Cargo Management System - Class Diagram

SimulationController

typedef boost::variant<Event_TrainArrived, Event_TrainUnloaded, Event_Shutdown> **Event**;

- cms_: CMS*
- eventQueue : std::queue<Event>
- event_visitor: ScHandleEventVisitor
- event_sc: std::thread
- cond: std::condition_variable
- cond_m: std::recursive_mutex

+ boost::signals2::signal<void(cm::Train::Ptr)> trainArrivedAtStation;
+ boost::signals2::signal<void(cm::Train::Ptr)> trainUnloaded;

+ SimulationController(CMS*)
+ ~SimulationController()
+ StartSimulation(std::list<Train::Ptr>&): void
+ PushEvent(Event)

- EventHandler(): void
- ReceiveTrain(Train::Ptr): void
- UnloadTrain(Train::Ptr): void
- Sendtrain(Train::Ptr): void

CMS (Cargo Management System)

typedef boost::variant<Event_TrainAtStation, Event_TrainAtPlatform, Event_TrainFullyLoaded, Event_TrainLeftStation, Event_Shutdown> **Event**;

- ID: std::string
- simControl: SimulationController*
- station: Station
- decommissionedTrains: std::list<Train::Ptr>

- eventQueue: std::queue<Event>
- event_visitor: CmsHandleEventVisitor

- event_cms: std::thread
- cond: std::condition_variable
- cond_m: std::recursive_mutex

+ boost::signals2::signal<void(cm::Platform*)> trainAtPlatform;
+ boost::signals2::signal<void(cm::Platform*)> trainFullyLoaded;
+ boost::signals2::signal<void(cm::Train::Ptr)> trainLeftStation;

+ CMS(std::string, int)
+~CMS()
+ SetSimulationController(SimulationController): void
+ getID(): std::string {query}
+ StartCMS(): void
+ StopCMS(): void
+ HasCargo(): bool
+ PushEvent(Event): void
+ getNumDecommissionedTrains(): int
+ addCargo(std::list<Cargo::Ptr>&int = -1)
- EventHandler(): void
- SendTrainToPlatform(Train::Ptr): void
- DequeueTrains(): void
- LoadTrain(Platform*): void
- SendTrain(Platform*): void

ScHandleEventVisitor : boost::static_visitor

- _sc: SimulationController*

+ ScHandleEventVisitor(SimulationController* sc)
+ operator()(const Event_TrainArrive&) const: void
+ operator()(const Event_TrainUnloaded&) const: void

CmsHandleEventVisitor : boost::static_visitor

- _cms: CMS*

+ CmsHandleEventVisitor(CMS*)
+ operator()(const Event_TrainsAtStation&) :void {query}
+ operator()(const Event_TrainsAtPlatform&) :void {query}
+ operator()(const Event_TrainsFullyLoaded&) :void {query}
+ operator()(const Event_TrainsLeftStation&) :void {query}

Train

typedef std::shared_ptr<Train> **Ptr**;

- _id: int
- _name: std::string
- nxt_id: static int
mut: std::mutex

+ Train()
+ Train(std::string)
+ ~Train()
+ operator=(const Train&): Train&
+ Train(const Train&&)
+ operator=(const Train&&): Train&&
+ virtual canHold(Cargo::Ptr) = 0: bool
+ virtual load(Cargo::Ptr) = 0: bool
+ virtual unload() = 0: Cargo::Ptr
+ virtual getTotalWeight() =0: int
+ virtual getCapacity() = 0: int
+ virtual calculatePossibleLoad(std::list<Cargo::Ptr>) = 0: int
+ getID(): int
+ getName: std::string
+ isEmpty(): bool
+ isFull() bool

Station

- trainQueue: std::queue<Train::Ptr>
- platforms: std::list<Platform>
- name: std::string

+ Station(std::string, int)
+ getName: std::string {query}
+ getPlatforms(): std::list<Platform>*
+ getTrainQueue(): std::queue<Train::Ptr>*
+ isFull(): bool
+ isEmpty(): bool
+ hasCargo(): bool

template<int cap> Locomotive

typedef boost::mpl::bool_<true> **IS_LOCOMOTIVE;**
+ capacity: static const int

template< int cap, CL> Carriage

typedef boost::mpl::bool_<true> **IS_CARRIAGE**
typedef IS_A_VALID_CARGO_LIST<CL> **META_INFO**
typedef CL **CARGO_LIST**
typedef boost::mpl::int_<cap> **CAPACITY**;

+ cargo: std::list<Cargo::Ptr
+ isTanker = META_INFO::IS_TANKER::value: const bool
+ hasLivestock = META_INFO::HASLIVESTOCK::value const bool
+ carriage_cap: const int

+ Carriage()
+ ~Carriage()
+ Carriage(const Carriage&)
+ operator=(const Carriage): Carriage&
+ Carriage(const Carriage&&)
+ operator=(const Carriage&&): Carriage&
+ canHold(Cargo::Ptr): bool {query}
+ load(Cargo::Ptr): bool
+ unload(): Cargo::Ptr
+ getTotalWeight(): int {query}
+ getCapacity: int {query}

Platform

- num_id: static int
- id: std::string
- train: Train::Ptr
- cargo: std::list<Cargo::Ptr>

+Platform()
+ getID(): std::string {query}
+ getTrain(): Train::Ptr
+ getCargoList(): std::list<Cargo::Ptr>
+ trainArrive(Train::Ptr: bool
+ trainDepart(): Train::Ptr
+ isFree(): bool

Cargo

+ weight: const int
+ loadtime: const int

+ Cargo(int, int)
+ ~Cargo()

Cargo Variations (child classes)

Regular:	Timber	Coal	Grains
Liquids:	Oil	Gasoline	Water
Livestock:	Sheep	Cows	Pigs