A logistic control framework for automated transportation networks

Matthieu van der Heijden[1], Mark Ebben and Aart van Harten

University of Twente, The Netherlands

Abstract

A logistic control system is needed for a fully automated cargo transportation system using Automatic Guided Vehicles, as is currently being developed around Schiphol Airport, Amsterdam. This control system should cover a variety of logistics decisions, such as order consolidation, order scheduling, resource capacity management and traffic control. We address the problem of control flexibility and software adaptability versus stable, predictable and optimised overall system behaviour. We design a distributed logistic control framework in which local agents use heuristics to solve their control problems. We use simulation to evaluate our object library. The flexibility of the logistic control structure is illustrated by some examples of decision integration and model extensions. We conclude that *modification flexibility*, related to changes in the system structure and processes, is easier to achieve than *integration flexibility*, the integration of control tasks, because the latter is strongly influenced by the underlying information structure.

Key words: transportation, logistics, optimisation

[1] Faculty of Technology and Management, Dept. of Management Science and Logistics, P.O. Box 217, 7500 AE Enschede, The Netherlands, e-mail m.c.vanderheijden@sms.utwente.nl

1. Introduction

A transportation network generally consists of various organizations (shippers, freight forwarders, transport companies, logistic service providers) cooperating to move cargo from its origin to its destination. A accompanying logistic planning and control system should take into account the distributed responsibilities and spans of control in such networks. Also the information on orders and resource utilization is distributed over the various parties. This naturally leads to a distributed planning and control framework in which logistic decisions are decentralized. In order to achieve efficiency, information exchange between decision units is needed.

We encountered such a transportation network when participating in a project on the design of a fully automated, underground cargo transportation system around Schiphol Airport, Amsterdam, the Netherlands, see Ebben [2001]. This system is called OLS, a Dutch abbreviation for Underground Logistics System. The OLS uses Automatic Guided Vehicles (AGVs) to provide a fast, efficient and flexible connection between the modalities air (Schiphol), rail (a future rail terminal close to Schiphol) and road (10-15 terminals of freight forwarders, shippers and air freight handlers, both around Schiphol and at the world's largest flower auction in Aalsmeer). The network spans 10-20 km (depending on the specific layout) and focuses on time-critical products such as flowers, vegetables, newspapers and spare parts. One of our tasks in the project was to design a logistic control system for such a network and to evaluate it using computer simulation. The organizations involved include an AGV system operation company and terminal owners such as freight forwarders, air freight handlers and a railroad company. These organizations plan their own activities and cooperate in order to achieve optimal logistic performance in terms of order throughput times and on-time percentages.

We aim to design an agile and efficient control system based on the principles of Evers et al [2000] and Damen [2001]. Cooperating agents provide a flexible framework for control, each agent being responsible for a specific decision or domain. The logistic agents in such networks communicate with each other and reach their specific objectives through negotiations (Jennings and Wooldridge [1998], Espinasse et al [1998]). Negotiations require pricing strategies for the outcomes of planning decisions leading to bids on orders available. It is well known that bid calculation can be a difficult task (Qinhe et al [2001]). Even if this is solved, a dynamic environment

may require frequent renegotiation for which separate procedures need to be developed. Renegotiation may be essential to maintain planning flexibility. Further, there are major drawbacks of an agent-based approach (cf. Jennings [2000]):

- the patterns and outcomes of the interactions between agents may be inherently unpredictable;
- predicting the behaviour of the overall system, based on its constituent components, is extremely difficult
- overall system behaviour may be insufficient

Because of these drawbacks, there has been a separate line of research on stability of multi-agent networks, see e.g. Kephart et al. [2000] and Wellman et al [2001]. An option to prevent these problems is to introduce network coordination, for example by clustering related decisions in an integrated control block. Coordination is natural in case of capacitated shared resources. It may also prevent that simple goal seeking heuristics at a local level lead to suboptimal overall system performance. However, a disadvantage is that this may hamper flexibility in the sense of responsiveness to disturbances (e.g. equipment failure, rush orders) and adaptability to logistic system or process modifications. Therefore, there is a trade-off between control flexibility and software adaptability on one hand and a stable, predictable and optimised logistic performance on the other hand.

In this paper, we aim to develop a flexible logistic control framework based on local control with information exchange for automated transportation networks like the OLS, allowing to integrate control decisions where useful. Such a framework can be implemented using an object-oriented approach for logistics control, see Van der Zee [1997], based on the theory of object oriented design (Bootch [1994], Zeigler [1990]). In this approach, control tasks are explicitly separated from physical tasks and information flows, thereby providing the flexibility to interchange alternative control rules and algorithms for the same logistic control decision. The framework is such, that both local autonomous control agents as well as higher level coordination units are facilitated.

Although the principle of separating physical activities, control tasks and information flows is simple, the design of such a control system with corresponding unambiguous information flows for a specific application is not trivial. The main cause is that various options for control rules and algorithms have different information

requirements and/or different needs for the *timing* of information. For example, advanced algorithms for vehicle planning require early information on order arrivals or forecasts of workloads per terminal in order to outperform simple heuristics. Therefore, it is important to define the information interface carefully to facilitate exchange of control objects without affecting information flows. This facilitates testing of several logistic planning and control scenarios. Also, it may provide the opportunity to include integral control objects that can temporarily be replaced by several local control objects as *fall back* scenario in the case of emergencies. For example, if the communication system fails, essential information may be lacking for an advanced order scheduling algorithm and then a set of simple dispatch rules facilitates continuing system operation based on the (limited) information that is locally available.

