

Table des matières

1	Préambule	2
1.1	Principes généraux	2
1.2	Mode de fonctionnement	3
1.3	Architecture générale de l'application ARIA3	3
1.3.1	Fonctionnalités générales	3
1.3.2	Principes des échanges et communications avec le serveur	4
2	Contraintes techniques	4
3	Principes généraux de l'application	5
3.1	Fonctionnement MVC	5
3.2	Paramètres de l'application	5
3.2.1	Paramètres généraux	5
3.2.2	Paramètres spécifiques métier.....	6
3.3	Gestion des modèles basés sur les données	6
3.3.1	Principes généraux	6
3.3.2	Référence du modèle de données	7
3.3.3	Structure générale d'un modèle de données.....	7
3.3.4	Propriétés standards pour les champs.....	8
3.3.5	Champs standards	9
3.3.6	Conception et place d'un modèle de données dans l'application	9
3.3.7	Contraintes de nommage des champs.....	9
3.3.8	Formats de champs et paramètres associés	10
3.3.9	Synthèse des formats par type de champs	14
3.4	Administration des modèles de données.....	14
3.4.1	Représentation « Données »	15
3.4.2	Représentation « Modèle ».....	15
3.5	Modèles métier hors données	16
3.6	Gestion des vues.....	16
3.6.1	Gestion des vues en mode CRUD	16
3.6.2	Gestion des vues en mode « Masques »	17
3.6.3	Gestion des vues en mode « WYSIWYG »	18
3.7	Gestion des règles	20
4	Requêteur générique	20

	Document de conception technique	Annexe 3 CCTP
	ARIA3	Date : 05/06/2026 Version : 7

5	Modèle générique et Contraintes de pérennité	21
5.1	Applications connexes	21
5.2	Contraintes de sécurité et accès	22
6	Contexte technique du site Web ARIA3	22

1 Préambule

1.1 Principes généraux

La mise en œuvre de l'application ARIA3, dont la mise en production a eu lieu en mars 2014, avait pour objectif de répondre à un certain nombre de besoins exprimés dans le cadre de la refonte du logiciel ARIA2.

L'une des principales exigences reste la maintenabilité de l'application sur le long terme, que ce soit l'ajout d'un champ, la refonte d'un formulaire, la modification de règles métier de gestion des dépendances entre les champs...

Ces actions doivent pouvoir être réalisées sans l'intervention d'un développeur informatique.

Dans le cadre de la rationalisation des développements souhaitée pour ARIA3, il apparaît que pour apporter une réponse à l'ensemble de ces besoins, il est nécessaire de centraliser les paramètres relatifs aux champs : leur position dans les masques, les liens entre les champs...

L'application ARIA3 peut répondre à ce besoin en permettant :

- à un administrateur de définir visuellement, depuis une interface d'administration, un ensemble de masques de saisie ainsi que les règles logiques reliant les champs de ces masques ;
- à un utilisateur d'afficher les masques de saisie créés par l'administrateur et de les renseigner afin de créer de nouveaux enregistrements en base de données.

En ce sens, l'application ARIA3 constitue un outil universel permettant de générer des formulaires, dont les principes de fonctionnement sont décrits dans l'annexe 2.

En pratique, ces principes généraux ont permis de faire évoluer l'application depuis sa mise en production jusqu'à la rédaction de la présente version de ce document :

- création de la vue principale « Evénements industriels », reprenant les grands principes de la vue initiale de la précédente version de ARIA (ARIA2) ;
- ajout du contexte « CATNAT » pour la saisie et recherche d'événements liés à des catastrophes naturelles (plus usité à ce jour) ;
- développement d'une application parallèle pour la saisie des événements de transport de matières dangereuses (TMD) (plus usitée à ce jour) ;
- ajout d'une vue réduite BEA-RI, pour un accès direct aux informations essentielles de l'évènement ;

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

- ajout de la notion de fiche parapheur, intégrant un système de workflow, destinée à la collecte des pièces et informations pour l'ensemble des événements dont le BARPI a connaissance.

1.2 Mode de fonctionnement

L'application s'appuie sur un modèle relationnel, lui-même géré par l'intermédiaire d'un panneau d'administration spécifique.

Les actions de cette application sont donc transverses :

- manipulation et description du modèle relationnel (collections, champs) ;
- manipulation directe des données par un modèle CRUD (Create Read Update Delete) ;
- manipulation des vues par un modèle CRUD adapté et par une interface de prévisualisation permettant de positionner les champs (WYSIWYG) ;
- exécution d'une vue afin de visualiser et de manipuler les données d'une collection.

1.3 Architecture générale de l'application ARIA3

1.3.1 Fonctionnalités générales

L'application ARIA3 est structurée en un ensemble de processus client-serveur, gérant les actions de génération de formulaires, automatismes sur les données, actions de maintenance..., piloté par des vues gérant les différentes actions des utilisateurs de l'application. Côté client, les vues communiquent avec le serveur via des appels asynchrones (AJAX), permettant d'exécuter des calculs, réaliser des enregistrements ou de rendre des portions de page.

L'application ARIA3 est conçue de manière globale afin de faire appel aux outils optimums selon l'affichage demandé par l'utilisateur :

- page d'accueil de l'outil : panneau de contrôle, présentant les données essentielles ;
- préférences : panneau de préférences de l'utilisateur (règles d'alertes, préférences des éditions...);
- modélisation : outil de modélisation, faisant lui-même appel aux services de génération de masques adaptés à la vue « **Modélisation** » ;
- requêteur : outil de création de requêtes, réalisation d'extractions de données, affichage cartographique pour une requête donnée ;
- cartographie : système d'affichage géographique des données d'accidentologie, permettant de superposer des données issues de plusieurs requêtes ;
- parapheur : collecte des informations sur les événements avant décision de les retenir ou non dans ARIA ;
- échéancier : planification des échanges avec les tiers (rappels par mail, consultations...);
- téléprocédures : import des téléprocédures saisies sur le portail numérique de télédémarche ;
- photomanager : pour la gestion des médias (images, vidéos) associés aux accidents ;
- administration : lexiques et tables simples : affichage des données en mode CRUD ;
- tickets : **outil tiers** ARIA Suivi (*Jira de Atlassian*) ;
- gestion de sources : via aussi un outil tiers, GITLAB ;
- recherche d'accidents et masques : fonction de recherche en vue « Masques ».

	Document de conception technique	Annexe 3 CCTP
	ARIA3	Date : 05/06/2026 Version : 7

1.3.2 Principes des échanges et communications avec le serveur

Pour l'ensemble de l'application, les pages sont générées côté serveur puis transmises directement au client Web (principe de base d'une application PHP).

