Trames de communication standard entre API et Simulateur(s)

Syntaxe de la documentation:

Nom de l'action
→ Sens API vers simulateur
<Opcode > <Trame de communication>\n
← Sens simulateur vers l'API
< Opcode> <Trame de communication>\n

SHUTDOWN : Eteindre le simulateur
Opcode : 0
→ Envoie le signal de fin au simulateur pour que le programe se ferme proprement.
0 \n
← Pas de reponsse de la part du simulateur dans ce cas.

NEWMODULE : Sélection d'un module
Opcode : 1
→ Envoie le nom du module que nous voulons charger au simulateur. Ce nom fait reference a un fichier ranger sous le dossier "XML/" dans le dossier du simulateur.
1 <nom_du_module>\n
← Le simulateur confirmation du chargement du module a l'interface utilisateur.
1 <chemin absolu vers le fichier du module charger>\n

TICK : Lancement d'une simulation
Opcode : 2
→ Envoie de la commande pour lancer la simulation d'un tic.
2 \n
← Renvoie de l'état a la fin de la simulation, 1 : pas d'erreur , 0 : une erreur est survenue.
2 <boolean erreur>\n

PULLID : Demande de recuperation des donnes de la simulation
Opcode : 3
→ Envoie d'une requête pour récupérer des informations sur l'état des donnes dans la simulation. Pour ce faire nous requittons chaque arête avec son id.
3 <id_arête>,<id_arête>,<id_arête>, ...\n
← Le simulateur renvoie une liste des id demander suivie des valeurs qu'ils ont.
3 <id_arête>: <data>,<id_arête>:<data>,... <boolean_erreur>\n

COMMITID : Demande de l'API vers le simulateur pour changer la valheur de la donne sur une ligne de donnée.
Opcode : 4
→ Envoie des ligne de donnes a changer avec leurs nouvelles valeurs.
4 <id_arête>:<new_data>,<id_arête>:<new_data> ,... \n
← Validation de la modification des valeurs des aretes.
4 <boolean_erreur>\n

RESET : Demande de l'API pour réinitialiser les valheurs de la simulation.
Opcode : 5
→ Envoie de la commande pour reinisialiser les donnes de la simulation.
5 \n
← Renvoie de la validation du reset.
5 <boolean_erreur>\n
ASKDATAFILE : Demande de l'API récupérer le chemin absolu vers le fichier ou est sauvegarder la donne contenu dans un bloc logique, eg: PC , Registres , PD , ect ...
Opcode : 6
→ Demande de chemin vers le fichier ou est contenu la donne d'un bloc logique.
6 <id_du_bloc_logic>\n
← Renvoie du chemin absolu vers le fichier ou est contenu la donne de bloc logique demander.
6 <chemin_absolu_vers_le_fichier> <boolean_erreur>\n

LOADDATAFILE : Demande de l'API de remplacer les donnes contenu dans un bloc logique par les donnes contenu dans un fichier dont le chemin relatif est passer par l'API.
Opcode : 7
→ Demande de modification de la donnée dans un bloc logique.
7 <id_bloc_logique> <path_vers_fichier_buffer>\n
← Renvoie de la confirmation comme quoi nous avons effectuer la modification de la donnée.
7 <id_bloc_logique> <path_vers_fichier_buffer> <boolean_erreur>\n

SAVETOXML : Sauvgarder l'etat du graph vers un xml
Opcode : 8
→ Demande l'ecriture du module charger par le simulateur dans son etat courant vers le fichier xml preciser dans la trame.. Si le fichier n'existe pas il sera cree.
8 <path_vers_fichier_xml>\n
← Renvoie de la confirmation comme quoi nous avons effectuer l'ecriture des donnees vers le xml.
8 <path_vers_fichier_buffer>\n