Summarized, the goals of our research are:

1. to develop a well structured framework for logistic control;
2. to define a set of goal seeking heuristics for the logistic control agents;
3. to test the modification flexibility of our framework, i.e. the ability to adjust to a modified system structure and to modified processes;
4. to test the integration flexibility of our framework, i.e. the ability to combine the control tasks of various agents into one overall agent.

In the next section, we discuss our model of the transportation network, the order structure and the logistic activities to be planned. In Section 3, we introduce our logistic planning and control framework. We discuss the goal seeking heuristics for the local agents in Section 4. Next, we explain how we applied this framework to the underground logistic system around Schiphol (OLS) in Section 5. As for *modification* flexibility, we show that the basic framework can easily be adjusted to account for new aspects like layout changes and for new control tasks to be added such as equipment failure management, AGV energy management and finite storage capacities at terminals (Section 6). As far as performance optimisation through improved coordination (*integration* flexibility), we give an example of local control object integration in Section 7. Finally, we present our conclusions in Section 8.

2. Model of transportation network and logistic processes

2.1. Network and order structure

Let us first describe the physical resources. An automated transportation network consists of a set of nodes connected by links, see Figure 1.

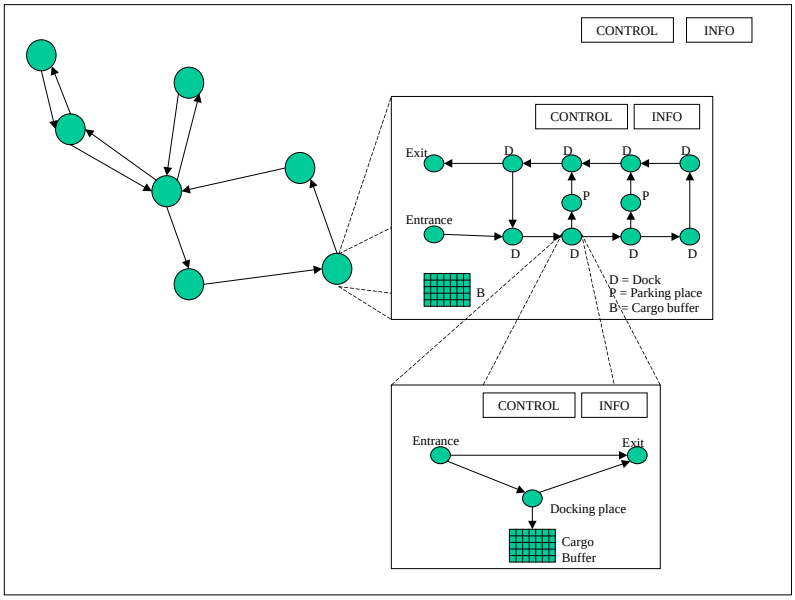


Figure 1. Example of an automated transportation network

Nodes are terminals between which cargo is transported by vehicles (AGVs) and parking areas to store vehicles when they are not needed elsewhere. Links can both be unidirectional and bi-directional tracks, i.e. suitable for traffic in one or two directions. The connections between two nodes can consist of (a) two unidirectional tracks, one for traffic in each direction (2) one unidirectional track, such that a detour is necessary for traffic in the opposite direction (see Figure 1) and (3) one bi-directional tube alternately to be used by traffic in each direction. The latter option, a so-called *two-way track*, is included because we encountered it in the OLS project as an investment saving for the (expensive) underground AGV track (tube) construction. Bi-directional tracks imply that vehicles have to wait if the track is occupied by vehicles travelling in the opposite direction, thereby causing delay and batching of vehicles.

Each node has an internal structure, consisting of various nodes connected by links. Internal nodes include AGV parking areas, docks for loading and unloading of cargo, buffers and possibly workstations for activities like cargo checking and (de)groupage. Links are tracks along which AGVs can drive and internal cargo transportation systems between docks, buffers and workstations. This structure may be further refined. For example, a dock may consist of an entrance, exit, docking place and a small buffer for cargo waiting to be loaded. Each level in the hierarchy contains control objects (agents) for the local decisions and information objects containing the input and output of the control objects.

Orders to be handled by this system have the following characteristics: (a) an origin and destination (terminals) and possibly a routing via other terminals for temporary storage or for processing activities like (customs) checking and consolidation; (b) a weight and a volume (for consolidation decisions); (c) timing parameters like a load arrival time (i.e. the time at which order handling may start), an order due time and possibly earliest arrival times and latest dispatch times for some terminals to control the work load.

2.2. Primary process

The primary process consists of the cargo handling process and the AGV routing process, see Figure 2. Order handling starts when the cargo arrives at the terminal. Before preparing for transportation, possibly some other terminal handling is required like physical checks or administrative procedures. Next, cargo from several orders may be consolidated to increase the capacity utilization. Next, the (consolidated) load may have to wait in a buffer until both a loading dock and an AGV is available. Then the cargo can be loaded on the AGV and the AGV drives to the next destination terminal of the cargo. Note that AGVs only handle orders to transport cargo from a single origin to a single destination (so no pick-up and delivery en route). In the OLS case, this is partly a requirement because of customs regulations around Schiphol airport, partly due to handling restrictions in the automated equipment (standardized cargo units).

