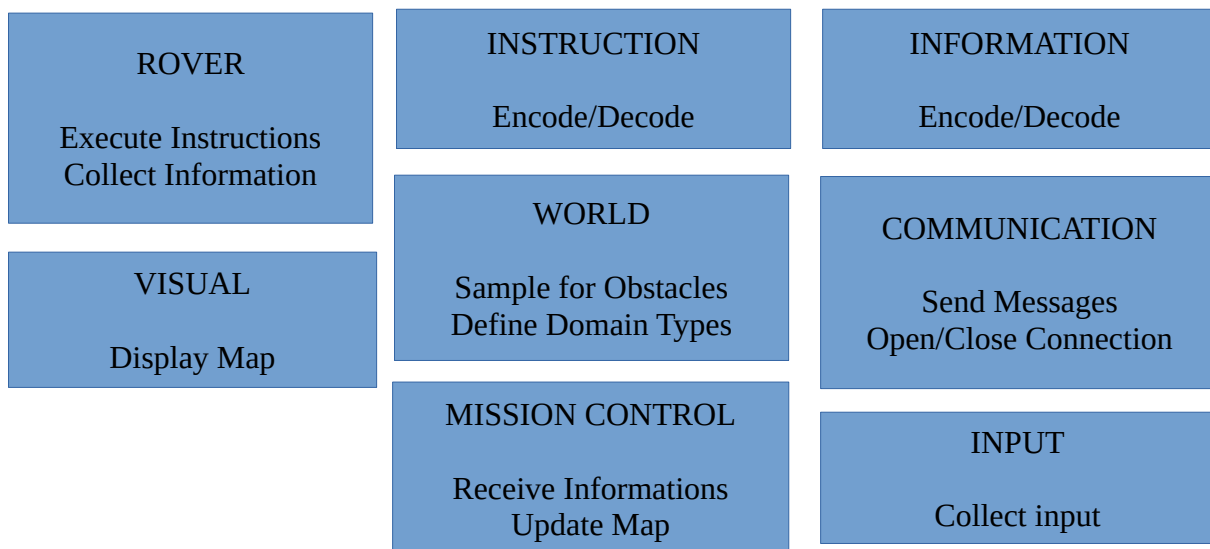


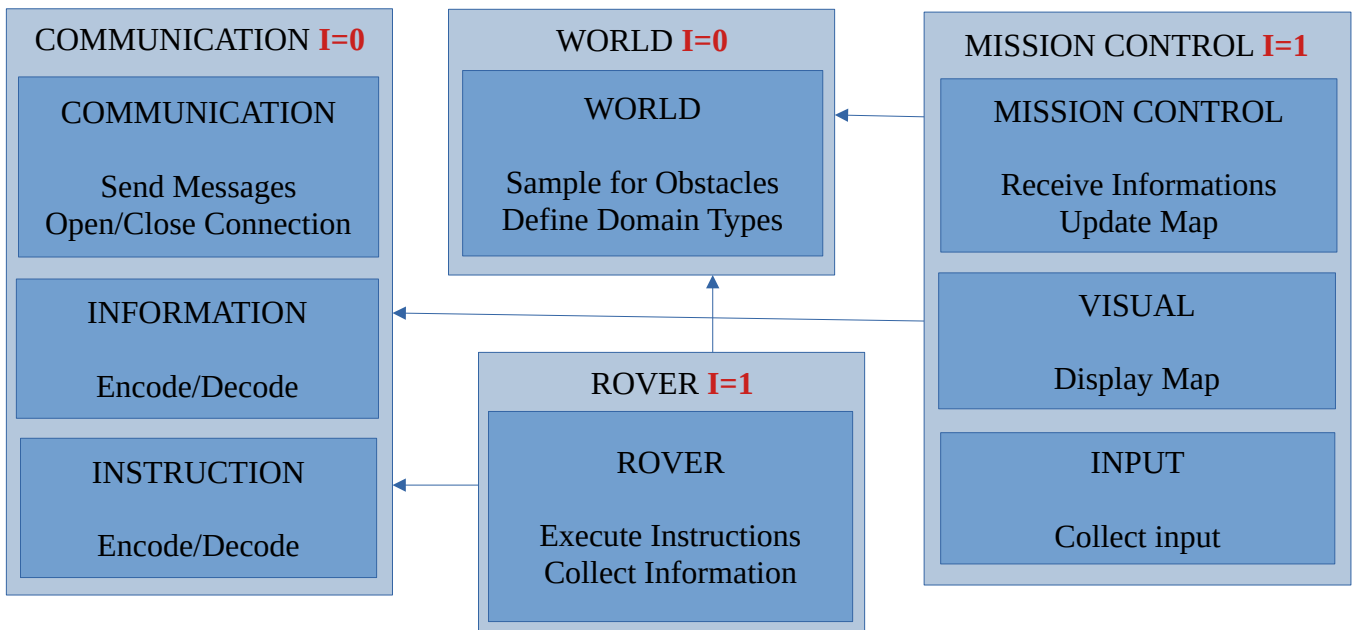
On commence par lister les “soucis” du Rover

- Définir les types du métier
- Établir la connexion entre Control et Rover
- Collecter les commandes de l'utilisateur
- Encoder/Décoder l'instruction pour le rover
- Envoyer des messages
- Recevoir des messages
- Exécuter l'instruction
- Interroger le monde pour des obstacles
- Encoder/Décoder l'information
- Mettre à jour la carte
- Représenter la carte à l'utilisateur

En combinant les soucis fortement liés on peut les répartir de la manière suivante.



