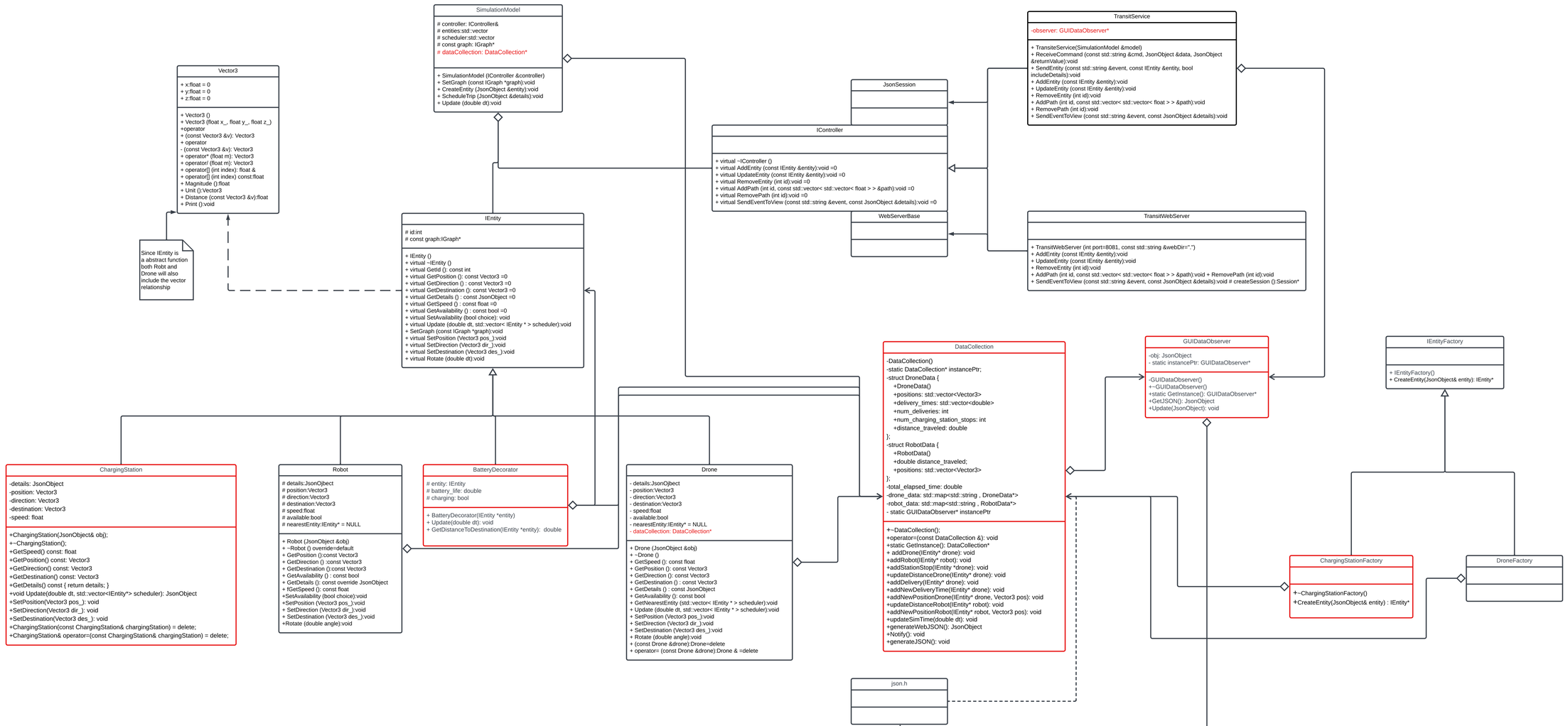
SimulationModel
controller: IController&
entities:std::vector
scheduler:std::vector
const graph: IGraph*
dataCollection: DataCollection*
+ SimulationModel (IController &controller)
+ SetGraph (const IGraph *graph):void
+ CreateEntity (JsonObject &entity):void
+ ScheduleTrip (JsonObject &details):void
+ Update (double dt):void
Vector3
+ x:float = 0
+ y:float = 0
+ z:float = 0
+ Vector3 ()
+ Vector3 (float x_, float y_, float z_)
+operator
+ (const Vector3 &v): Vector3
+ operator
- (const Vector3 &v): Vector3
+ operator* (float m): Vector3
+ operator/ (float m): Vector3
+ operator[] (int index): float &
+ operator[] (int index) const:float
+ Magnitude ():float
+ Unit ():Vector3
+ Distance (const Vector3 &v):float
+ Print ():void
Since IEntity is a abstract function both Robt and Drone will also include the vector relationship
IEntity
id:int
const graph:IGraph*
+ IEntity ()
+ virtual ~IEntity ()
+ virtual GetId (): const int
+ virtual GetPosition (): const Vector3 =0
+ virtual GetDirection () : const Vector3 =0
+ virtual GetDestination (): const Vector3 =0
+ virtual GetDetails () : const JsonObject =0
+ virtual GetSpeed () : const float =0
+ virtual GetAvailability () : const bool =0
+ virtual SetAvailability (bool choice): void
+ virtual Update (double dt, std::vector< IEntity * > scheduler):void
+ SetGraph (const IGraph *graph):void
+ virtual SetPosition (Vector3 pos_):void
+ virtual SetDirection (Vector3 dir_):void
+ virtual SetDestination (Vector3 des_):void
+ virtual Rotate (double dt):void
JsonSession
IController
+ virtual ~IController ()
+ virtual AddEntity (const IEntity &entity):void =0
+ virtual UpdateEntity (const IEntity &entity):void =0
+ virtual RemoveEntity (int id):void =0
+ virtual AddPath (int id, const std::vector< std::vector< float > > &path):void =0
+ virtual RemovePath (int id):void =0
+ virtual SendEventToView (const std::string &event, const JsonObject &details):void =0
WebServerBase
TransitService
-observer: GUIDataObserver*
+ TransiteService(SimulationModel &model)
+ ReceiveCommand (const std::string &cmd, JsonObject &data, JsonObject &returnValue):void
+ SendEntity (const std::string &event, const IEntity &entity, bool includeDetails):void
+ AddEntity (const IEntity &entity):void
+ UpdateEntity (const IEntity &entity):void
+ RemoveEntity (int id):void
+ AddPath (int id, const std::vector< std::vector< float > > &path):void
+ RemovePath (int id):void
+ SendEventToView (const std::string &event, const JsonObject &details):void
TransitWebServer
+ TransitWebServer (int port=8081, const std::string &webDir=".")