Des appels asynchrone (AJAX) en mode client-serveur sont réalisés pour générer différents types de rendus interactifs, notamment :

- les vues de type « Masques », affichant des masques conçus par les utilisateurs métier ;
- les vues de type « Modélisation » représentant l'accident de manière visuelle ;
- la vue en accès direct en mode simple (CRUD) destinée à la maintenance des données techniques ;
- l'affichage dynamique des modules présents dans la colonne latérale de compléments à la saisie.

Lors de la saisie de données dans les masques d'évènements, les processus suivants sont suivis :

- les données sont transmises au serveur masque par masque, uniquement en cas de modification de données ;
- lors de chaque sauvegarde, des traitements automatiques sont réalisés en suivant les consignes décrites dans les tables de configuration de l'application :
 - vérification de l'état du workflow ;
 - vérification de modifications dans l'échéancier ;
 - exécution des règles et automatismes métier de mise à jour des champs côté serveur.

2 Contraintes techniques

Pour des raisons de pérennité de l'application et de compétences internes, les technologies utilisées pour le projet doivent être le plus en phase possible avec les développements antérieurs, un changement technologique ne doit pas nécessiter d'apprentissage lourd pour être mis en œuvre.

Les briques technologiques « serveur » principales mises en œuvre dans ARIA3 sont :

- Serveur web Apache2
- Langage de programmation : PHP version 8.3 ou +
- Base de données : MongoDB v4.4 ou +
- Moteur d'indexation : SOLR 9+

L'utilisation de l'outil ARIA3 nécessite l'utilisation d'un navigateur Web moderne supportant les dernières normes de technologies (HTML, Javascript, CSS, Websocket...).

3 Principes généraux de l'application

3.1 Fonctionnement MVC

L'application ARIA3 est construite autour d'un cœur de Framework MVC (Modèle / Vue / Contrôleur) conçu spécialement pour ce projet, pour fonctionner en phase avec la technologie de base de données MongoDB.

Le MVC de ARIA3 a été conçu pour être évolutif et proposer une évolutivité simple basée sur les spécificités du métier.

Pour l'affichage de chacun des différents contenus de l'application, une URL spécifique est forgée selon le principe d'un routage automatique basé sur les composants actifs disponibles.

Les routes du MVC sont calquées sur les contrôleurs existants et permettent l'accès à des composants ou des modules internes, afin que ces derniers servent de micro-contrôleurs pour le rendu de sous-éléments de la page, tels que les blocs dans la colonne latérale de compléments présente dans les différentes pages de l'application.

Un système de composants permet donc d'étendre les contrôleurs de base par des sous-routes apportant les fonctionnalités principales pour les contrôleurs. Exemple : le contrôleur de saisie est secondé par des composants de rendu en maquette ou en modélisation.

Un contrôleur spécifique permet l'intégration de modules transverses aux différents contrôleurs et apportant des fonctionnalités complémentaires : prises de notes, import-export, recherche spécifique, accès aux documents et médias joints...

L'interaction client/serveur est réalisée via des sous-routes automatiques offrant des accès à l'API via des appels AJAX systématisés sur les composants et le système de modules.

Contrôleurs, composants et modules ont un fonctionnement type MVC chacun à leur niveau.

3.2 Paramètres de l'application

3.2.1 Paramètres généraux

Pour l'exploitation de l'outil, un certain nombre de paramètres applicatifs sont indispensables : nom et accès à la base de données évidemment, mais aussi chemins de stockage des données, options par défaut...

Les paramètres généraux sont sauvegardés dans la collection « adm_configurations ». Exemple de paramètres principaux :

Paramètre	Valeur par défaut	Définition
full_date_format	d/m/Y H:i:s	Format d'affichage par défaut date & heure
date_format	d/m/Y	Format d'affichage par défaut date seule
date_format_saisie	dd/mm/yy	Format de saisie des dates
nom_fichier_modele	(...)	Nom & localisation du modèle freemind (dev)
encodage	utf-8	Encodage des textes de l'application
editor_toolbars	(tableau)	Liste les codes autorisés pour les barres d'outils
data_list_items_per_page	30	Nombre de lignes par page pour les tableaux
champs_modele_valeur		Code HTML dans lequel la valeur est insérée

champs_modelle_formulaire	Code HTML dans lequel le formulaire est inséré
champs_modelle_	Code HTML complet contenant valeur et
form_complet	formulaire

L'application dispose d'environ 150 paramètres techniques stockés sous des formats variés : scalaire (entier, texte ...) ou objet.

3.2.2 Paramètres spécifiques métier

Pour répondre aux paramétrages complexes de certaines fonctionnalités, des groupes de réglages sont mises en place, notamment :

- consultations : organisation des destinataires et des modèles de mails pour les envois de consultations manuelles bimestrielles ;
- échéanciers : réglages des modèles d'échéanciers, logs de mails, destinataires, listes de référence, pour la gestion des étapes des échéanciers automatiques dans les événements ;
- workflows (définition, transitions, listes) : liste des étapes des workflows, actions automatiques et manuelles, description des listes d'évènements en fonction du statut...
- Webservices : liste des webservices entrants et sortant, définition des API, données de réglages pour les échanges automatisés avec les outils tiers (GUNenv...).
- types des pièces jointes : paramétrage des types de pièces jointes pour l'ajout de documents aux événements.

3.3 Gestion des modèles basés sur les données

3.3.1 Principes généraux

La *structure* du modèle de données global de l'application ARIA3 est décrit via un fichier Freemind. Ce modèle, lorsqu'il est modifié, est réimporté dans l'application afin de répercuter les changements de structure. Il est constitué de 169 collections organisées en neuf grands groupes principaux :

- Données : les collections contenant des données produites par les utilisateurs finaux ;
- Administration : les réglages liés à l'administration, préférences générales, logs ;
- Utilisateurs : gestion des utilisateurs, droits, groupes, droits d'accès aux webservices ;
- Réglages métier : réglages généraux de l'application, configurations, types de PJ, gestion des codes OTP... ;
- Consultations : destinataires des mails de consultations manuelles, logs d'envoi, modèles ;
- Echéanciers : règles métier, logs spécifiques, modèles d'échéances, destinataires des mails... ;
- Lexiques : les référentiels externes à l'application (communes, départements, produits...) ;
- Modélisation des vues : Contient la description des masques et champs pour chacune des différentes vues de l'application (événements, CATNAT, BEA, Parapheur) ;
- Modèles de données : description technique des modèles de données pour chaque collection suite à la conversion du fichier Freemind en un JSON directement exploitable par l'application.

L'interface d'administration de ARIA3 permet de gérer directement les données contenues dans chacune des collections présentes dans ces groupes.

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

3.3.2 Référence du modèle de données

L'évolution du modèle de données est faite exclusivement par l'intermédiaire du fichier de modèle au format « Freemind » : ajout, suppression, modification de champs ou de collections se fait à partir de ce fichier qui, transféré dans l'application ARIA3, permet de mettre en œuvre un nouveau modèle.

