Chapitre 3 : ShipMine : un atelier d'extraction de connaissances pour l'analyse de comportements

3.1. Introduction

Nous souhaitons à partir de ce travail de thèse répondre à la problématique d'aide à l'analyse des comportements à risques de navires en proposant une méthodologie d'extraction de connaissances sur les comportements, basée sur la fouille de données. Pour cela, un ciblage des problèmes, méthodes et algorithmes permettant d'extraire ces connaissances est fondamental. Ensuite, pour valider la méthodologie, les algorithmes ciblés pour l'analyse des comportements de navires vont être testés sur des données réelles de mouvements et d'événements historiques de navires. Ces algorithmes ont été intégrés dans un seul environnement qui va servir de preuve de concept pour montrer l'intérêt de la fouille de données au domaine de la modélisation de comportements potentiellement à risques. Cet environnement reprend les différentes phases de la méthodologie proposée dans ce travail de recherche.

Ce chapitre est organisé en deux parties : la première partie concerne la conception de l'environnement et la seconde, la présentation de son fonctionnement. Dans la partie conception, les utilisateurs potentiels ainsi qu'une analyse de besoins d'extraction de connaissances sur les comportements à risques sont présentés, la sélection des algorithmes et leur intégration dans l'environnement sont détaillées, les données à utiliser dans l'exploration sont aussi présentées et enfin l'architecture du prototype est exposée avec les technologies qui ont été choisies pour son implémentation. Dans la seconde partie de ce chapitre, la présentation du prototype et son fonctionnement sont exposés.

3.2. Conception et réalisation de l'atelier

L'environnement que nous voulons développer est un atelier d'aide à l'extraction de connaissances sur les comportements normaux et anormaux de navires. L'atelier, que nous avons nommé ShipMine, va intégrer un ensemble d'algorithmes et de programmes informatiques visant à automatiser l'extraction de comportements. L'interprétation de ces comportements, va aider à l'acquisition de connaissances sur les risques et leur modélisation.

ShipMine va utiliser la fouille de données pour explorer des données maritimes de déplacement de navires et d'événements historiques. Aujourd'hui, il y a énormément de problèmes, méthodes et algorithmes de fouille de données qui ont été conçus pour

l'extraction de connaissances à partir de données. Les méthodes les plus récentes permettent d'extraire des motifs de mouvements inhabituels, fréquents, périodiques et des résumés à partir de données de déplacements. Dans ce travail, nous nous intéressons à la découverte de connaissances sur les comportements à risques de navires. La difficulté à laquelle nous sommes confrontés est que les méthodes de fouille de données actuelles n'ont pas été conçues pour extraire ce genre de comportements.

Nous rappelons que le comportement étant une combinaison de mouvements dans des situations (Cf. section 2.1 du chapitre 2), il devient alors possible d'initier l'analyse des comportements à partir de motifs de mouvements et de situations pouvant décrire des comportements à risques. L'exploration des données maritimes en utilisant la fouille de données va donc permettre de découvrir des connaissances sur les comportements potentiellement à risques, sous forme de mouvements et de situations.

Pour supporter tout le cycle de vie du développement de notre atelier d'extraction de connaissances basé sur la fouille de données, un modèle en spirale[69] a été retenu. Le choix de ce modèle peut être justifié par le fait qu'il est incrémental, ce qui permet d'avoir des résultats intermédiaires. Ce modèle est bien adapté à notre cas car les résultats ne sont pas connus à l'avance. Les tests des résultats intermédiaires vont enrichir l'atelier au fur et à mesure de l'avancement des développements.

Les phases de développement sont représentées par un schéma en spirales où la première boucle est la définition des besoins, la dernière est la livraison, et entre les deux, un nombre variable de boucles (itérations) qui sont décomposées en quatre phases :

- Détermination des objectifs : choix d'une fonctionnalité, ciblage des algorithmes, acquisition et traitement des données,
- Identification et gestion des risques : données (non accessibles, non exploitables, etc.), algorithme (non adapté, très complexe à implémenter, performances),
- Développements, tests et validation,
- Planification de la prochaine phase : choix d'une nouvelle fonctionnalité.

Dans la suite, la phase de *Détermination des objectifs* et la phase de *Gestion des risques* sont incluses dans les sections *Ciblage et adaptation des algorithmes* (Cf. section

[69] Boehm, 1988, http://weblog.erenkrantz.com/~jerenk/phase-ii/Boe88.pdf

3.2.4) et *Construction de l'espace de données à explorer* (Cf. section 3.2.5). Le développement de l'atelier est traité quant à lui dans la section *Choix technologiques* ci-dessous et le *test et la validation* dans le chapitre suivant.

3.2.1. A qui s'adresse cet atelier ?

Sur l'échelle temporelle, l'atelier permet une analyse *a posteriori* (Cf. section 2 de l'introduction). Les connaissances extraites automatiquement à partir des comportements observés vont aider à modéliser des comportements à risques. Donc pour bien utiliser l'atelier, il faut avoir des compétences en analyse de données et des connaissances dans le domaine maritime.

L'atelier est destiné à des analystes qui seront en charge d'analyser, de raisonner sur les résultats obtenus par l'atelier et de modéliser des comportements à risque à partir de ces résultats. Les autorités maritimes ne sauvegardent pas les déplacements de navires et travaillent juste sur les dernières positions connues. Seuls quelques centres de recherche sauvegardent les historiques de données maritimes dans l'objectif de les utiliser en recherche. Il est difficile donc d'imaginer quels acteurs maritimes actuels vont utiliser cet atelier. Nous supposons que les analystes dont on parle, peuvent êtres des experts maritimes qui ont reçu une formation sur l'utilisation de l'atelier, des scientifiques qui s'intéressent à l'analyse des risques maritimes ou des analystes recrutés spécialement pour utiliser l'atelier. Nous appelons « expert maritime » toute personne capable de valider les résultats de l'atelier, de décider seul ou après approbation de l'intérêt des informations générées et de pouvoir construire des connaissances aidant à modéliser les comportements à risques.

3.2.2. Analyse de besoins

Dans l'objectif de tester et valider notre méthodologie d'extraction de connaissances sur les comportements potentiellement à risques, un ensemble de ces comportements doit être établi pour essayer de cibler des méthodes de fouille de données permettant de les extraire. Il est à rappeler que l'analyse de ces besoins est issue de la littérature.

Pour cela, nous avons choisi deux types de situations et trois types de mouvements pouvant décrire des risques (Figure 3-1). Les deux types de situations (en bleu) sont les zones à risques et les facteurs de risques et les trois types de mouvements (en rouge) sont

les navigations proches, les trajectoires et les routes de navigation (trajectoires types) à risques. Il est possible à partir de ces types de situations et mouvements à risques de générer plusieurs comportements à risques. Les mouvements et les situations qui composent ces comportements vont être définis comme des fonctionnalités que doit assurer notre atelier.