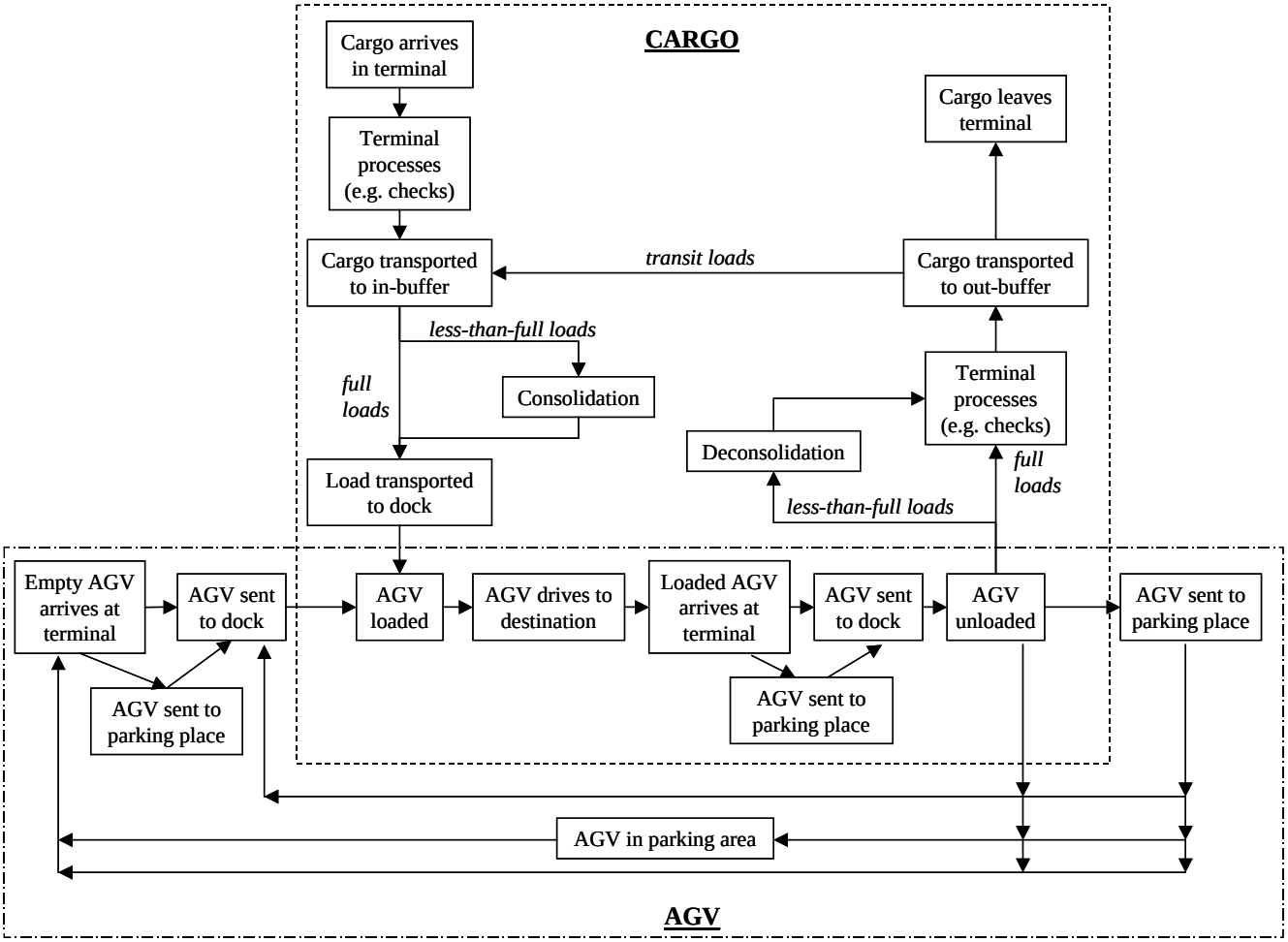


Figure 2. Physical primary process in an automated transportation network

When an unloading dock can be assigned upon arrival at the destination terminal, the AGV drives to this dock and the cargo is unloaded, otherwise the AGV can be parked in a parking place or wait at the terminal entrance. After unloading, the empty AGV can be sent to a loading dock, a parking place, another terminal or to a parking area. Consolidated loads are broken up according to the original transportation orders. Again, it is possible that other activities are required at the terminal. Then the cargo is moved to the out buffer and it leaves the terminal if it has reached its final destination. Transit orders are moved to the in-buffer for further transportation. The key performance indicator in this process is the on-time service percentage, i.e. the percentage of orders that is finished before the due time.

3. Logistics planning and control structure

3.1. Overview of planning decisions

A range of logistic planning and control decisions have to be taken in order to guide the orders through the system in time, see Table 1. We classify the decisions according to the scope and as static versus dynamic.

Scope	Decision	Description
Network	Network routing	Choose the route between two nodes (terminals, parking area) in the network
	Global AGV planning	Issue orders for empty AGV movements between terminals
Node (terminal, parking area)	Node routing	Choose the route between two locations (local parking areas, docks) within a node
	Order acceptance	Accept customer order requests for handling transport orders or not
	Consolidation planning	Select orders to be transported jointly by a single AGV
	Parking management	Select a parking location for an AGV within a node
	Order sequencing	Determine the sequence in which (consolidated) orders should be handled at a terminal
	Dock assignment	Select a dock for an order (load or unload)
	Empty AGV assignment	Select an AGV for a load order
	AGV dispatch	Determine the time at which an AGV should leave its current location in order to arrive in time at its destination within a node (dock, parking area)
	Load dispatch	Determine the time at which a load should leave the buffer in order to arrive in time at a dock to be loaded, taking into account internal transportation time
	Terminal access control	Determine when an AGV waiting for the terminal entrance may enter the terminal
	Two-way track control	Determine the times at which the driving direction on two-way tracks for traffic in both directions should be switched.
Track	Traffic control	Determine the time at which an AGV may enter a specific track section in order to avoid congestion or even dead locks
	Distance control	Adapt the AGV driving speed depending on the position and speed of other AGVs in order to avoid collisions