En cas de création d'une nouvelle collection, la mise à jour du modèle dans ARIA3 déclenche la création d'un nouveau fichier source de classe métier dans le dossier des modèles métier, s'il n'existe pas encore. L'implémentation de nouvelles méthodes métier nécessite la récupération de ce fichier dans l'environnement de développement et son ajout à la gestion de sources.

3.3.3 Structure générale d'un modèle de données

Toutes les collections décrites dans le fichier Freemind sont, après conversion, stockées dans la collection spéciale « mod_donnees » qui contient l'intégralité des descriptions des modèles de données.

La base ARIA3 étant gérée sous MongoDB, chaque enregistrement d'une collection contient en pratique un objet au format BSON, soit un objet structuré semblable à du JSON mais adapté aux spécificités de MongoDB. Dans le cas des modèles de données, chaque enregistrement de la collection contient la description technique des collections décrites dans le Freemind.

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

Les champs décrivant les collections sont les suivants :

- [_id] Id MongoDB : identifiant créé automatiquement pour définir cette collection
- [author] Auteur : de la dernière modification (initiales)
- [clef] Encodage MD5 du modèle de donnée
- [code] Code : nom exact de la collection en base
- [date_modification] Date : de dernière modification de la structure
- [description] Description : description complète de la collection
- [title] Libellé : Nom en clair de la collection
- [ref] Tableau de Références des tables MySQL d'origine de la donnée
- [version] Numéro de version du modèle relationnel courant
- [properties] : Propriétés d'un modèle de données, c'est-à-dire les champs qu'il contient. Cette propriété est récursive et pour chaque champ elle contient les données suivantes :
 - [type] Type : décrit le contenu exact du champ en base de données. Valeurs possibles : *string, number, integer, boolean, object, array, null, any*
 - [title] Libellé : nom du champ tel qu'il sera utilisé dans les formulaires CRUD, ou dans les vues s'il n'est pas remplacé par un autre libellé
 - [tooltip] Tooltip : information au survol lors de la saisie
 - [description] Description : description complète du champ si nécessaire
 - [format] Format : le format définit la manière dont la valeur du champ doit être traitée. Le format peut être par exemple : **mongoid**, texte, entier, flottant, case à cocher, liste, référence ... la liste complète des formats est définie de manière exhaustive dans ce document.
 - [properties] Propriétés du sous-objet, si le type vaut « object ». Objet contenant les paramètres spécifiques du champ si le format l'impose
 - [index] : la donnée doit-elle être indexée
 - [position_libelle] : position du libellé dans les vues (dessus, gauche etc)
 - [type] : string, array, object, integer ...
 - [vue] => sous-objet indiquant la visibilité de la données dans les différentes vues du modèle CRUD

Les champs du modèle sont donc placés dans un objet [properties], chaque champ est un objet dont le nom correspond au code du champ en base de données, et peut lui-même contenir un nœud [properties] de manière récursive.

3.3.4 Propriétés standards pour les champs

Ces propriétés se retrouvent dans la plupart des champs des collections :

- Type
- Title
- Format

D'autres champs peuvent être présents pour décrire spécifiquement des objets ou tableaux par exemple.

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

3.3.5 Champs standards

Ces champs sont présents dans la plupart des collections. Ils sont décrits ci-dessous selon la forme qu'ils prennent habituellement en base, en tant que modèles de conception :

3.3.5.1 *Champ « _id » de MongoDB*

Ce champ, situé à la racine de chaque collection, est rendu obligatoire par MongoDB. La définition de ce champ dans le modèle permet de s'assurer que le champ soit disponible dans le modèle de données lors de l'utilisation des objets.

Il s'agit d'un champ standard obligatoire dans tous les enregistrements de toutes les collections.

- Nom du champ : `_id`
- Type : object
- Format : mongoid
- Propriétés :
 - Nom : `$id`
 - Type : string

3.3.5.2 *Champ `_self_id`*

Ce champ n'est pas à mentionner dans les descriptions de formats, il est systématiquement actualisé avec une version en chaîne de caractères du champ `_id`.

3.3.6 Conception et place d'un modèle de données dans l'application

Lorsqu'une nouvelle collection est ajoutée dans le fichier de modèles, elle répond à un besoin métier et s'intègre à plusieurs emplacements de l'application.

Un modèle est présent à ces différents emplacements :

- la description du modèle est stockée dans la collection « mod donnees » : cette organisation est décrite en base de données dans la collection générique « mod_donnees » qui contient donc, par définition, un enregistrement pour chaque type de données manipulées. Lors de l'instanciation d'un objet la description du modèle est lue dans cette collection pour construire l'objet ou gérer l'enregistrement des données ;
- la logique métier : ou en d'autres termes les méthodes spécifiques à cette collection. Cette logique métier existe sous forme d'une classe nommée de manière identique à la collection qui lui correspond, dans le dossier « classes/metier/ » ;
- la vue : il peut exister une ou plusieurs « vues » pour cette collection. Pour chaque, un enregistrement existe dans la collection « mod_vues ». Chaque vue propose une description d'un ensemble de formulaires conçus sur mesure et permettant la manipulation des données contenues dans la collection. Cf. le chapitre relatif aux vues.

3.3.7 Contraintes de nommage des champs

Les noms des champs doivent :

- être en minuscules ;
- contenir uniquement des caractères alphanumériques ou des « `_` » (underscore) ;
- débuter par une lettre, les champs débutant par un « `_` » étant considérés privés.

Au sein des formulaires, chaque champ est identifié selon son code et sa position dans l'arborescence de l'objet :

Champ	Id	Nom
nomduchamp: (valeur)	nomduchamp	Nomduchamp
champobjet champenfant: (valeur)	champobjet-champenfant	champobjet-champenfant
champtableau [123] : (valeur)	champtableau-123	champtableau[123]
champtableau [123] : (objet) champenfant	champtableau-123- champenfant	champtableau[123]- champenfant

3.3.8 Formats de champs et paramètres associés

Les formats ci-dessous sont les formats disponibles lors de la conception d'un modèle de données. Les paramètres associés correspondent à des paramètres de même niveau définis spécifiquement si ce format est utilisé.

Le type requis défini dans quel contexte un format peut être appliqué.

3.3.8.1 *Paramètres génériques*

Dans tous les cas, les paramètres associés suivants sont possibles et optionnels :

- [required] : Booléen, par défaut défini à « false », indiquant si le champ est requis.
- [vue] : Objet, possédant trois propriétés :
 - [liste] : numéro d'ordre d'affichage dans les listes CRUD
 - [detail] : numéro d'ordre d'affichage dans les pages de détail CRUD
 - [form] : numéro d'ordre d'affichage dans les formulaires CRUD

Une valeur à 0 pour l'une de ces propriétés indique : ne pas afficher (champ caché).