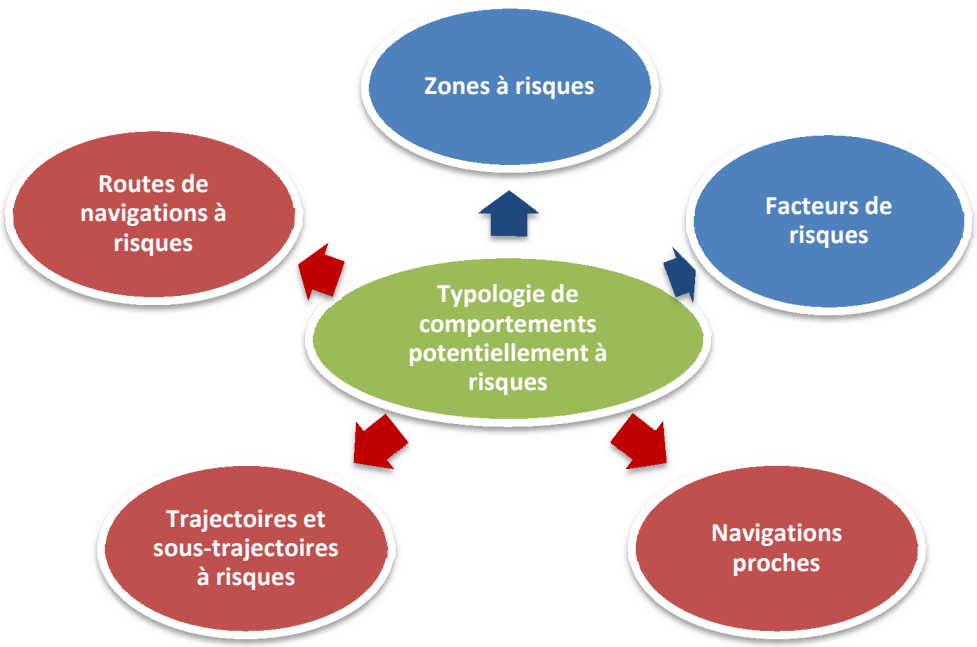

Figure 3-1 : Typologies non exhaustive de comportements potentiellement à risques choisis pour être des fonctionnalités de l'atelier ShipMine.

Cette typologie est composée de cinq comportements :

- **Facteurs à risques :** ce sont des relations entre des facteurs propices à une situation à risque comme par exemple, une relation entre un type de navire et un accident quand il se trouve dans des conditions de navigation particulières,

- **Zones à risques** : zones à forte concentration spatiale d'événements à risques (zones accidentogènes, de piraterie, etc.),

- **Trajectoires et sous trajectoires à risques** : trajectoires décrivant potentiellement un comportement à risque (trajectoire de dérive, trajectoire avec plusieurs changements de directions, etc.),

- **Routes de navigation à risques** : trajectoires type décrivant des navigations à risques comme par exemple les trajectoires de navires transportant des matières dangereuses et sensibles,

- **Navigations proches** : trajectoires proches décrivant les navires qui se rapprochent pendant une durée de temps. Cela peut représenter un risque d'abordage, de transbordement ou de pêche parallèle.

3.2.3. Ciblage et adaptation des algorithmes de fouille de données

Dans les sous-sections suivantes, nous exposons la méthode qui a été adoptée pour identifier, évaluer et tester les algorithmes de fouille de situations et de mouvements pour l'extraction de connaissances décrivant des risques. Nous avons présenté dans la section 3.2.2, quelques comportements à risques. L'extraction de ces comportements est assurée par les fonctionnalités de ShipMine. Il faut trouver pour ces fonctionnalités des méthodes et des algorithmes de fouille de données pouvant les assurer. La Figure 3-2 montre bien cette correspondance entre les fonctionnalités, les méthodes de fouille de données et les algorithmes qui ont été choisis. Nous avons pris la précaution de choisir des méthodes issues de plusieurs domaines, à savoir la fouille de données classique, spatiales et spatio-temporelles pour montrer leur intérêt.

A ce stade de développement, notre atelier ShipMine n'intègre pas toutes les méthodes présentées sur la Figure 3-2 mais juste celles qui ont recours à un affichage cartographique. C'est le cas des méthodes d'analyse de mouvements et l'une des méthodes d'analyse de situations (détection de zones denses). Les méthodes de détection des associations vont être utilisées dans un programme tiers, en dehors de ShipMine. A terme, toutes les méthodes seront intégrées.

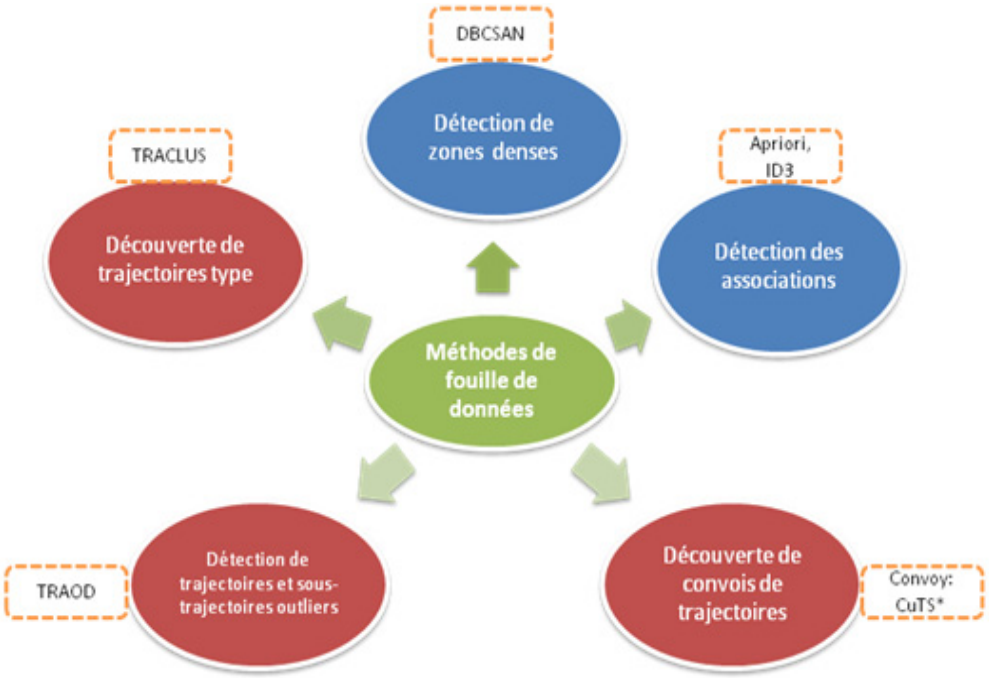

Figure 3-2 : Les méthodes de fouille de données et les algorithmes choisis pour extraire des situations (bleu) et des mouvements (rouge) pouvant décrire des comportements à risques.

3.2.3.1. Analyse de situations

Une situation à risque est définie par une relation entre facteurs propices à un type de risque ou par une concentration spatiale d'événements à risques. Dans l'analyse des relations, nous avons comme ambition de sélectionner des méthodes permettant de découvrir des relations entre des facteurs de contexte (avarie, feu cassé, etc.), d'environnement (conditions météorologiques, océanographiques, etc.) et l'apparition d'événements non souhaités. Dans l'analyse de concentration, nous voulons sélectionner des méthodes permettant de découvrir des zones à risques à partir de concentrations spatiales d'événements à risques.

3.2.3.1.1. Détection de facteurs de risques

Pour découvrir les facteurs de risques et leurs relations automatiquement, il est possible de mettre toutes les variables au même niveau et de chercher les relations (implication, corrélation, etc.) entre elles ou de définir une variable cible à expliquer par d'autres variables dites explicatives. Les deux approches ont été testées et présentées ci-dessous. La première utilise les règles d'association et la deuxième les arbres de décision.

Les règles d'association ont été choisies dans l'objectif de trouver des régularités dans les bases de données d'événements à risques sous forme d'éléments associés fréquemment ensemble. Nous nous sommes inspirés pour cela de l'application du panier de la ménagère (Cf. section 2.2.1.1 du chapitre 2).

Nous avons défini trois grandes catégories de résultats des règles d'association qui sont de la forme « *si antécédent alors conséquent* », pour faciliter leur exploitation :

- **Règles de prédiction** : Nous appelons « règle de prédiction » toute règle ayant son antécédent connu *a priori*, son conséquent non connu et la confiance de la règle est supérieure à 50%. Une règle de prédiction peut être du genre "*Si nous avons un accident dans un contexte Ci alors à c% des cas, il est causé par le type d'accident Ti* ",

- **Règles de ciblage** : Ce sont les règles de connaissances générales qui identifient les relations entre les différentes dimensions (type de navire, type d'accident, zone maritime, etc.). L'antécédent et le conséquent de la règle sont connus mais pas la relation d'implication entre les deux parties. Les règles sont par exemple du genre "*Les accidents de navires de type Ti, concernent à c% les navires de type Ni*" et "*Les accidents de navires de type Ti sont localisés dans c% des cas dans la zone Zi*",

- **Règles banales** : Ce sont les règles qui n'apportent pas d'informations nouvelles.