Table 1. Overview of logistic decisions

3.2. Scope of the decision: network, node or track.

Decisions at network level need network state information, decisions at node level need in principle only local (terminal) information, but multiple vehicles and orders are involved; at an even lower level, track decisions

only need information on the track state. Distributed agent-based control implies that each decision should be taken at the lowest level in the network hierarchy possible in order to achieve flexibility. Global AGV planning should be performed at network level, because it involves a priority setting for order departing from various terminals. Although the position of two-way track control is somewhat ambiguous (track or node level), we choose to model it at node level. The reason is that long two-way tracks as we encountered in our application (up to 10 minutes driving time) have a much higher impact on the system performance than simple traffic control for a single track section due to delays and vehicle batching.

3.3. Static versus dynamic control

Static control implies that the decisions are taken off-line, independent of the system state, whereas dynamic control depends on the system status at a specific point in time (availability of AGVs and docks, order status, etc.). Most decisions should be taken in a dynamic context (see also De Koster and Van der Meer [1998]), but (network and node) routing can be determined off-line using a shortest path algorithm. An alternative is dynamic routing taking into account local congestion, but this is only applicable if there are many alternative routes (which is particularly not true for network routing in our case). Also, a static variant of two-way track control can be considered, namely a simple periodic traffic light control rule (see van der Heijden et al [2001]). In the remaining part of this paper, we focus on *dynamic* control. Therefore, we will not further elaborate on network routing and node routing in the sequel.

3.4. Logistic decisions

In Figure 3 we show the relation between all relevant logistic decisions, indicated by rectangles. Recall that *static* decisions like network and node routing are excluded from the framework for dynamic control, because these decisions are taken off-line as mentioned before. Also, we exclude decisions at track level in this figure, because even advanced traffic control and distance control can be implemented independently of agent behaviour at node and network level, see Verbraeck et al. [1998].

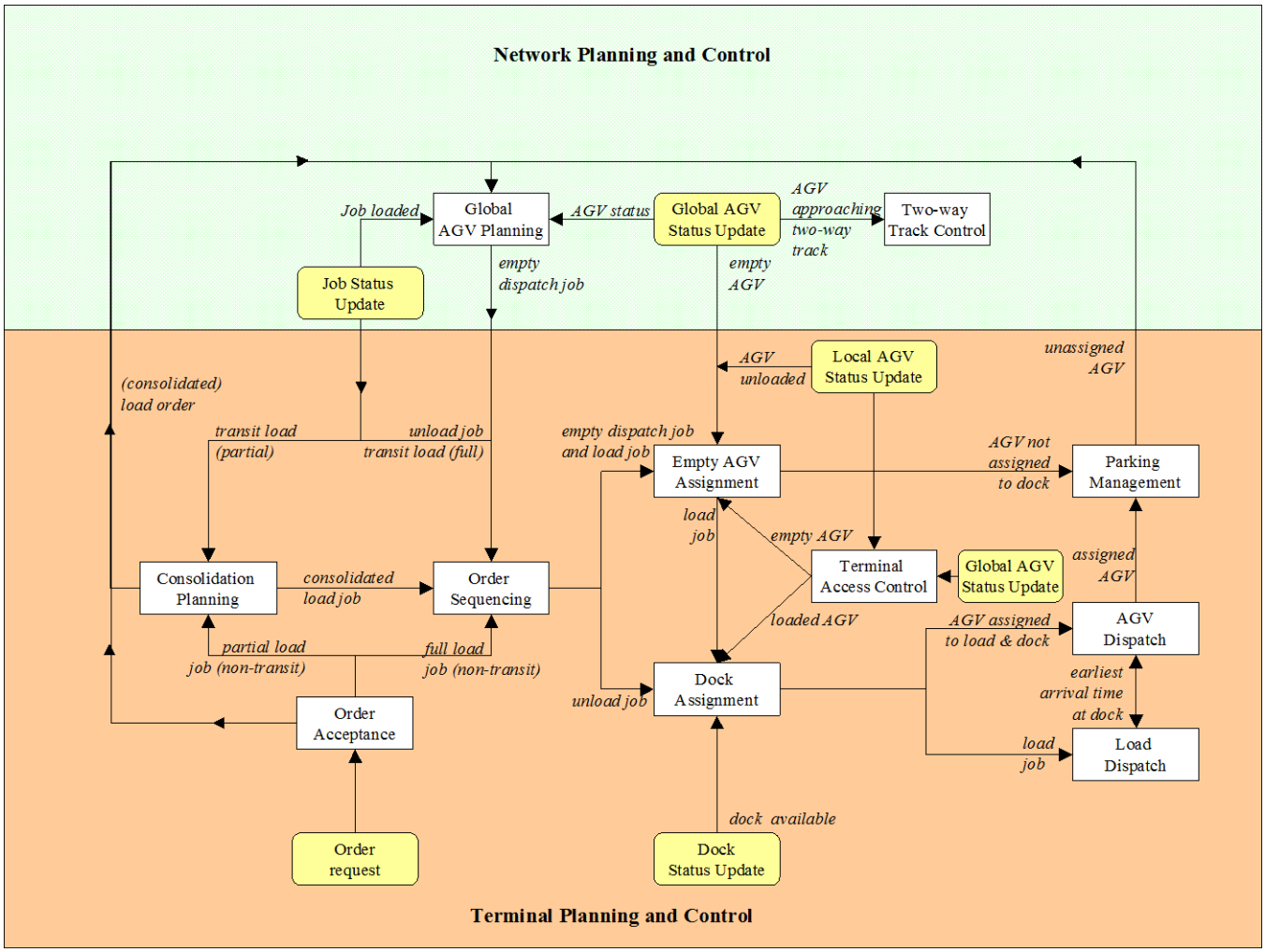


Figure 3. A framework for dynamic control and the mutual relations between logistic agents