3.3.8.2 *Mongoid et ParentId*

Le champ est un identifiant Mongoid à créer lors de l'enregistrement. ParentId peut être nul. Il correspond à un enregistrement parent utilisé pour la reconstruction de l'affichage d'un lexique.

- Code du Format : mongoid ou parentId
- Type requis : object
- Paramètres : aucun

3.3.8.3 *Texte*

- Code du Format : text
- Type requis : string
- Paramètres associés :
 - maxLength (optionnel) : nombre de caractères autorisés pour le texte
 - minLength (optionnel) : nombre de caractères minimum obligatoires
 - default (optionnel) : valeur par défaut à mettre dans la case si l'enregistrement n'existe pas

	Document de conception technique	Annexe 3 CCTP
	ARIA3	Date : 05/06/2026 Version : 7

- pattern : (saisir un format de saisie), l'opérateur devra saisir le champ texte selon le modèle de masque indiqué, contrôle réalisé en javascript à l'aide d'une librairie spécifique

3.3.8.4 *Mot de passe*

- Code du Format : password
- Type requis : string
- Paramètres associés :
 - minLength (optionnel) : nombre de caractères minimum autorisés pour le texte
 - double : false/true (défaut : true) présence ou non d'un second champ de saisie pour réaliser une saisie en double du mot de passe. En cas de double saisie un contrôle est inclus

3.3.8.5 *Texte long*

- Code du Format : longtext
- Type requis : string
- Paramètres associés :
 - minLength (optionnel)
 - maxLength (optionnel)

3.3.8.6 *Texte HTML*

- Code du Format : html
- Type requis : string
- Paramètres associés :
 - minLength (optionnel)
 - maxLength (optionnel)
 - toolbar (optionnel) : code de la barre d'outils à utiliser pour cet éditeur. Le code est issu du paramètre général « editor_toolbars »

3.3.8.7 *Entier*

- Code du Format : integer
- Type requis : integer
- Paramètres associés :
 - default (optionnel) : valeur à mettre dans la case si l'enregistrement n'existe pas

3.3.8.8 *Flottant*

- Code du Format : number
- Type requis : number
- Paramètres associés :
 - format (optionnel) : format d'affichage au format sprintf : ex. `%01.2f`

3.3.8.9 *Case à cocher*

Une seule case à cocher indépendante.

- Code du Format : checkbox
- Type requis : boolean
- Paramètres associés :

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

- default (optionnel) : true/false checkbox cochée ou non par défaut

3.3.8.10 Liste « oui/non »

Liste déroulante proposant un choix oui/non.

- Code du Format : listyesno
- Type requis : boolean
- Paramètres associés :
 - default (optionnel) : true (oui) / false(non)

3.3.8.11 Liste à sélection multiple

Une liste (listbox) à sélection multiple, ou plusieurs cases à cocher groupées du même niveau.

- Code du format : listmultiple
- Type requis : array
- Paramètres associés :
 - displayformat : list / checkbox
 - default (optionnel) : liste des codes des items cochés ou sélectionnés
 - items : objet contenant le paramètre « type », qui peut avoir la valeur « string », « number » ou « integer ».
 - enum : tableau de valeurs possibles pour chaque item de la liste
 - minItems : nombre minimum d'items qui peuvent être sélectionnés
 - maxItems : nombre maximum d'items qui peuvent être sélectionnés

3.3.8.12 Liste à sélection unique

Bouton radio, liste déroulante.

- Code du Format : listradio
- Type requis : string
- Paramètres associés :
 - displayformat : radio / combo
 - enum : tableau de valeurs possibles pour chaque item de la liste
 - default (optionnel) : valeur de l'item coché par défaut

3.3.8.13 Date

- Code du Format : date
- Type requis : string
- Paramètres :
 - format (optionnel) : code format utilisant les principes php : « d/m/Y »

3.3.8.14 Date et heure

- Code du Format : date-time
- Type requis : string
- Paramètres associés :
 - format (optionnel) : code format utilisant les principes php : « d/m/Y H:i:s »

3.3.8.15 Heure

- Code du Format : time

	Document de conception technique	Annexe 3 CCTP
	ARIA3	Date : 05/06/2026 Version : 7

- Type requis : string
- Paramètres associés :
 - format (optionnel) : code format utilisant les principes php : « d/m/Y H:i:s »

3.3.8.16 Année

- Code du Format : year
- Type requis : string
- Paramètres associés : aucun

3.3.8.17 Heure Unix

Heure unix définie selon le nombre de secondes écoulées depuis le 1^{er} janvier 1970.

- Code du format : timestamp
- Type requis : integer
- Paramètres associés : aucun

3.3.8.18 Référence

Ce type de champ permet de référencer une entité de référence située dans une collection distante.
Exemple : lexique.

- Code du Format : reference
- Type requis : object
- Paramètres associés : aucun
- Propriétés [properties], les champs sont de type « string »
 - [ref_collection] Collection : classe php gérant la collection à référencer
 - [ref_key] Clef distante : code du champ servant distant servant de référence
 - [ref_label] Label distant : code du champ à afficher dans la collection distante
 - [ref_typeselect] Type de sélection : liste, dialogue : lors de l'édition de la valeur de ce champ, la sélection d'une entrée se fera soit par choix dans une liste déroulante, soit par l'intermédiaire d'une boîte de dialogue permettant de rechercher et choisir la valeur (utile pour les collections contenant de nombreuses entrées)

3.3.8.19 Fichier

- Code du Format : file
- Type requis : object
- Paramètres associés :
 - extensionsAllowed (optionnel) : tableau d'extensions autorisées
 - extensionsForbidden (optionnel) : tableau d'extensions interdites
 - maxSize (optionnel) : taille max autorisée en Ko
- Propriétés [properties] :
 - _id : type objet, de format mongoid
 - name : type string, format text, nom réel du fichier
 - ondiskname : string, text, nom du fichier codé sur disque
 - size : integer, espace disque pris par le fichier en Ko
 - md5 : string, format text, checksum du fichier (à voir si utile ?)

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

3.3.8.20 Carte

Lors de l’affichage de la valeur du champ, la carte est appelée. Lors de l’affichage du formulaire de modification, 2 champs « flottant » sont affichés pour la saisie des coordonnées.

- Code du Format : map
- Type requis : object
- Paramètres :
 - virtual : booléen (true/false)
 - url : adresse de la carte à insérer avec des codes pour les coordonnées X et Y
 - fieldCoordx : nom du champ de coordonnée X dans la collection
 - fieldCoordy : nom du champ de coordonnée Y dans la collection

3.3.8.21 Format spécial « Any »

Ce format est utilisé pour gérer les champs de configuration variables qui ne possèdent pas de modèle. Dans ce cas la portion de champ dans le « any » sera manipulée depuis un éditeur JSON.