Pour l'implémentation, l'algorithme *Apriori* (Srikant & Agrawal 1996) développé par *Christian Borgelt* a été choisi. Cette implémentation est récupérée à partir du package *Rattle 2.6.4* de R[70]. Les mesures *support*, *confiance et Lift* (Cf. section 2.2.1.1) vont être utilisées pour sélectionner les règles intéressantes. Le lift est calculé en plus du support et de la confiance pour vérifier que les résultats obtenus ne sont pas le fruit du hasard. Si la mesure du lift est supérieure à 1, la règle est gardée pour une analyse de pertinence.

Dans la deuxième approche, nous allons définir une variable cible à expliquer par d'autres variables dites explicatives. Pour cela, le classement par arbre de décision a été

[70] http://www.inside-r.org/packages/cran/rattle

choisi car il fournit des règles explicites de classement, il est robuste et sa représentation en graphe permet une meilleure facilité de compréhension. L'algorithme ID3 implémenté dans Weka[71] 3.6.8 a été choisi. Cet algorithme est détaillé dans la section 2.2.1.2.

3.2.3.1.2. Détection de zones à risques

Le groupement de localisations d'événements non souhaités comme les accidents de navires en utilisant une mesure de proximité géographique va peut être nous permettre d'extraire des zones à risques. Pour découvrir ces zones à risques, plusieurs méthodes de fouille spatiale et plus particulièrement de clustering peuvent être utilisées comme les méthodes par partitionnement, les méthodes hiérarchiques et les méthodes par densité (Cf. section 2.2.1.3).

Nous avons choisi d'utiliser les méthodes de clustering par densité pour deux raisons. La première concerne la détection automatique du nombre de clusters ce qui permet de ne pas biaiser les résultats en fixant ce nombre *a priori*. La deuxième raison est relative à la capacité de ce type de méthode à récupérer des clusters ayant des formes arbitraires. Parmi les algorithmes de clustering de densité, notre choix s'est porté sur DBSCAN par ce qu'il présente une faible complexité *O(n log n)* et les résultats générés sont visuels et textuels ce qui permet une validation plus facile de ces résultats. Pour plus de détails sur cet algorithme voir la section 2.2.1.3.

Pour l'implémentation de DBSCAN, un programme *java* est proposé sur internet[72] en téléchargement et exploitation libre. Ce programme est proposé avec une Graphical User Interface (GUI) que nous avons dû enlever pour pouvoir appeler le programme d'une manière transparente à partir de ShipMine. Nous avons réalisé d'autres modifications pour retourner les résultats à ShipMine à la fin de l'exécution et pour calculer des polygones englobant les zones de densité découvertes. L'appel du programme à partir de ShipMine comprend les valeurs des paramètres d'Eps (rayon de voisinage), MinPts (seuil minimum de points), les chemins des fichiers de données à explorer et ceux des résultats de l'exécution.

[71] http://www.cs.waikato.ac.nz/ml/index.html
[72] https://github.com/KanwarBhajneek/DBSCAN/tree/master/src/dbscan

Les résultats retournés sont de deux types : les clusters avec les positions d'événements associées et les polygones englobant chaque cluster. Les polygones sont calculés en utilisant le parcours de Graham (Graham & Yao 1983) pour représenter les zones à risques. Le parcours de Graham décrit les plus petits polygones contenant l'ensemble des points de chaque cluster. Le code source de cet algorithme est proposé en téléchargement libre[73].

3.2.3.2. Analyse de mouvements à risques

Les mouvements potentiellement à risques peuvent êtres des formes de trajectoires particulières, la proximité de deux ou plusieurs trajectoires et des formes de trajectoires différentes de l'habituel (trajectoire avec des cercles, des zigzags, etc.). Ces mouvements peuvent décrire un risque comme une dérive, une pollution, un objet tombé à la mer ou un accident. Pour extraire ces motifs, nous nous sommes inspirés des travaux de l'équipe de J. Han (Lee et al., 2007) (Lee et al., 2008a) (Li et al., 2010c) qui proposent l'identification de trajectoires aberrantes (*Outlier Trajectory*), le clustering de trajectoires et les travaux de Jeung sur les convois (Jeung et al., 2008a; Jeung et al., 2008b).

3.2.3.2.1. Détection de trajectoires anormales de navires

Pour extraire les motifs de trajectoires anormales peuvant correspondre à un risque, l'algorithme d'extraction de trajectoires aberrantes TRAOD (Cf. section 2.2.3.1.1 du chapitre 2) a été choisi. Nous rappelons qu'une trajectoire aberrante est une trajectoire qui ne ressemble pas au comportement général des autres trajectoires voisines.

Le programme Visual C++ de cet algorithme a été téléchargé à partir de la page personnelle de Jae-Gil Lee[74] qui est l'un des auteurs de TRAOD. Ensuite, il a été testé sur leurs données d'expérimentation pour vérifier qu'il donne les mêmes résultats.

Le programme original possède une GUI pour sélectionner les données, saisir les paramètres, exécuter le programme et afficher les résultats sur une interface graphique. Les auteurs ont utilisé un référentiel spatial relatif à l'interface d'affichage.

[73] Code source sur http://www.script-tout-fait.com/fr/script-Concevoir-une-enveloppe-convexe-avec-l-algorithme-de-Graham-31.html
[74] http://dm.kaist.ac.kr/jaegil/sources/

Nous avons enlevé l'interface du programme pour permettre son exécution d'une manière transparente par rapport à l'utilisateur, qui aura l'impression que tout se fait au niveau de ShipMine. D'autres changements ont été opérés pour adapter le programme à notre application comme la modification des paramètres en entrée, omettre la phase de simplification des données MDL (Cf. section 2.2.3.1.1) et la repondération des calculs de distance. Ces changements sont détaillés ci-après.

Le paramètre F représentant la proportion de longueur des partitions aberrantes pour que toute la trajectoire soit considérée comme aberrante a été écarté. Le fait que les trajectoires soient très longues, la proportion F même petite soit-elle, élimine beaucoup trop de trajectoires aberrantes. L'idée est de garder toutes les partitions aberrantes quelle que soit la proportion de ses partitions sur l'ensemble de la trajectoire.

Après avoir fait plusieurs tests sur nos jeux de données, nous nous sommes rendus compte que l'algorithme d'origine ne détectait que les longues partitions dues souvent à la perte de signal AIS et non à des comportements de navires aberrants. La Figure 3-3 montre l'un des résultats de ces tests qui a donné 30 pertes de signal AIS à partir d'un historique de 65 trajectoires de tankers localisés en Méditerranée. Le résultat obtenu n'est pas celui espéré car l'objectif est d'identifier les partitions aberrantes des trajectoires dues à des mouvements de navires et non à des pertes de signal AIS. Tout porte à croire que la simplification MDL utilisée, enlève les petites partions aberrantes qui nous intéressent, c'est pour cela qu'elles ne sont pas détectées. En effet, après la simplification, les trajectoires sont devenues trop lisses, peut être à cause de l'hétérogénéité des formes et des longueurs de partitions de trajectoires.

Nous avons choisi de suspendre la simplification MDL pour remédier au problème. Ce qui donne une partition de trajectoire entre deux positions successives. Cela n'aura pas d'impact sur les performances car nous travaillons sur des données non volumineuses.

Figure 3-3 : Partitions de trajectoires aberrantes détectées après une phase de simplification MDL (p=98%, D=10Km).