The arrows in Figure 3 indicate triggers that invoke agent activities and show which information is exchanged at that time. Also status updates, indicated by rounded boxes, invoke *information flows* to agents and may trigger agent activities. We distinguish the following status updates:

- Order request: Request to transport a load with given characteristics (weight, volume, expected time available at departure terminal, routing along intermediate terminals, due time at destination).
- Order status update: Start or end of an activity in the order list (checking, physical consolidation, internal transport to dock, loading, unloading, consolidation).
- Dock status update: Switching between the statuses Free, Loading, Unloading.

- Global AGV status update (relevant at network level): order assignment (starting/finishing order) or position update (entering/leaving a node or a crucial network location like a two-way track, crossing or roundabout.
- Local AGV status update (relevant within a node only): starting/finishing an order; assignment of AGV to load, dock or parking assignment; local position update (i.e. terminal entrance / exit, parking lane or dock).

We should keep in mind that variants for the agents carrying out a specific control task may require more or less *detail* of the information (for example from status updates) and/or a different *timing* of information and triggers. Also, the *frequency* of status updates is a variable. For example, an AGV status update may occur when an AGV has (un)loaded or when it arrives at a terminal entrance, but it is also possible that the AGV position is recorded when an AGV passes some specific locations in the transportation network, so that more accurate information on expected arrival times at terminals can be provided. The agents should be able to deal with such variants.

4. Control objects: Local goal seeking heuristics and information exchange

In this section, we describe the individual agent behaviour in the local control framework (single task), for dynamic control activities. We select the relevant part of Figure 3 for each control object for sake of clarity. For each agent, we will describe its goal seeking behaviour, consisting of:

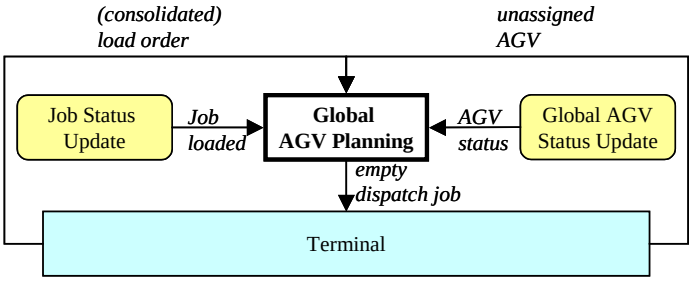
- the agent's task;
- the local goal, that is, the optimisation criterion;
- the goal seeking heuristic (if not trivial);
- the information needed;
- triggers to invoke the agents activities;

To prevent lengthy explanations, we will only describe the key characteristics of the agent's behaviour.

4.1. Global AGV planning

A task at network level is to distribute empty vehicles amongst nodes (terminals). In a local control setting, the agent issues orders to dispatch empty vehicles to terminals. Later on, it is decided at terminal level which AGV to dispatch when, depending on all order priorities (empty and loaded). We choose the goal to maximize the

number of orders for which an AGV is available in time. The corresponding goal seeking behaviour is as follows. An empty dispatch order consists of a destination (other terminal) and a priority, expressed by a *latest dispatch time*, i.e. the latest time that an empty vehicle has to leave the terminal in order to arrive on time at its destination. If too early arrival at the destination is not allowed (e.g. because the storage capacity at the destination is small), an order may be given an *earliest dispatch time*. Input to the decision process are (1) consolidated order lists from the terminals and (2) AGV status information (position, current order). The terminals confirm loading of orders to update the order list of global AGV planning.

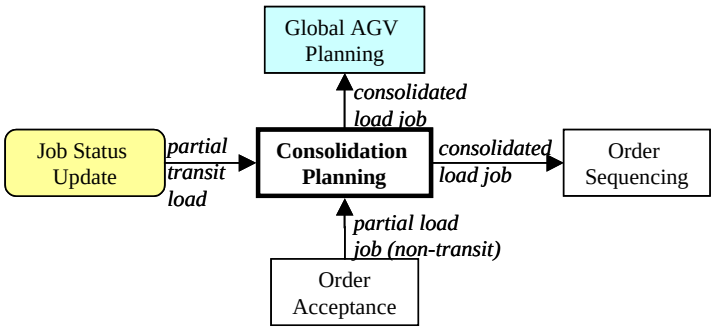


The control object sorts orders on their priorities and subsequently selects the AGV that can serve the order at the earliest instance. The outcome is used to issue *empty AGV dispatch orders* to terminals. So global AGV planning does *not* assign AGVs to orders, but merely uses order priorities to redistribute AGVs over terminals. The planning procedure can be triggered by any information update. Because of the high frequency of status and order list updates and the relatively long time required to execute empty dispatch orders, a periodic planning procedure is more appropriate. Van der Heijden et al [2002] discuss this control object in more detail and show that the planning frequency has only limited impact on the system performance if the frequency is not too low.