- Code du Format : any
- Type requis : any
- Paramètres : aucun

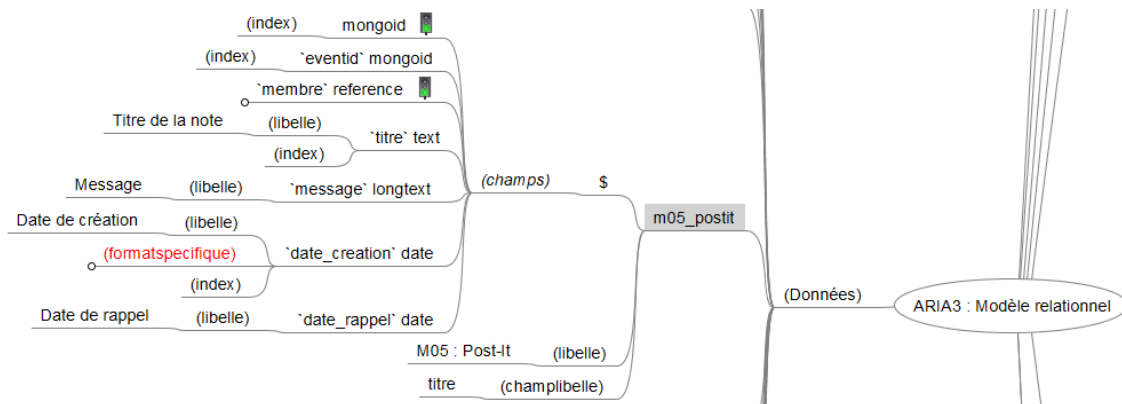
3.3.9 Synthèse des formats par type de champs

- **object** : mongoid, parentid, reference, file, map
- **string** : text, password, longtext, html, listradio, date, date-time
- **integer** : integer
- **number** : number
- **boolean** : checkbox, listyesno
- **array** : listmultiple
- **any** : any

3.4 Administration des modèles de données

L’interface d’administration de ARIA3 offre un accès en mode CRUD aux modèles et aux données, via deux onglets distincts. Chacun des deux onglets donne accès à l’ensemble des collection décrites dans le fichier Freemind de référence et propose une arborescence des données identique.

Dans les deux paragraphes suivants l’exemple de la collection simple des « post-its » sera présenté, le schéma décrit dans le Freemind est le suivant :



3.4.1 Représentation « Données »

Cet onglet permet d'accéder à toutes les données stockées dans les collections, selon les droits de l'utilisateur connecté. Exemple ci-dessous :

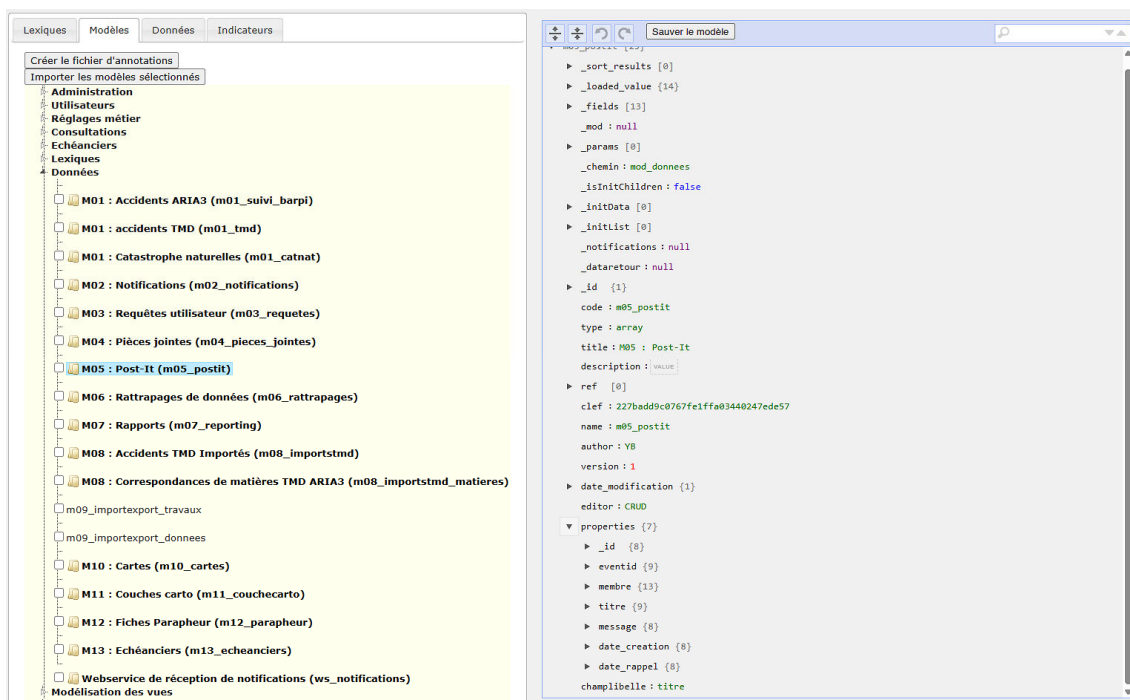
Lexiques	Modèles	Données	Indicateurs
Administration			
Utilisateurs			
Réglages métier			
Consultations			
Echéanciers			
Lexiques			
Données			
M01 : Accidents ARIA3			
M01 : accidents TMD			
M01 : Catastrophe naturelles			
M02 : Notifications			
M03 : Requêtes utilisateur			
M04 : Pièces jointes			
M05 : Post-It			
M06 : Rattrapages de données			
M07 : Rapports			
M08 : Accidents TMD Importés			
M08 : Correspondances de matières TMD ARIA3			
M10 : Cartes			
M11 : Couches carto			
M12 : Fiches Parapheur			
M13 : Echéanciers			
WebService de réception de notifications			
Modélisation des vues			
modèles de données			

Page	Précédente	1	/ 80	OK	Suivante	Rechercher	OK
eventid	Membre	Titre de la note	Message	Date de création	Date de rappel	Actions	
5447b39b1044d64e57d504ec	GR	géolocalisation	Je n'ai pas mis les coordonnées géo car je n'en suis pas sûr. Et j'en profite pour tester ce nouvel outil de notes.	22/10/2014 13:51:22		Détails Modifier Effacer	
5475dbf81044d6106326a151	GR	équipement	Je n'ai pas pu créer un équipement "engin terrestre", père de l'équipement "organe d'engin". Peut-être est-ce dû au cas TMD. Je ne sais pas si tu trouves ça normal, si ce n'est pas le cas, il faudrait peut-être faire un ticket.	27/11/2014 09:42:22		Détails Modifier Effacer	
54980dd21044d6596e82a408	GR	seveso	Sur la fiche gidic, le régime indique AS (autorisation avec servitude) mais NS (non Seveso). Le précédent accident était coché Seveso dans ARIA. Donc je n'ai pas rempli le tableau info à chaud dans le doute.	22/12/2014 12:55:05		Détails Modifier Effacer	
54992df61044d63c1682a44c	GR	seveso	Même site que l'accident du 02/11 donc même pôm pour le classement Seveso.	23/12/2014 09:06:32		Détails Modifier Effacer	
54a3b2351044d69e70e6e233	GR	exploitant	Entre les infos sur l'activité (eaux usées, déchets?) et la localisation d'après les différentes infos, j'ai choisi un exploitant mais sans certitude.	31/12/2014 09:01:47		Détails Modifier Effacer	
54aa95181044d6ec3b23e05f	GR	exploitant	Il y a plusieurs gidic pour le centre spatial, j'ai choisi celui qui était classé seveso.	05/01/2015 13:58:23		Détails Modifier Effacer	
54ac108e1044d60a1023ec90	DP	exploitant inconnu	localisation et nom de la société inconnue car source UIC qui préserve l'anonymat de ses membres	13/01/2015 10:51:23		Détails Modifier Effacer	