Les distances peuvent être pondérées selon l'importance que l'on veut donner à chacune des distances que sont les distances perpendiculaire, parallèle et angulaire (Cf. section 2.2.3.1.1, page 84). Après plusieurs tests, les pondérations ont été fixées à $w_{\perp}=1$, $w_{\parallel}=0$, $w_{\theta}=5$ car cela donne plus d'importance à la distance angulaire et permet de détecter les trajectoires qui font par exemple des zigzags, des cercles et des dérives. La distance parallèle a été mise à zéro car elle n'a pas d'intérêt dans notre application dû au fait que les positions ne sont pas récupérées d'une manière synchrone.

3.2.3.2.2. Découverte de routes de navigation

Dans la majorité des cas, les navires de transport de personnes et de marchandises suivent les mêmes routes. Ils font souvent des allers et retours entre différents ports. La découverte de ces routes de navigation peut aider par exemple à identifier les routes de transport de matières dangereuses, polluantes ou identifier les trajectoires de navires qui ne suivent pas ces routes de navigation (comportement anormal).

Plusieurs méthodes de groupement de trajectoires peuvent être utilisées pour la découverte de ces routes. Notre choix s'est porté sur les méthodes de clustering de trajectoires basées sur les densités. L'algorithme TRACLUS vu dans la section 2.2.3.1.2 répond bien à nos attentes. Il est basé sur l'algorithme DBSCAN. Il utilise donc les mêmes concepts de densité en considérant les segments de ligne à la place des points. Une trajectoire représentative est créée pour chaque cluster, cette trajectoire décrit le mouvement du cluster.

L'algorithme commence par partitionner les trajectoires en petites partions avant de commencer la phase de groupement. Dans le partitionnement, il utilise des points caractéristiques pour simplifier les trajectoires avant leur exploration. Le choix de l'algorithme est motivé par le fait qu'il soit précis et efficace.

Les auteurs de TRACLUS proposent le téléchargement du code source C++ de l'algorithme sur internet[75]. Des modifications ont été apportées au programme pour pouvoir l'appeler à partir de ShipMine d'une manière transparente aux utilisateurs.

Après plusieurs tests sur la pondération de distances utilisée dans TRACLUS, elle a été fixée d'une manière empirique à $w_{\perp}=1$, $w_{\parallel}=0$, $w_{\theta}=1.2$. Cette pondération donne un peu plus d'importance à la distance angulaire et donne de meilleurs résultats. La distance parallèle a été mise à zéro car elle n'a pas d'intérêt du fait que les positions ne sont pas récupérées d'une manière synchrone. Cette pondération permet à l'algorithme de regrouper les partitions de trajectoires parallèles car elles sont considérées comme plus proches (voisines) que celles ayant des différences angulaires.

Le calcul de la trajectoire représentative à partir des clusters de segments de lignes est basé sur le concept de connexion de densité (Cf. section 2.2.3.1.2). Pour éviter de connecter les clusters espacés entre eux, une valeur de la distance égale à 500 mètres a été choisie après plusieurs tests. La validation de cette distance a été effectuée d'une manière visuelle en comparant les différents résultats de tests.

3.2.3.2.3. Découverte de navigations proches et parallèles

Trouver les groupes de navires qui se déplacent ensemble dans un voisinage proche peut être intéressant pour la découverte de risques de pêche parallèle, d'abordage ou de trafic illicite. Des motifs spatio-temporels de trajectoires proches et parallèles peuvent permettre la découverte de ces groupes de navires et leur durée de déplacement ensembles. Ces motifs sont des clusters spatiaux reliés successivement pendant une durée de temps par un seuil d'objets en communs. Plusieurs méthodes de clustering d'objets mobiles existent dans la littérature comme Swarm, Flock et Convoy (Cf. section 2.2.3.1.4 du chapitre 2).

[75] http://dm.kaist.ac.kr/jaegil/sources/

Notre choix s'est porté sur l'algorithme Convoy par ce qu'il permet de détecter les objets qui se déplacent ensemble pendant une durée de temps minimale. Cela est en adéquation avec notre problématique du fait qu'il est impossible de savoir par avance les durées de temps passées par des navires à naviguer à proximité. De plus, il se base sur la notion de densité ce qui nous permet de détecter des convois ayant des formes arbitraires.

Le programme de l'algorithme Convoy a été mis à notre disposition par J. Hoyoung qui est l'un de ses auteurs. Des modifications ont été apportées à cette implémentation pour l'intégrer à ShipMine. Les valeurs des paramètres m, k, e, δ (Cf. section 2.2.3.1.4) ainsi que le chemin des données à explorer et ceux des résultats sont fournis en argument de l'appel du programme. Après exécution, L'algorithme Convoy génère trois fichiers. Le premier, contient des métriques sur l'exécution à savoir, les valeurs de paramètres, le temps d'exécution, les convois candidats et les convois actuels. Le deuxième fichier comporte les convois actuels et le troisième, les trajectoires simplifiées par l'algorithme DP* (Cf. section 2.2.3.1.4). Le deuxième fichier est exploité par ShipMine pour afficher les convois résultats.

3.2.4. Structure de l'espace de données à explorer

Dans une approche de fouille de données, une phase d'acquisition des données à explorer est nécessaire. Les données à explorer sont de deux catégories : les données spatiales statiques et spatiales dynamiques.

3.2.4.1. Les données spatiales statiques

Ce sont des données décrivant des événements ponctuels à risques qui se sont produits à une date et à une localisation données. Ces données sont structurées comme suit :

<Événement (var_1,…,var_n), Timestamp, Localisation (Latitude, Longitude)>

- *Événement (var_1,…,var_n)* : Décrit un événement par un ensemble de variable comme le type d'événement, les objets concernés, le contexte et l'environnement,
- *Timestamp* : La date et l'heure à laquelle l'événement s'est réalisé,
- *Localisation (Latitude, Longitude)* : la localisation absolue de l'événement.

Ces données peuvent être par exemple des données décrivant des accidents maritimes ou des actes de piraterie.

L'exploration de cette catégorie de données, peut permettre la découverte de connaissances sur des situations à risques qui peuvent être soit des relations entre les valeurs, des variables décrivant un événement, ou des zones à forte densité des événements étudiés.

3.2.4.2. Les données spatiales dynamiques

Ce sont des données historiques recensant les déplacements d'objets. La structure de ces données est définie comme suit :

<Objet (var_1,…,var_n), Localisation <loc1,…,locm>, Timestamp <t1,…tm >>,

- *Objet (var_1,…,var_n)* : Décrit un objet mobile par un ensemble de variable comme le type d'objet, sa vitesse et l'environnement d'évolution,
- *Timestamp* : La date et l'heure à laquelle la donnée a été acquise,
- *Localisation (Latitude, Longitude)* : La localisation absolue de l'objet à l'instant *Timestamp*.

Ces données peuvent être des traces AIS ou radar. L'exploration de cette catégorie de données par des méthodes de fouille de données d'objets mobiles peut permettre l'extraction de motifs de mouvements décrivant des comportements à risques.

Les données spatiales statiques et dynamiques qui ont été acquises pour le test et la validation de notre méthode sont présentées dans le chapitre suivant (Cf. section 4.2.1).

3.2.5. Architecture

Dans notre architecture représentée sur la Figure 3-4, trois interfaces sont distinguées : une interface de visualisation ; une interface de fouille de données et ; une interface des données à explorer. Cette architecture a été pensée d'une manière extensible pour pouvoir accueillir d'autres fonctionnalités.