4.2. Order acceptance

The decision which orders to accept and which to reject is influenced by order throughput time requirements and resource capacity restrictions. Goal seeking criteria can be the probability that an order can be served in time, or the expected reward. It is not straightforward to develop a clever heuristic, because we have to deal with many interacting capacity restrictions. Because our aim is to analyse the system performance for various interacting sets of local control objects, we decided to accept all orders. Otherwise, the comparison between various control variants is disturbed by the fact that the order acceptance may generate different workload levels per variant.

4.3. Consolidation planning



Several small loads can be combined to a single AGV load to obtain transport efficiency. Given the restriction that AGVs only transport point-to-point without intermediate pick-ups or deliveries (see Section 2.2), the issue is to combine loads having the same origin and destination. Naturally, the consolidation agent is located at the departure terminal. Its goal is to minimize the number of AGV loads, given a restriction on the order priorities that may be combined. Such restrictions aim to avoid scheduling problems for other agents in a later stage.

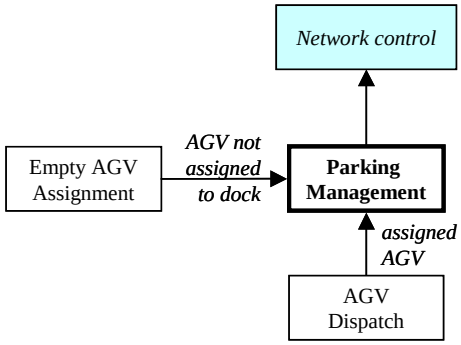
Its goal seeking behaviour is as follows. Load orders are sorted on destination and latest dispatch time. Loads are subsequently added to a consolidated order insofar the following restrictions are satisfied:

- all orders have the same destination;
- it is physically possible to combine the loads, i.e. both the total weight and volume are below AGV capacity;
- the latest dispatch times of the various orders are within a certain range *or* the slack of the combined order (= minimum of latest dispatch times - maximum of earliest dispatch times) exceeds some lower bound; the logic behind this rule is that we prefer to combine loads having similar priorities (latest dispatch time), but we allow the priorities to be different if there is still plenty of time to dispatch the orders.

As input, the agent needs an order list with physical characteristics (weight, volume) and a time window for the order starting time. The time window is determined from load arrival times, expected handling times and order due times. The agent may benefit from early information, because (a) transport requests usually arrive at order acceptance before the load actually arrives at the terminal and (b) information on transit orders can be provided when the transport to the current terminal has started. The agent communicates its planning results to order sequencing for further terminal scheduling and to the global AGV planning at the network level to plan empty AGV movements. We choose for periodic planning because information is updated frequently and order information is usually available some time before loads can be consolidated physically.

4.4. Parking management

Parking management assigns a parking location in a node to an AGV if it has to wait until it can load or unload at a dock. A node may contain several areas where AGVs can park, so that an appropriate assignment may reduce throughput times. Each area can be represented by a queue where AGVs cannot pass each other. Then, a bad assignment may cause that an AGV can be blocked when it is dispatched to the dock, because it is parked behind another AGV that cannot be dispatched yet. The parking agent should try to avoid this.



The parking agent is triggered if (1) an AGV has not been assigned to a dock yet, or (2) an AGV has been assigned, but it cannot be dispatched immediately because of the parking capacity at the dock is insufficient. An AGV for which no parking location can be found and that is already admitted to the terminal can be transferred to network control to find another solution (e.g. dispatch to central parking). The agent has the goal to minimize order delays caused by bad positioning for the next order and local congestion. A simple corresponding heuristic selects a parking area that has capacity left and that satisfies a local criterion mimicking the goal, such as:

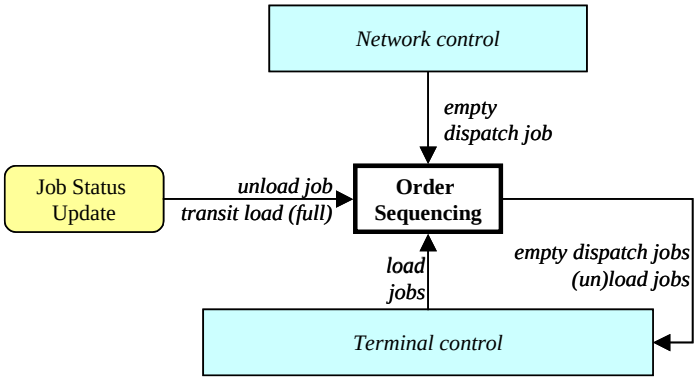
1. minimum distance from the current AGV position;
2. minimum distance to the dock to which the AGV has been assigned (if any);
3. minimum queue length in the parking area;
4. maximum available parking capacity.

Also, more advanced heuristics can be used. It depends on the internal node layout whether advanced heuristics are necessary. We chose for rule (2), followed by rule (3) if an AGV has not been assigned to a dock yet. If a parking cannot be found, the agent triggers global empty vehicle planning to search a location outside the node.