On peut visualiser dans le tableau CRUD les données décrites dans le modèle, les entêtes de colonne suivant les consignes du fichier de référence.

3.4.2 Représentation « Modèle »

La représentation au format « Modèle » donne un aperçu de la manière dont les champs décrits dans le Freemind sont retranscrits dans le modèle de données de ARIA3.



3.5 Modèles métier hors données

Dans le contexte d'un projet développé selon le principe MVC, les classes décrites dans le modèle de données constituent une partie importante des modèles métier développés dans l'application.

Au côté de ces modèles métier issus des données, des modèles spécifiques ont été conçus pour gérer les règles spécifiques de fonctionnement de l'application ARIA3.

- Modèle SOLR : Le modèle gère l'indexation et la réalisation des recherches sur le moteur d'indexation SOLR
- Modèles pour l'import et l'export des données externes
- Modèles de génération des extractions de données
- Echanges avec les services tiers : Webservices, GUN
- Mailings
- Sécurisation à double facteurs (2FA)
- Conversions de format (xlsx, docx, pdf) et manipulations de données (CSV)
- ...

3.6 Gestion des vues

Le système des vues est le système générique mise en œuvre dans ARIA3 pour créer des affichages basés sur des collections de données.

Les vues sont de type « CRUD » ou de type « formulaire », tel que décrites ci-dessous.

3.6.1 Gestion des vues en mode CRUD

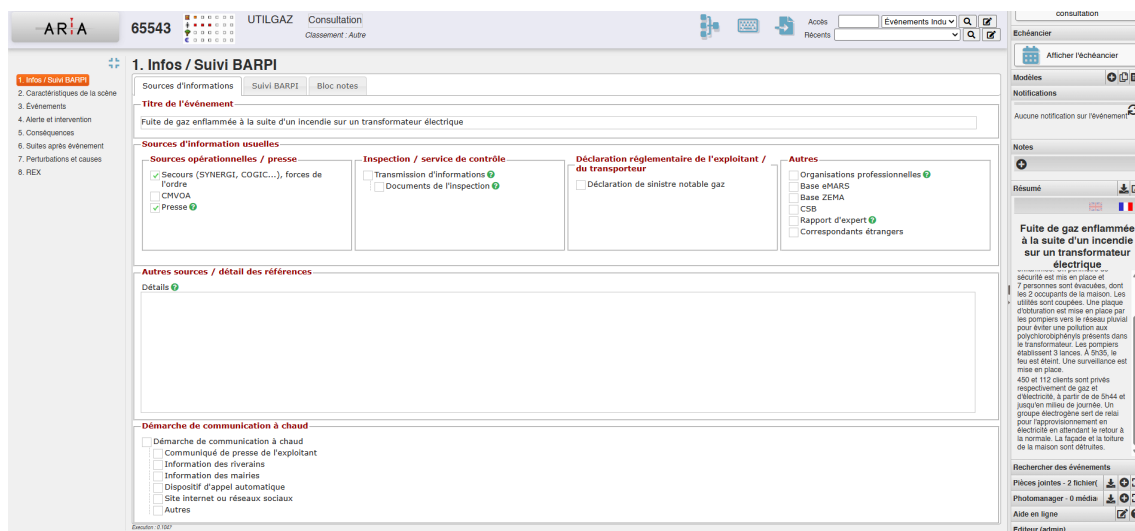
Pour chaque modèle relationnel défini, le mode « CRUD » permettra d'accéder directement à la donnée. Le mode CRUD permet de gérer des objets MongoDB complexes avec leur arborescence complète : tableaux d'objets contenant des tableaux, etc.

Lors de l’affichage des données du modèle, le mode CRUD affiche chaque champ selon ses consignes spécifiques. Ainsi pour chaque champ du modèle, lors de l’affichage du formulaire un champ spécifique sera affiché en fonction du format décrit dans le modèle relationnel.

La vue en mode « CRUD » est la vue par défaut utilisée pour accéder aux données et aux paramétrages depuis le panneau d’administration. Il permet, selon les droits de l’utilisateur, tous les types de manipulation de données : liste, affichage détaillé, modification, suppression, création.

3.6.2 Gestion des vues en mode « Masques »

C’est le mode standard d’exploitation des données par les utilisateurs finaux. Pour certaines collections principales (événements industriels, CATNAT et Parapheurs) des formulaires sur mesure sont présentés à l’utilisateur, structurées en masques, onglets, blocs et champs.



Cette représentation intègre l’exécution des règles métier client et serveur ainsi que l’interactif avec la colonne de compléments.

Cette vue est celle dont dispose l’utilisateur final de l’application : des masques de saisie personnalisés permettant de renseigner les informations relatives à la collection de référence (en l’occurrence « m01_suivi_barpi » la plupart du temps), sous une forme ergonomique et adaptée au métier.

Cette vue intègre l’exécution des règles métier.

Le modèle d’exécution des vues peut être appelé à n’importe quel niveau de la représentation.

3.6.2.1 Création d’un nouvel enregistrement

Lors de la création d’une nouvelle entrée à l’aide du système de vues, les éléments d’interface sont grisés hors masque initial de saisie. L’enregistrement sera créé (et l’identifiant créé) lorsque le formulaire initial aura été enregistré.

Lors de la conception de la vue, il est nécessaire d’indiquer le masque de saisie utilisé pour la saisie initiale de l’enregistrement. Deux possibilités :

	Document de conception technique	Annexe 3 CCTP
	ARIA3	Date : 05/06/2026 Version : 7

- sélectionner un masque par défaut : lors de l’affichage d’une vue sans identifiant d’enregistrement, c’est ce masque qui est appelé. Une fois validé, le numéro d’enregistrement est créé ;
- créer un masque « Saisie initiale » qui est appelé pour la création de l’enregistrement. Il peut contenir, par exemple, uniquement les informations essentielles à la création rapide d’un accident.