Ces blocs peuvent être combinés en un nombre minimal de modules faiblement couplés de la manière suivante. Les flèches représentent les dépendances.

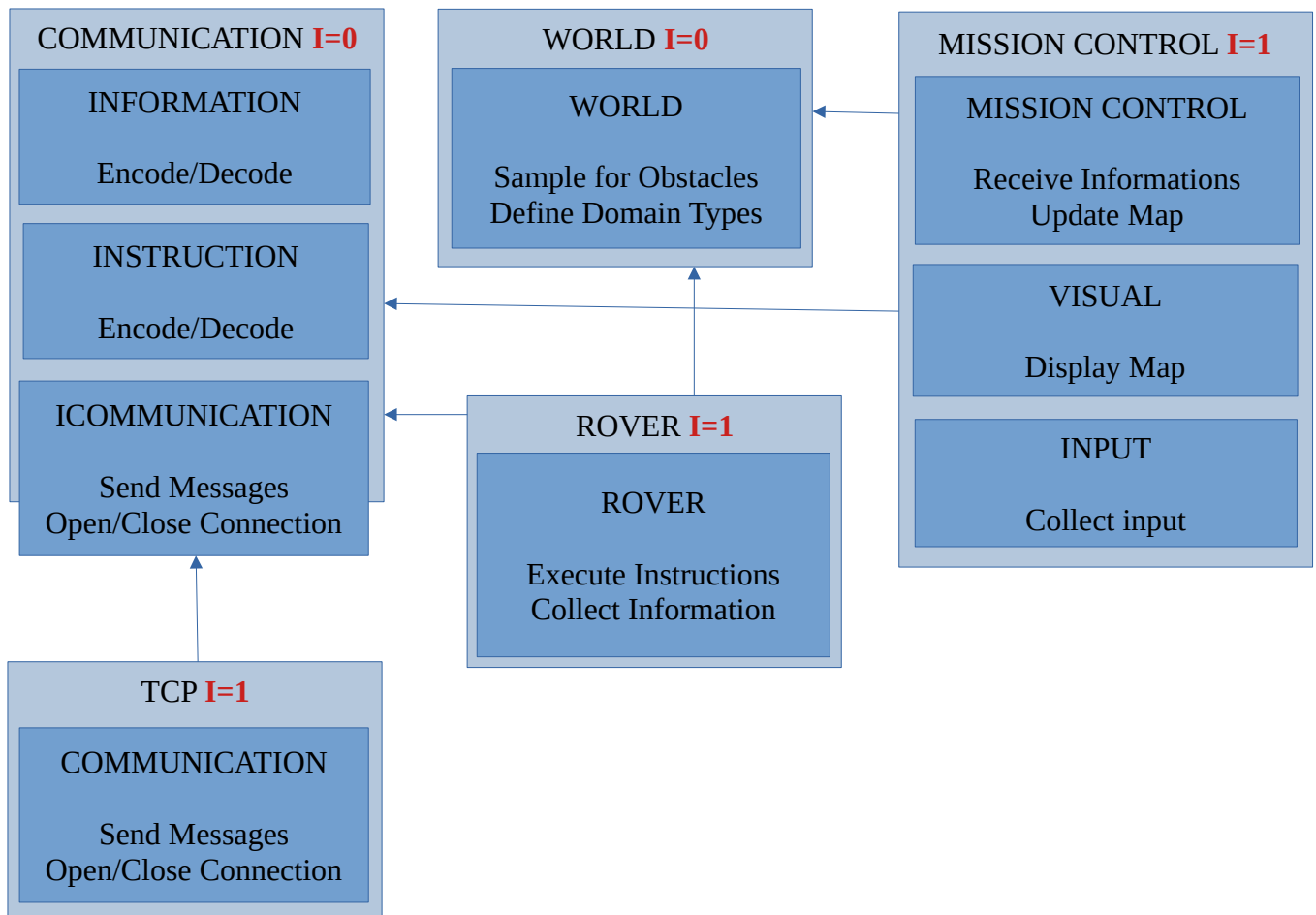


Ainsi MissionControl et Rover sont tous deux dépendants de Communication.

Mais Communication est un module volatil et stable, il est donc souhaitable d'isoler les spécificités de son implémentation du reste de l'application afin d'amortir les conséquences d'un changement de protocole.

Pour cela nous séparons communications, en :

- d'une part un module abstrait, intégrant les règles de communications de notre application et une interface que les différentes implémentations devront respecter.
- d'autre part, des modules individuels contenant l'implémentation pour chaque protocole, ici TCP.



Le style architectural résultant de ces choix est une architecture en Onion dans laquelle :

- le cœur de l'application repose sur les fondamentaux métier (planète, Vector, ...)
- Sur lesquels viennent se greffer les applications spécifiques (rover, mission control)
- Autour de laquelle gravitent différentes implémentations des communications dépendantes de l'infrastructure choisies, et découplées du reste par une interface côté Application afin de ne faire transiter que des messages neutres (ici du texte).

Pour l'analyse SWOT :

Strengths :

- Limite le surdesign
- L'application est centrée sur son module le moins volatil.
- Ses modules les plus volatils (TCP, ...) sont isolés du reste par une interface.

Weaknesses :

Opportunities :

- Le module de communication est très simple à modifier/remplacer

Threats :

- Une modification des primitives Métiers auraient des répercussions partout.