4.5. Order sequencing

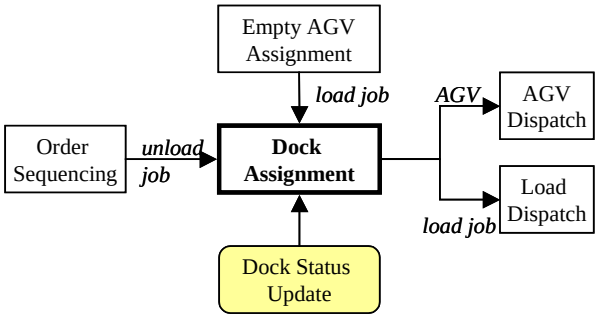
At a terminal, load orders, unload orders and empty AGV dispatch orders compete for AGV and/or dock capacity. The order sequencing agent sets priorities for local order processing. Its goal is to minimize the fraction of jobs that is started too late to guarantee that they will be finished in time. Hence its behaviour boils

down to a simple sorting procedure on the latest starting times of orders. Orders whose earliest starting time has not been reached yet are not transferred to empty AGV assignment and dock assignment yet. Earliest and latest starting times are derived from (consolidated) order due times and the estimated time at which the (consolidated) order will be ready for processing. Order sequencing triggers empty AGV assignment and dock assignment to search for resource capacities to handle the jobs.



4.6. Dock assignment

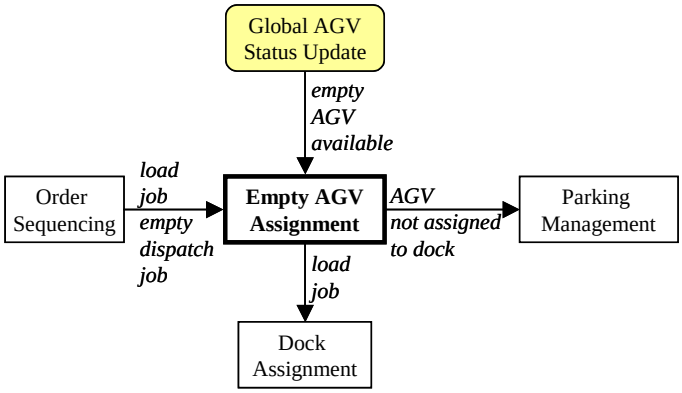
The agent assigns AGVs to docks within a certain planning horizon. Some planning ahead is necessary, because it takes time to move both AGVs and loads to the docks. On the other hand, planning too far ahead does not make sense, because updated information will reveal better dock assignment opportunities in the near future. Thus, a typical planning horizon is not much longer than the maximum transport time of AGVs and loads to the docks.



The agent has the goal to maximize the dock utilisation given the order priorities provided by order sequencing. The agent is triggered by the receipt of information that either an AGV with an (un)load order or a dock will be available within the planning horizon. The goal seeking heuristic assigns an AGV to a dock such that the highest priority order from order sequencing can start (un)loading at the earliest instance. The agent triggers the load dispatch and AGV dispatch agent to assure that both the load and the AGV will arrive in time at the dock. Usually, the docking time and the transport time to the dock cannot be predicted exactly. To cover stochastic deviations, at least one load storage location and at least one AGV parking place at the dock is recommended as buffers. As a consequence, dock assignment can provide both AGV dispatch and load dispatch with a *time window* between which the AGV and load may arrive to be served in time without causing congestion.

4.7. Empty AGV assignment

Triggered by an order arrival or an AGV becoming available, this control object matches an empty AGV with a load order or an empty dispatch order. Its goal is to minimize AGV waiting time, given order priorities provided by order sequencing. The agent roughly selects an empty AGV as follows:

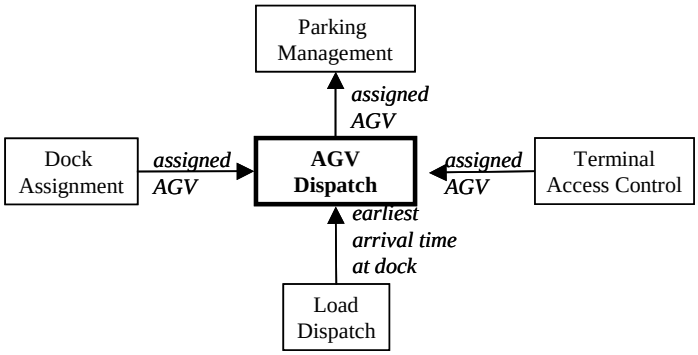


- Empty dispatch order: select the AGV closest to the terminal exit;
- Load order: select the AGV closest to the dock that will be available at the earliest point in time; ask the dock assignment for the latter information;

Information needed for this heuristic includes order information from order sequencing and a list with empty AGVs and the time and location of availability. If an assignment is found, the agent triggers dock assignment, otherwise it triggers parking management.

4.8. AGV dispatch

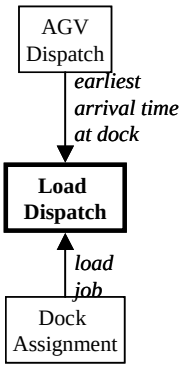
The agent decides when an AGV should leave its current location in order to arrive in time at a dock. Its goal is to maximize dock utilisation for jobs provided by the dock assignment agent, taking into account the parking capacity at the dock. It is triggered by dock assignment or by terminal access control. The goal seeking heuristic proceeds from the earliest and latest time at which the AGV is allowed to arrive at the dock (see Dock Assignment). The agent calculates the earliest and latest dispatch time from its current location using an offset for the travel time. If the AGV cannot be dispatched immediately, parking management is triggered to find a solution.



