parcours utilisateurs dans les bibliothèques numériques qui se caractérisent par des processus métier faiblement structurés. Nous allons avoir recours à la fouille de processus afin d'extraire les modèles à partir de ces parcours. Les modèles découverts permettront aux concepteurs de ces systèmes de répondre d'une manière plus efficace aux besoins des utilisateurs et à leurs présenter un ensemble de recommandations. Dans cette thèse, nous avons généré des modèles à partir des traces de navigation des utilisateurs du portail web de la bibliothèque nationale de France, Gallica. Dans un premier temps, nous avons adapté ces traces sous un format compatible avec les algorithmes de fouille de processus. Dans un second temps, nous avons regroupé ces traces. L'originalité de notre apport porte sur le regroupement des parcours similaires, en tenant compte des caractéristiques existantes dans les traces, afin d'éviter la génération des modèles complexes, souvent non exploitables, à partir de telles traces volumineuses et non structurées. Enfin, nous avons validé notre méthode sur deux jeux de données simulé et réel. Nous avons comparé notre méthode à deux méthodes inspirées des travaux existants et les résultats montrent que notre méthode surpasse celles existantes sur les deux jeux de données à la fois dans le regroupement et la modélisation.

Mots clés : Découverte des modèles de comportements, Fouille de processus, Bibliothèque numérique, Regroupement des traces.

User's Journey Modeling Based on Process Mining for Information Systems with Unstructured Processes

Abstract: This research focuses on extracting users' journeys in a digital library characterized by weakly structured business processes. In this thesis, we investigate whether it is possible to model user journeys using process mining. The discovered models allow system designers to respond more efficiently to users' needs and to present them with a set of recommendations. For our study, we have chosen to extract the users' journey models of the digital library Gallica, based on real traces generated by their users. First, we adapt these browsing traces in a well-defined format compatible with process mining techniques. The originality of our contribution concerns the grouping of similar paths, considering the existing characteristics in the traces, to avoid the generation of complex models, often not exploitable, from such voluminous and unstructured traces. Finally, we validate our method on two simulated and real data sets. We compare our method to two other methods inspired by existing works. The results show that our method outperforms the existing ones on both datasets in clustering and modeling.

Keywords: User's journey modeling, Process mining, Digital library, Traces clustering.

Laboratoire Informatique, Image, Interaction
Faculté des Sciences & Technologie
Avenue Michel Crépeau
17042 LA ROCHELLE CEDEX 1