+ AddEntity (const IEntity &entity):void
+ UpdateEntity (const IEntity &entity):void
+ RemoveEntity (int id):void
+ AddPath (int id, const std::vector< std::vector< float > > &path):void + RemovePath (int id):void
+ SendEventToView (const std::string &event, const JsonObject &details):void # createSession ():Session*
DataCollection
-DataCollection()
-static DataCollection* instancePtr;
-struct DroneData {
+DroneData()
+positions: std::vector<Vector3>
+delivery_times: std::vector<double>
+num_deliveries: int
+num_charging_station_stops: int
+distance_traveled: double
};
-struct RobotData {
+RobotData()
+double distance_traveled;
+positions: std::vector<Vector3>
};
-total_elapsed_time: double
-drone_data: std::map<std::string , DroneData*>
-robot_data: std::map<std::string , RobotData*>
- static GUIDataObserver* instancePtr
+~DataCollection();
+operator=(const DataCollection &): void
+static GetInstance(): DataCollection*
+ addDrone(IEntity* drone): void
+addRobot(IEntity* robot): void
+addStationStop(IEntity *drone): void
+updateDistanceDrone(IEntity* drone): void
+addDelivery(IEntity* drone): void
+addNewDeliveryTime(IEntity* drone): void
+addNewPositionDrone(IEntity* drone, Vector3 pos): void
+updateDistanceRobot(IEntity* robot): void
+addNewPositionRobot(IEntity* robot, Vector3 pos): void
+updateSimTime(double dt): void
+generateWebJSON(): JsonObject
+Notify(): void
+generateJSON(): void
GUIDataObserver
-obj: JsonObject
- static instancePtr: GUIDataObserver*
-GUIDataObserver()
+~GUIDataObserver()
+static GetInstance(): GUIDataObserver*
+GetJSON(): JsonObject
+Update(JsonObject): void
IEntityFactory
+ IEntityFactory()
+ CreateEntity(JsonObject& entity): IEntity*
ChargingStation
-details: JsonObject
-position: Vector3
-direction: Vector3
-destination: Vector3
-speed: float
+ChargingStation(JsonObject& obj);
+~ChargingStation();
+GetSpeed() const: float
+GetPosition() const: Vector3
+GetDirection() const: Vector3
+GetDestination() const: Vector3
+GetDetails() const { return details; }
+void Update(double dt, std::vector<IEntity*> scheduler): JsonObject
+SetPosition(Vector3 pos_): void
+SetDirection(Vector3 dir_): void
+SetDestination(Vector3 des_): void
+ChargingStation(const ChargingStation& chargingStation) = delete;
+ChargingStation& operator=(const ChargingStation& chargingStation) = delete;
Robot
details:JsonOjbect
position:Vector3
direction:Vector3
destination:Vector3
speed:float
available:bool
nearestEntity:IEntity* = NULL
+ Robot (JsonObject &obj)
+ ~Robot () override=default
+ GetPosition ():const Vector3
+ GetDirection () :const Vector3
+ GetDestination ():const Vector3
+ GetAvailability () : const bool
+ GetDetails (): const override JsonObject
+ fGetSpeed (): const float
+SetAvailability (bool choice):void
+SetPosition (Vector3 pos_):void
+ SetDirection (Vector3 dir_):void
+ SetDestination (Vector3 des_):void
+Rotate (double angle):void
BatteryDecorator
entity: IEntity
battery_life: double
charging: bool
+ BatteryDecorator(IEntity *entity)
+ Update(double dt): void
+ GetDistanceToDestination(IEntity *entity): double
Drone
- details:JsonOjbect
- position:Vector3
- direction:Vector3
- destination:Vector3
- speed:float
- available:bool
- nearestEntity:IEntity* = NULL
- dataCollection: DataCollection*
+ Drone (JsonObject &obj)
+ ~Drone ()
+ GetSpeed (): const float
+ GetPosition (): const Vector3
+ GetDirection () :const Vector3
+ GetDestination () : const Vector3
+ GetDetails () : const JsonObject
+ GetAvailability (): const bool
+ GetNearestEntity (std::vector< IEntity * > scheduler):void
+ Update (double dt, std::vector< IEntity * > scheduler):void
+ SetPosition (Vector3 pos_):void
+ SetDirection (Vector3 dir_):void
+ SetDestination (Vector3 des_):void
+ Rotate (double angle):void
+ (const Drone &drone):Drone=delete
+ operator= (const Drone &drone):Drone & =delete
ChargingStationFactory
+~ChargingStationFactory()
+CreateEntity(JsonObject& entity) : IEntity*
DroneFactory
json.h