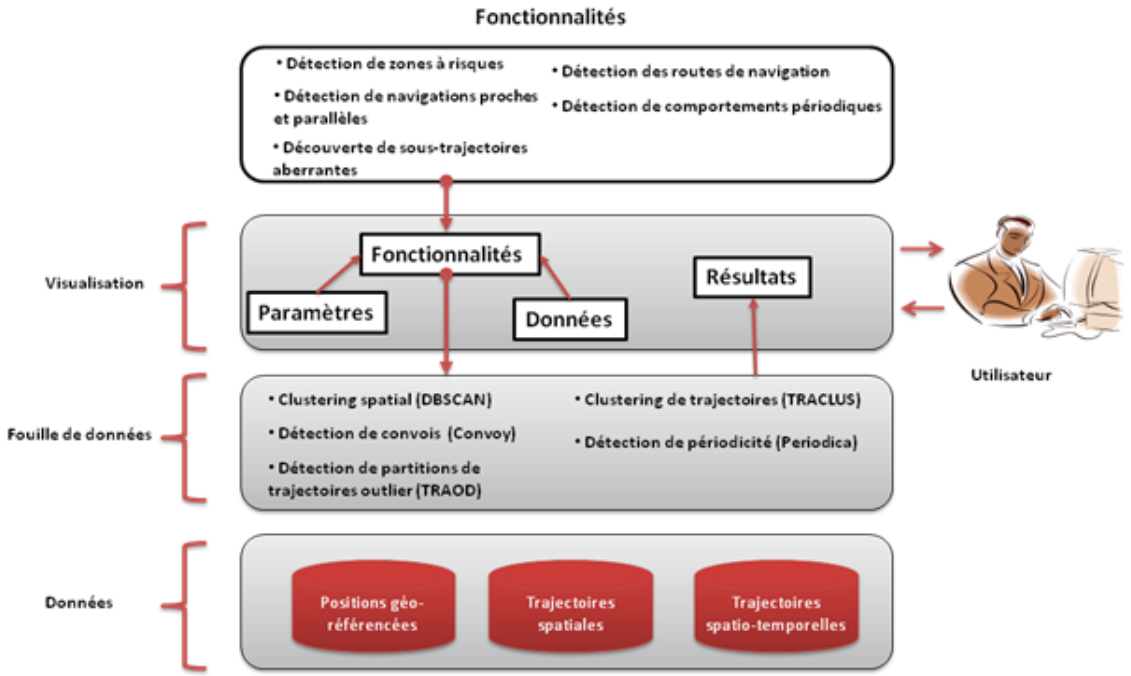

Figure 3-4 : Architecture système de ShipMine.

L'architecture de ShipMine à été inspirée de l'architecture proposée dans MoveMine (Li et al., 2011).

3.2.5.1. Interface de visualisation

L'utilisateur de l'atelier peut interagir avec l'interface de visualisation pour choisir une fonctionnalité, sélectionner des données, visualiser ces données sur une cartographie, entrer les valeurs des paramètres, exécuter une fonctionnalité et interagir avec les résultats affichés sur la cartographie (sélectionner, zoomer, afficher des informations complémentaires, etc.).

Pour comprendre le fonctionnement de ShipMine et ses interactions avec l'utilisateur, un diagramme de cas d'utilisation UML[76] de l'interface *Visualisation* est représenté sur la Figure 3-5. Ce diagramme montre une vue générale et simplifiée des cas d'utilisations de ShipMine et leur relations d'inclusion et d'extension

.

[76] Unified Modeling Language est un langage de modélisation graphique utilisé dans le développement logiciel.

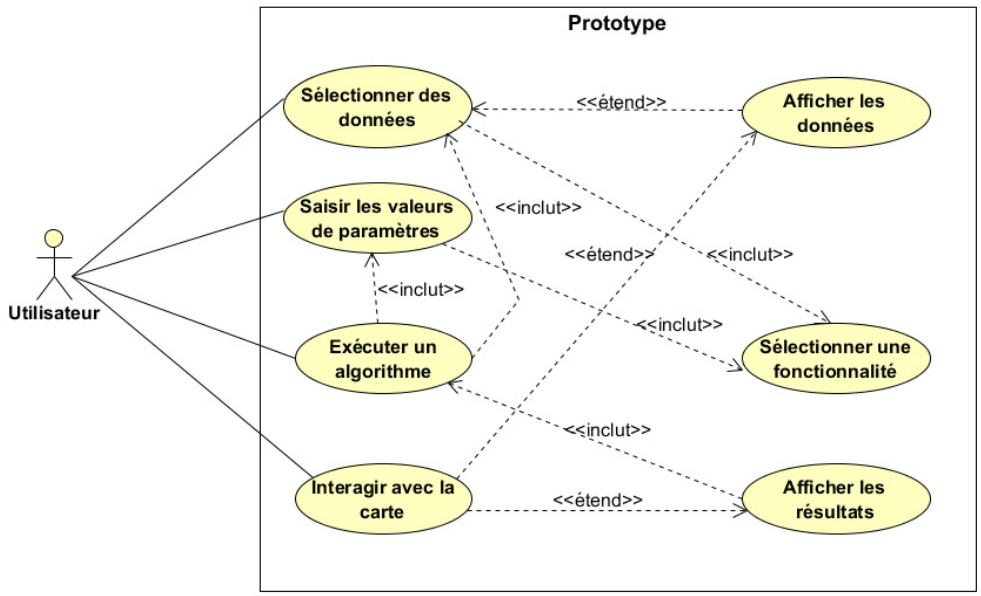

Figure 3-5 : Diagramme de cas d'utilisation de l'interface Visualisation de ShipMine.

Il est à noter sur ce diagramme (Figure 3-5) que le cas d'utilisation *Sélectionner une fonctionnalité* inclut forcément celui de la sélection des données et la saisie des valeurs des paramètres. Ces deux cas à savoir, *Sélectionner des données* et *Saisir les valeurs de paramètres* sont inclus à leur tour dans le cas *Exécuter un algorithme*. Concernant le cas d'utilisation *Interagir avec la carte*, l'utilisateur peut optionnellement afficher les données et/ou les résultats pour interagir avec eux.

L'interface *Visualisation* est l'entrée du système. C'est à partir de cette interface que l'utilisateur peut communiquer avec les composantes du système, se trouvant dans différentes interfaces.

3.2.5.2. Interface de fouille de données

L'interface *Fouille de données* est le cœur du système. Elle contient un ensemble d'algorithmes implémentés, adaptés et intégrés au système. Quand une fonctionnalité est choisie par l'utilisateur, cette interface est sollicitée pour exécuter l'algorithme associé.

L'exécution d'un algorithme dans ShipMine passe par une séquence chronologique d'actions. Cet enchaînement d'actions est représenté sur la

Figure 3-6 par un diagramme de séquence UML.