3.6.2.2 Affichage d’une vue

3.6.2.2.1 Paramètres d’entrée :

Code de la vue : le code unique identifiant une vue et son ensemble de masques.

Identifiant d’un enregistrement : une vue est nécessairement associée à une collection donnée. Si un identifiant est fourni, la vue est initialisée pour afficher cet identifiant. En l’absence d’identifiant, la vue est proposée en mode de saisie initiale.

3.6.2.2.2 Présentation

Le rendu présente le menu de navigation avec l’intégralité des masques, ainsi que le masque par défaut.

3.6.2.3 Affichage d’un masque :

3.6.2.3.1 Paramètres d’entrée :

Code de la vue : le code unique identifiant d’un masque.

Identifiant d’un enregistrement : un masque est nécessairement associé à une collection donnée. Si un identifiant est fourni, le masque est initialisé pour afficher cet identifiant. En l’absence d’identifiant, le masque est proposé en mode de saisie initiale.

3.6.2.3.2 Présentation

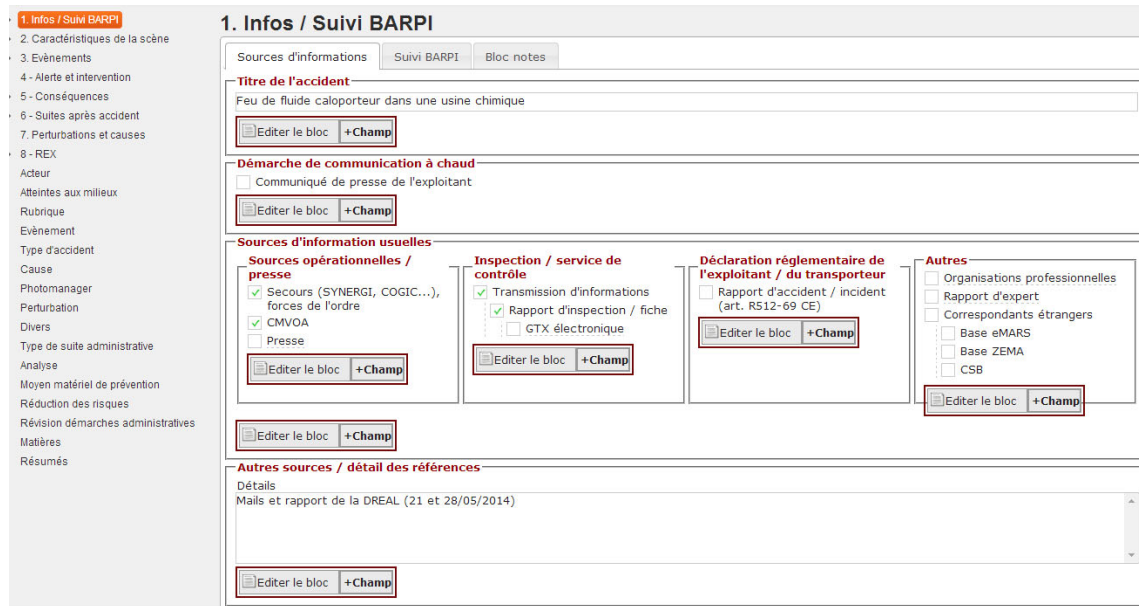
Le rendu présente uniquement un masque, avec ses blocs et sous-blocs.

3.6.3 Gestion des vues en mode « WYSIWYG »

Il est possible, lors de l’utilisation d’une vue en mode « masques », de passer du mode « Utilisateur » au mode « Concepteur WYSIWYG ». Ce dernier propose une vue similaire à la vision dont dispose l’utilisateur final, mais permet d’ordonner les champs et les blocs de champs les uns par rapport aux autres.

Le principe de présentation est le suivant pour les vues WYSIWYG :

Colonne gauche : Centre : la vue en cours d'édition
Le menu de navigation



1. Infos / Suivi BARPI

Sources d'informations | Suivi BARPI | Bloc notes

Titre de l'accident
Feu de fluide caloporteur dans une usine chimique
[Editier le bloc] +Champ

Démarche de communication à chaud
☐ Communiqué de presse de l'exploitant
[Editier le bloc] +Champ

Sources d'information usuelles

Sources opérationnelles / presse ☒ Secours (SYNERGI, COGIC...), forces de l'ordre ☒ CMVOA ☐ Presse [Editier le bloc] +Champ	Inspection / service de contrôle ☒ Transmission d'informations ☒ Rapport d'inspection / fiche ☐ GTX électronique [Editier le bloc] +Champ	Déclaration réglementaire de l'exploitant / du transporteur ☐ Rapport d'accident / incident (art. R512-69 CE) [Editier le bloc] +Champ	Autres ☐ Organisations professionnelles ☐ Rapport d'expert ☐ Correspondants étrangers ☐ Base eMARS ☐ Base ZEMA ☐ CSB [Editier le bloc] +Champ
---	---	--	---

[Editier le bloc] +Champ

Autres sources / détail des références
Détails
Mails et rapport de la DREAL (21 et 28/05/2014)
[Editier le bloc] +Champ

L'administrateur a à sa disposition une palette d'outils. Il a la possibilité de créer les éléments de vue :

- Groupe : ensemble de cadres qui pourront être représentés sous forme d'onglets au besoin ;
- Cadres : Grands blocs encadrés (ex : consultation d'une substance) ;
- Blocs : sous-ensembles des cadres, qui permet de réunir un ensemble de champs. Un bloc peut contenir d'autres blocs ;
- Champs : Unité de saisie, il s'agit de champs unitaires inclus dans un bloc. Le champ peut être affiché avec son libellé ou non ;
- Libellé : Elément purement textuel visible sur la représentation.

Chacun de ces éléments est décrit, a minima, par les propriétés suivantes :

- Mongold
- Code interne unique
- Libellé
- Libellé visible : oui/non
- Position du libellé : au-dessus, gauche, bas, droite
- Elément d'appartenance
- Position par rapport à l'élément d'appartenance

Il est possible d'interagir avec les éléments de mise en page de la manière suivante :

- cliquer sur l'élément de mise en forme souhaité dans la barre d'outils ;
- les éléments correspondants s'activent visuellement ;
- au survol, le curseur change et il devient possible de les déplacer d'une zone à l'autre ;

	Document de conception technique	Annexe 3 CCTP
	ARIA3	Date : 05/06/2026 Version : 7

- au clic, une zone ou une fenêtre de propriétés apparaît, permettant de modifier les propriétés énoncées ci-dessus.

3.7 Gestion des règles

Les règles métier sont un élément essentiel de la saisie des masques car elles permettent d'automatiser certains processus de contrôle ou de modification de données en fonction de données saisies par l'utilisateur, afin de s'assurer de la cohérence des données de l'accident.

