À cause de la taille de l’application et de la taille des données à manipuler, il a été décidé d’utiliser un style de programmation shiny plus impératif basé sur des couples ObserveEvent/reactiveValues plutôt que d’utiliser des expressions reactive() plus traditionnelles. Vu que l’application gère plusieurs pages et plusieurs subsets de données simultanément, il est essentiel de garder en mémoire l’état de l’application ce qui se fait facilement avec des reactiveValues. Mais la mise à jour d’une reactiveValue peut entraîner des effets de bord importants car si une valeur à l’intérieur est modifiée, celle-ci entraîne l’invalidation de la valeur réactive et pourrait entraîner des chaînes d’évènement imprévues. En utilisant des observeEvent, la mise à jour des valeurs réactives ne se fait qu’en réaction à des évènements ciblés. Ceci permet donc de s’assurer que la mise à jour des données ne se fait que lorsque désiré. Pour plus de consistance dans le code et plus de simplicité, tout le fonctionnement de l’application se fait de cette manière, même si parfois l’utilisation de valeurs réactives traditionnelle serait plus justifiée.

userInfo : reactiveValues qui contient toute l’information nécessaire au bon fonctionnement de l’application. Contient les valeurs suivantes :

lang : Character. Langue courante.

page : Character. Id de la page courante.

data : data.frame. Contient toutes les données brutes chargées au lancement de l’application.

subsetCpt : numeric. Compteur incrémental pour garder la trace du nombre de subsets créés.

subsets : list(ECSubset). Liste de tous les subsets crées.

subsetData : liste(data.frame). Liste des données filtrées pour chaque subset. Les noms de la liste sont les ids des subsets correspondant aux données.

La classe ECSubset a été créée. Elle contient les champs suivants :

id : character. Identifiant unique du subset. Généré automatiquement en ajoutant la valeur de subsetCpt au moment de la création du subset à la chaine de caractères subset. E.g. subset1

filters : list(ECFilter). Liste de filtres à appliquer pour créer un subset de données.

label : character. Nom à afficher du subset.

misc : list(). Liste d’informations variables et spécifiques pour chaque subset utilisées principalement pour des raisons de performance, afin d’éviter des calculs inutiles lors de la mise à jour de l’UI, tel que par exemple les dates minimales ou maximales