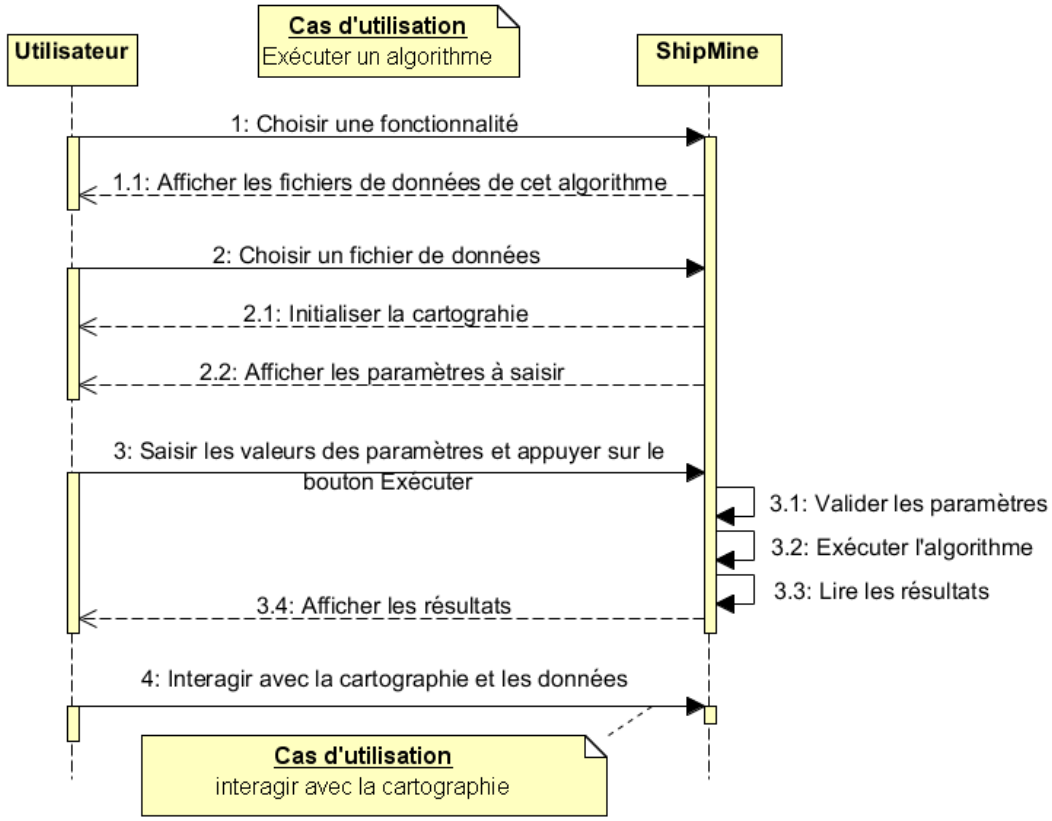

Figure 3-6 : Diagramme de séquence des interactions au cours de l'exécution d'un algorithme.

Quand l'utilisateur choisit une fonctionnalité, ShipMine lui associe un algorithme et affiche une liste de jeux de données spécifiques à cet algorithme. Chaque algorithme utilise un format particulier des données en entrée.

L'utilisateur choisit un jeu de données à explorer et le système initialise la cartographie, affiche le contenu de ce jeu de données et les champs des paramètres à saisir. L'utilisateur est libre d'interagir avec ce jeu de données initiales (zoomer, afficher des informations complémentaires, etc.) ou exécuter la fonctionnalité. Si le bouton *Exécuter* est pressé, le système valide les valeurs des paramètres saisies, appelle le

programme associé et affiche les résultats de l'exécution pour que l'utilisateur puisse les étudier en interagissant avec la cartographie.

La représentation d'un cas d'utilisation d'une interaction avec la cartographie est présentée par un diagramme de séquences UML (Figure 3-7). La première séquence du diagramme représente la réponse de l'API[77] Google Maps aux actions de l'utilisateur comme le clic, le double clic et le changement de type de cartographie (plan, satellite, relief, etc.). Ces actions sont implantées par défaut dans l'API Google Maps qui va être présentée plus en détail dans la section 3.3.

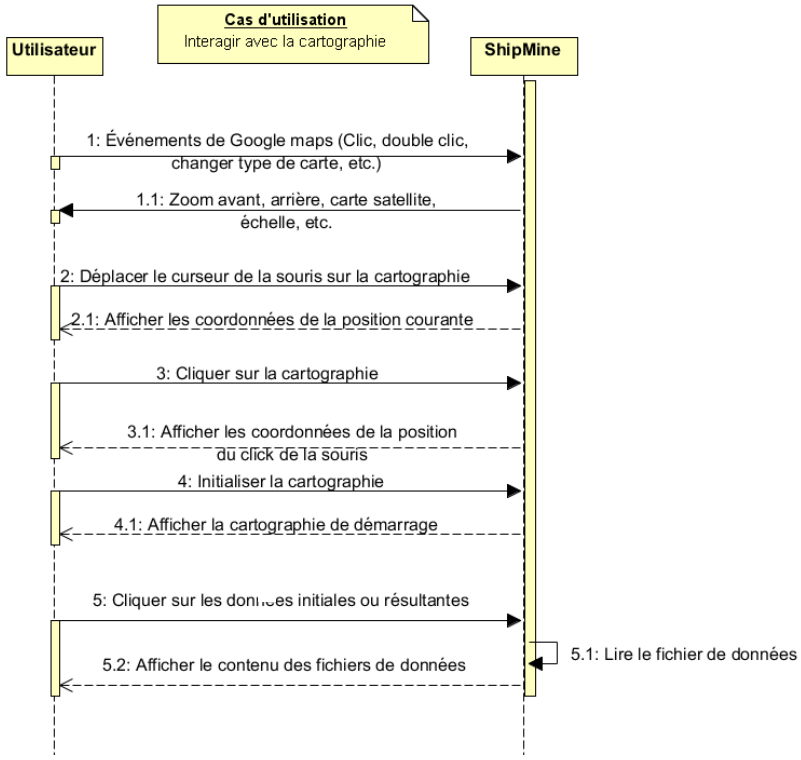

Figure 3-7 : Diagramme de séquence des interactions de l'utilisateur avec la cartographie.

En plus des actions implantées par défaut dans l'API Google Maps, d'autres sont ajoutées à ShipMine concernant l'affichage de coordonnées géographiques du déplacement de la souris, l'affichage de coordonnées de positions cliquées, l'affichage de nos données, de nos motifs sur cette cartographie ainsi que d'autres informations attributaires.

[77] Application Programming Interface est une interface de programmation contenant un ensemble de classes, méthodes et fonctions qu'il est possible d'utiliser dans de nouveaux programmes.

3.2.3.3. Interface de données à explorer

Concernant l'interface de données de l'architecture ShipMine, elle comporte des données sur des déplacements de navires et des événements maritimes passés. Ces données sont récupérées, traitées et transformées selon l'exigence des algorithmes de fouille de données qui vont les utiliser. Chaque algorithme peut avoir un format et un contenu de données spécifiques.

Dans cette interface, il est possible de distinguer trois types de données : positions géo-référencées, les trajectoires spatiales et les trajectoires spatio-temporelles. Les trajectoires spatiales sont composées de séquences de positions géo-référencées et les trajectoires spatio-temporelles sont des trajectoires spatiales où le temps de chaque position est renseigné.

Cette typologie identifie la nature des données et par conséquent les méthodes de fouille de données qui peuvent être utilisées.

3.2.6. Choix technologique

Nous avons opté pour une application web car elle permet d'avoir une architecture client-serveur avec un accès client léger. Le développement de l'application a été réalisé lors d'un stage que nous avons encadré (El Moussawi, 2013). L'utilisateur n'a rien à installer sur son ordinateur et il n'a besoin que de son navigateur web pour envoyer des requêtes au serveur et afficher les résultats sous forme de pages web. Toutes les interactions avec le navigateur sont transformées en requêtes. L'atelier est composé de pages web stockées sur un serveur web de type Apache et jouant le rôle d'interface entre le client et les implémentations des algorithmes. Les échanges entre le client et le serveur se font par le protocole HTTP. Nous n'avons pas utilisé un protocole sécurisé car les données qui circulent entre le client et le serveur sont rendues anonymes et peuvent donc transiter en claire sur le réseau.

Pour représenter les pages web, le langage HyperText Markup Language (HTML) est utilisé. L'acronyme HTML signifie que le langage permet de traiter des hypertextes en se basant sur un langage de balisage. Les balises permettent de mettre en forme les textes affichés sur la page web et d'afficher des éléments interactifs comme les boutons, les images et les liens hypertextes. Pour gérer les interactions côté client et avoir des pages

dynamiques, le langage de programmation JavaScript est utilisé. Ce langage apporte des améliorations à l'HTML en permettant d'exécuter des commandes au niveau client, c'est-à-dire par le navigateur web du client. L'affichage des résultats d'exécution des algorithmes de l'atelier se fait en utilisant JavaScript. Dans le développement de ShipMine, nous avons choisi d'utiliser le PHP pour l'appel des exécutables binaires des algorithmes qui composent l'atelier et lire les résultats des exécutions à partir de fichiers. Le PHP[78] est un langage de programmation très utilisé pour la création de pages web dynamiques. Ce qui a motivé ce choix est le fait qu'il permette au serveur de répondre au client sans dévoiler le contenu du code source exécuté. De plus, la version 5 de de PHP utilise la programmation orientée objet ce qui permet une réutilisation plus souple des classes d'objets.