Les règles peuvent s'exécuter de 2 manières :

- règles serveur : lors de la sauvegarde de l'évènement, la règle est exécutée et un retour spécifique est généré par le serveur. Lorsque le client reçoit ce retour spécifique il pourra agir selon le type de retour fourni :
 - o il peut s'agir d'un simple acquittement ;
 - o ou une erreur en cas d'interdiction ou de problème de sauvegarde ;
 - o le serveur a aussi pu retourner un champ à mettre à jour dans la vue.
- règles client : une action sur un champ peut déclencher l'exécution d'une formule qui pourra avoir un impact sur la valeur d'un champ dans l'interface

Les règles serveur ou client disposent de nombreux paramètres d'exécution : champ déclencheur, opération en front montant/descendant/permanent, formule de calcul...

Les règles serveur sont stockées dans la collection « adm_regles_champs », qui contient 37 règles différentes :

- 24 règles « client » qui déclenchent dynamiquement des cochages ou décochages automatiques en fonction des conditions ;
- 13 règles « serveur » qui réalisent des modifications de données sur l'objet lors d'une sauvegarde.

Principe d'exécution d'une règle serveur :

1. Lors de la sauvegarde d'un évènement, les règles serveurs sont chargées et vérifiées ;
2. Une règle est exécutée si les champs qui la concernent font partie des données modifiées, ou si la règle doit s'appliquer systématiquement à toute sauvegarde ;
3. Si les conditions sont remplies, une formule décrite dans la règle peut s'exécuter et déterminer une nouvelle donnée à inscrire dans l'évènement ;
4. La nouvelle donnée est alors inscrite dans l'objet et le processus de sauvegarde se poursuit.

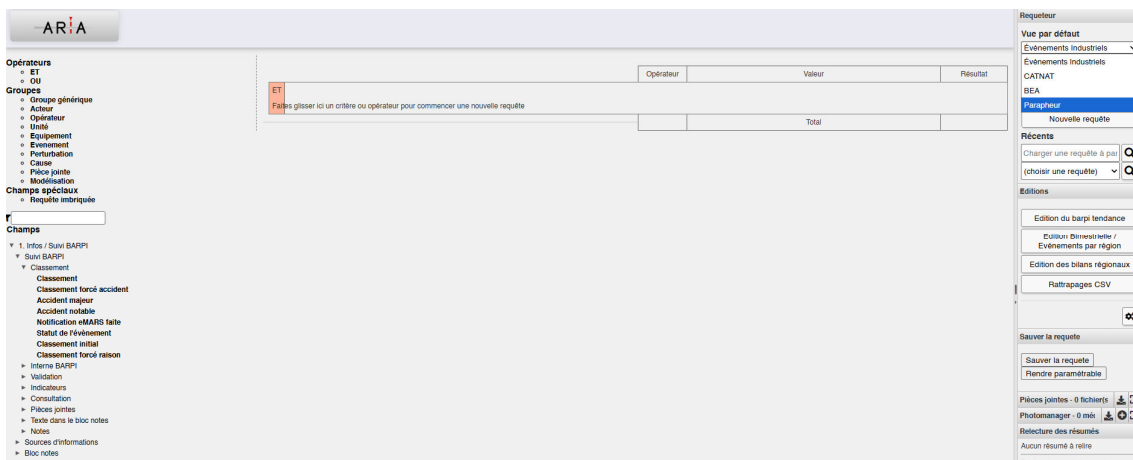
Le paramétrage détaillé des règles client et serveur est décrit dans la Documentation d'exploitation ARIA3.

4 Requêteur générique

La couche d'abstraction mise en œuvre pour décrire les masques de saisie permet d'implémenter un système de moteur de recherche au sein de l'application.

Les critères de filtrage proposés sont tirés directement de la combinaison du modèle de données (Freemind) et de la description des vues.

- en colonne gauche du moteur de recherche, les champs sont listés exactement selon leur emplacement dans les masques de saisie. En cas de modification d'une vue les modifications sont immédiatement prises en compte dans le requêteur ;
- lorsqu'un champ est glissé dans la zone de filtrage, le format d'affichage des critères de filtrage sont ceux correspondant aux paramètres du champ dans le modèle Freemind ;
- en colonne droite, les compléments permettent d'extraire sous différentes formes les données de l'échantillon résultat d'une recherche.



5 Modèle générique et Contraintes de pérennité

5.1 Applications connexes

L'application décrit précédemment est très générique et permet de répondre à des besoins très larges. L'application initiale, conçue sur cette base, a donc été reprise et étendue pour répondre à de nouveaux besoins métier.

Il existe trois contextes d'utilisation de l'outil :

- le contexte initial ARIA3 : l'application originale, permettant la saisie d'événements industriels, notamment, avec ses deux représentations : événements ou BEA-RI (simplifié) ;
- le contexte CATNAT : reprenant le même principe que le contexte industriel, permet la saisie d'accidents relatifs au domaine des catastrophes naturelles (ex. avalanches, tempêtes...) ayant impliqué des victimes ou des conséquences matérielles significatives (non usité actuellement) ;
- le parapheur : initialement traité hors informatique, le processus a été numérisé. Il consiste à identifier, via différentes sources, les événements entrant potentiellement dans le périmètre des événements industriels retenus pour ARIA3. Les fiches parapheur qui répondent aux critères sont alors converties en événements.

	Document de conception technique	Annexe 3 CCTP Date : 05/06/2026 Version : 7
	ARIA3	

L'application ARIA3 a été implémenté de manière identique entre les trois contextes énoncés, la différence se faisant dans le choix du modèle métier, d'une part, et la représentation des champs de données dans les masques d'autre part.

5.2 Contraintes de sécurité et accès

Pour éviter tout accès non-autorisé, tout accès à l'application doit passer par l'intermédiaire d'une authentification réalisée par le biais du protocole CAS avec le serveur d'authentification Cerbère :

<https://authentification.din.developpement-durable.gouv.fr/>

Les composants serveur (modules, version PHP, apache...) de même que le système d'exploitation doivent être maintenus à jour afin d'éviter tout risque lié à des failles exposées.

En complément d'une authentification décorrélée de l'outil ARIA3, un système de double authentification a été implémenté. Il se base sur des codes produits via une application localisée sur le téléphone des utilisateurs, le code étant demandé à intervalles réguliers ou en cas de changement de navigateur.

6 Contexte technique du site Web ARIA3

Le site ARIA3 est hébergé chez un hébergeur extérieur sur un serveur dédié, qui est sous la supervision d'un prestataire d'infogérance spécialisé.

Les opérations techniques de maintenance du site nécessitent donc de :

- superviser régulièrement les sauvegardes, fichiers et bases de données, afin de s'assurer qu'une restauration sera possible en cas de sinistre ;
- réaliser l'interface technique entre les utilisateurs métier, non techniciens, et l'infogérance.