Une autre technologie utilisée comme partie intégrante de l'IHM (Interface Homme Machine) de ShipMine est l'API Google Maps. Cette API permet de gérer l'affichage et la manipulation des objets cartographiques. Elle contient une bibliothèque de fonctions et de classes permettant de gérer des objets géographiques et des rasters dynamiquement à partir d'une interface web. Les services de base intégrés dans cette API sont par exemple le contrôle du zoom, le déplacement dans la carte, le changement d'échelle, le changement du type de carte, l'affichage des info-bulles, l'affichage des objets géométriques (positions, polylignes, polygones), le contrôle des événements ou des actions des utilisateurs sur la carte (superposition de plusieurs couches, etc). Les positions des navires vont être représentées dans Google Maps par des marqueurs, les trajectoires par des polylignes et les zones par des polygones. Les informations complémentaires sur les objets cartographiques vont être affichées dans des info-bulles.

Les programmes qui ont été choisis pour intégrer notre atelier sont développés la plupart du temps en Visual C++, C++ et Java. Le choix de ces technologies de développement s'est imposé par souci de réutilisation des codes sources existants.

3.3. Présentation de ShipMine

Après avoir sélectionné les algorithmes et préparé les données à utiliser, leur intégration dans l'atelier ShipMine a été opérée de manière à pouvoir ajouter d'autres

[78] http://php.net/docs.php

algorithmes et d'autres données à notre atelier. Les fichiers de données qui peuvent être analysés sont ceux déjà intégrés au préalable au système. Par souci d'intégrité des données, l'utilisateur de ShipMine ne peut pas analyser d'autres données sauf si l'administrateur les intègre dans l'un des répertoires de données créés à cet effet et paramètre le système pour qu'il prenne en compte ces données.

Comme les données brutes, les motifs résultants peuvent être aussi représentés par une typologie contenant trois formes géométriques (Figure 3-8) : un point, un polygone et une polyligne.

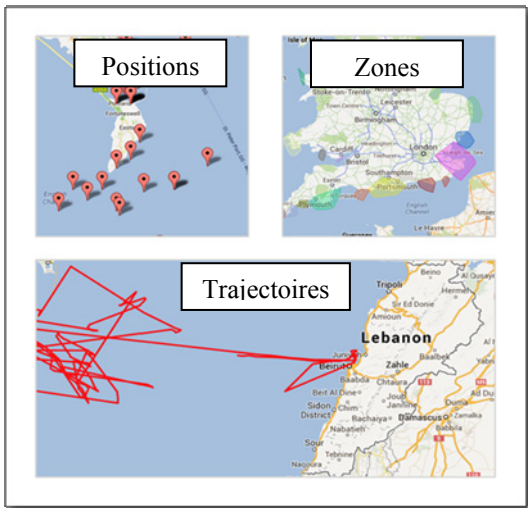

Figure 3-8 : Typologie de formes géométriques utilisée pour la représentation des données et des motifs dans ShipMine.

L'interface de ShipMine est composée de quatre cadres comme on le voit sur la Figure 3-9. Le premier est l'entête, il contient la bannière ; le deuxième est composé de deux listes déroulantes permettant de choisir quelles fonctionnalités et données utiliser ; le troisième cadre quant à lui concerne la saisie des valeurs de paramètres et le lancement de l'extraction et ; le quatrième et dernier cadre est l'interface cartographique Google Maps permettant d'afficher et d'interagir avec les données.

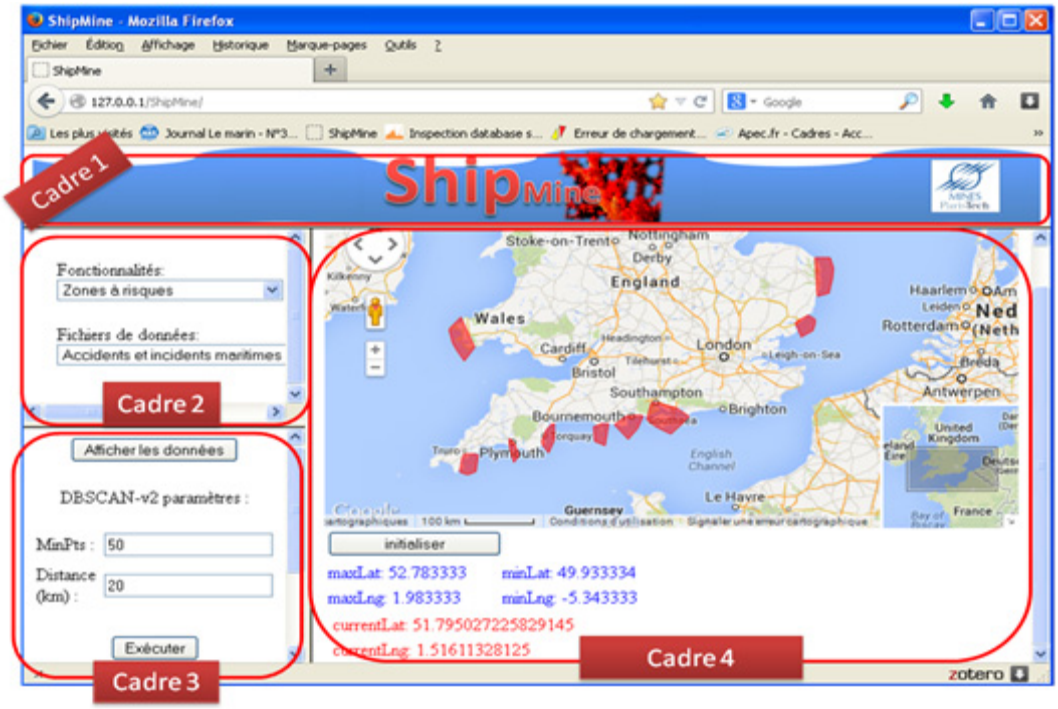

Figure 3-9 : Interface principale de ShipMine avec ses différents cadres.

Les différents cadres de l'interface sont gérés par des fichiers HTML et PHP permettant de faire des tâches spécifiques. Le fichier index est le premier fichier qui s'exécute pour initialiser l'interface. Ce fichier appelle par la suite, les autres fichiers qui sont le Header.html, Algorithme-data.php, Paramètres.php et Cartographie.php. Nous allons détailler par la suite chaque cadre de notre interface.

3.3.1. Bannière

Dans ce cadre, la bannière graphique est chargée à partir d'une image présentant le nom de l'atelier ainsi que le logo de MINES ParisTech.

3.3.2. Choix de la fonctionnalité et des données

Ce cadre est constitué de deux listes déroulantes. La première liste l'ensemble des fonctionnalités qu'il est possible d'utiliser et la deuxième les données qui sont liées à la fonctionnalité choisie (Figure 3-10). Ces listes sont créées dynamiquement au cours de l'initialisation de l'application en se basant sur un fichier de paramètres. Cela permet à l'atelier de rester évolutif en gardant la possibilité de changer la liste des fonctionnalités

en enlevant ou ajoutant d'autres fonctionnalités sans toucher au code source. La liste des données quant à elle est récupérée à partir du répertoire de données spécifié dans le fichier paramètres appelé « algorithmes implémentes.txt ». Pour ajouter une autre fonctionnalité, il suffit de rajouter une ligne dans ce fichier contenant, le nom de la fonctionnalité, les paramètres de l'algorithme associé à la fonctionnalité et le chemin du répertoire contenant les données.

Figure 3-10 : Cadre du choix d'une fonctionnalité et des données.

3.3.3. Fonctionnalité d'exécution et de paramétrage

Dans ce cadre d'interface, il est possible d'afficher les contenus des fichiers de données, de saisir les paramètres et d'exécuter l'exploration des données (voir Figure 3-11). Le programme associé à la fonctionnalité sélectionnée est exécutée avec les valeurs des paramètres transmises par le système. Des contrôles de vérification et de validation des paramètres saisis sont intégrés au système. Un message d'erreur est retourné si les paramètres saisis sont erronés. Prenons l'exemple du paramètre minimum de positions pour former un cluster (MinPts) de l'algorithme DBSCAN (Cf. section 2.2.1.3). La valeur de ce paramètre doit impérativement être entière et supérieure à deux.

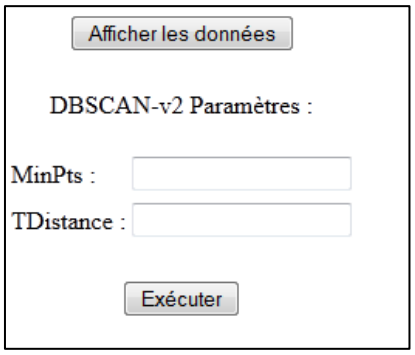

Figure 3-11 : Cadre paramètres et exécution de l'exploration.

Le cadre *Paramètres* dépend de la fonctionnalité choisie. En effet, avant de choisir la fonctionnalité, le cadre est vide. Quand l'utilisateur sélectionne une fonctionnalité, le système lit le fichier « algorithmes implémentes.txt » (Cf. section 3.3.2) pour récupérer dynamiquement les noms des paramètres associés à la fonctionnalité.

Les résultats de l'exploration sont issus d'un fichier d'échange entre ShipMine et le programme exécuté. Ces résultats vont être affichés dans le cadre de l'interface cartographique comme nous allons le voir dans la section suivante.

3.3.4. Interface cartographique

Ce cadre intègre une interface cartographique Google Maps utilisée pour l'affichage des données, des résultats et l'interaction entre eux. Cette interface cartographique est liée aux autres cadres et les changements effectués par l'utilisateur, les données en entrée ou les paramètres sont visualisés dans ce dernier cadre. Il est possible d'interagir avec les données et les motifs affichés et de donner à l'utilisateur des détails sur les éléments cliqués. Prenons deux exemples d'affichage d'informations, le premier est un clic sur une trajectoire pour afficher un info-bulle indiquant le numéro MMSI du navire et le nombre de positions de la trajectoire (Figure 3-12-a) et le second exemple est un info-bulle sur la localisation de la position et le type d'accident (Figure 3-12-b).

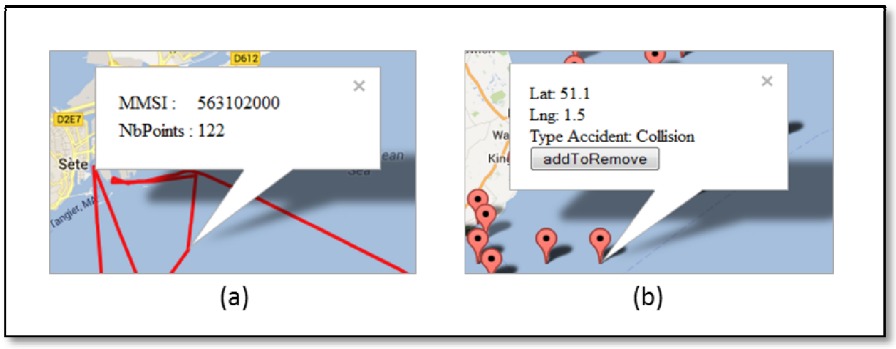

Figure 3-12 : Affichage d'informations attributaires sur des éléments de données (a) trajectoire (b) position.

Deux composants ont été ajoutés sur le bas du cadre cartographique pour permettre l'initialisation de la carte et la récupération d'informations de localisation comme la position du curseur et la position d'un clic de souris sur la cartographie. Les valeurs minimales et maximales de la latitude et longitude (*minLng* et *maxLng*) sont récupérées à partir du fichier de données et affichées dans le même cadre avec une couleur bleu. Les latitudes et longitudes courantes (*currentLat* et *currentLng*) dues au déplacement de la souris sur la carte sont par contre affichées en rouge. Les coordonnées affichées en vert représentent la position du dernier clic de souris sur la cartographie (*clickedLat* et *clickedLng*). Toutes ces petites fonctionnalités vont permettre d'analyser les données et les résultats affichés sur la cartographie.

L'initialisation est accessible à l'utilisateur par le biais du bouton *« Initialiser »*. Comme on le voit sur la Figure 3-13, un autre bouton a été ajouté pour permettre de

supprimer des positions dans le fichier de données originales. A la fin du nettoyage, l'utilisateur peut sauvegarder les données restantes dans un nouveau fichier. Le fichier est mis dans le même répertoire que le fichier original avec une dénomination clean devant l'ancien nom du fichier.

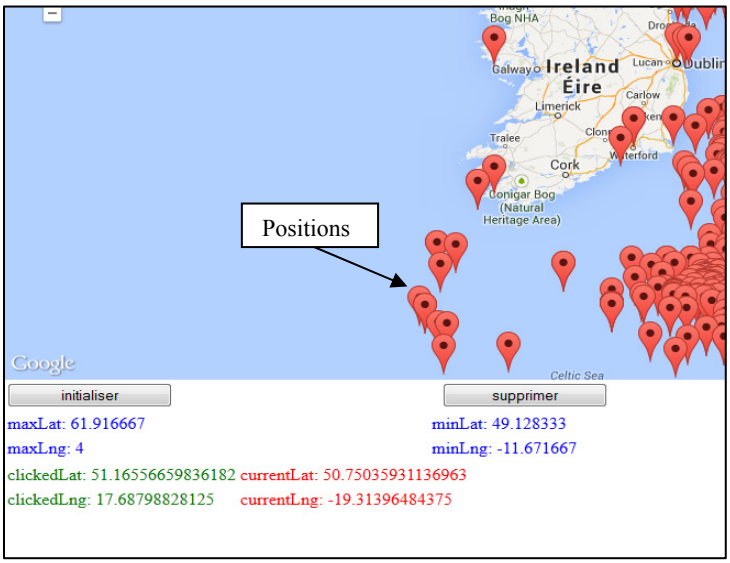

Figure 3-13 : Cadre cartographique de l'interface de ShipMine.

3.4. Conclusion

Dans l'objectif de valider notre méthodologie d'aide à l'analyse des comportements à risques, nous avons identifié et testé des méthodes de fouille de données susceptibles de décrire ces comportements. La fouille de données n'a pas été conçue pour extraire les comportements à risques mais elle permet d'extraire des motifs de mouvements, de situations (anormaux, fréquents, etc.) ou leurs combinaisons qu'il est possible d'interpréter comme étant à risques.

Des méthodes de fouille de données pouvant extraire des situations et mouvements à risques ont été intégrées dans un seul environnement pour servir d'atelier d'extraction de connaissances sur les comportements potentiellement à risques. Les méthodes qui ont été choisies pour extraire ces connaissances sont la détection des associations, la détection des zones à risques, la découverte de convois de trajectoires, la détection de trajectoires

aberrantes et la découverte de trajectoires type. Les étapes suivies pour concevoir et réaliser cet atelier (ShipMine) ont été présentées dans ce chapitre.

Pour tester et valider nos méthodes dans ce contexte d'analyse de comportements à risques, des motifs et règles décrivant les comportements à risques doivent être extraits en utilisant ShipMine. Les résultats de cette exploration vont être présentés et discutés dans le chapitre suivant.