Triggers may also come from AGV and dock status updates. For example, the current loading job is finished somewhat later than expected, so that the AGVs scheduled later at the same dock can (or may be should) arrive

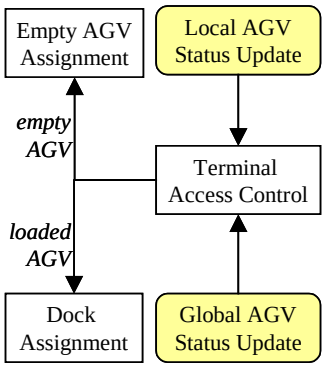
later. Rescheduling these jobs may influence parking occupation and terminal access. To avoid control system nervousness, the dispatch time windows are useful. As long as the scheduled load and AGV arrival time remains within the time window, it may be modified without consequences. Rescheduling only occurs if status updates show a large deviation between planning and realisation.

4.9. Load dispatch



This agent controls timely arrival of a load at a dock. Its goal is to minimize dock capacity loss due to late arrival of loads under the restriction of finite storage capacity at the dock. It calculates a dispatch time from the expected load time with an offset for the internal transportation time to the dock and a safety margin to cover uncertainties in docking time of preceding orders and internal transport time. It coordinates simultaneous arrival with the AGV at the dock with AGV dispatch.

4.10. Terminal access control

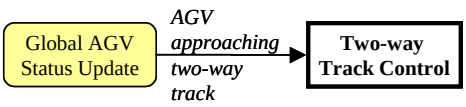


Congestion may occur if too many AGVs are driving simultaneously on a terminal. Therefore, this agents decides upon the time at which AGVs are allowed to enter the terminal. Its goal is to maximize the terminal throughput given a maximum expected delay caused by congestion. A corresponding heuristic only allows an AGV to enter the terminal if (1) a destination within the terminal (dock, parking area) can be found where the AGV can temporary park without hindering other traffic (parking management, dock assignment), and (2) the total number of AGVs driving on the terminal does not exceed some predefined threshold. It is triggered by AGV status updates. The agent bases its planning on an overview of the expected times at which AGVs will arrive at the terminal entrance and the expected times at which AGVs will leave the terminal.

4.11. Two-way track control

A single track shared by traffic in two directions may be part of the network (Section 2.1). Batches of AGVs have to pass this track alternately, and hence an agent has to decide about the points in time when the driving

direction has to be switched. The goal that we choose is to minimize average AGV waiting time. As two-way tracks can lead to significant AGV waiting times and thus affect on time service percentages, intelligent two-way track control is appropriate. Therefore, Ebben et al [2001] develop some dynamic control rules, taking into account information on queue lengths at both sides of the two-way track or even expected arrival times of AGVs. As the dynamic control rules outperform simple periodic rules, we include information flows about AGV arrivals in our logistic control framework. Such information may be provided by status updates of AGVs driving towards the two-way track. Upon each status update, the control agent is triggered.



4.12. Traffic control

A basic requirement for automated transportation networks is that AGVs can drive smoothly along the track system without significant delay because of congestion and without encountering deadlocks. This is not an easy task, especially on terminals where many AGVs have to manoeuvre in a small area. A suitable option is to embed an advanced traffic control system called TRACES that has been developed by Evers and Lindeijer [1999]. This control system has been refined and tested by Verbraeck et al [1998, 2000]. The basic idea is that conflicts between AGVs can be avoided using an advanced ticketing system. The AGV requests access to conflict locations, such as junctions or crossings, at local semaphores. If successful, the AGV receives a ticket, which it returns after leaving the conflict location. The number of tickets controls the traffic density on a track. Traffic Control operates independent of other control objects at terminal and network level.

4.13. Distance control

To avoid collisions, a minimum distance between AGVs should be maintained to account for deceleration time in case of emergencies. This can be achieved by continuous measurement of the distance from an AGV to other AGVs using sensors aboard. Based on these measurements, speed adjustments can be made. Although this seems a trivial task at first sight, it is not straightforward to maintain smooth driving behaviour. For details we refer to Verbraeck et al [1998, 2000], who also show that Distance Control can be executed independent of other control objects such as Traffic Control.

5. Case: The OLS project

5.1. Key characteristics of the OLS

As mentioned before, our research has been motivated by the development of an automated transportation network using AGVs around Schiphol Airport, Amsterdam. Here we discuss the main system characteristics.

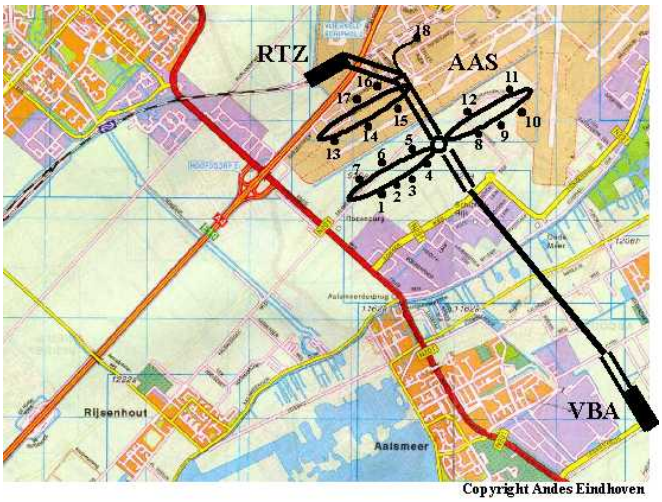


Figure 4. Layout option for OLS-Schiphol

Figure 4 shows one of the layout options under consideration. The automated transportation network connects three main locations, namely the rail terminal at Zwanenburg (RTZ), the flower auction at Aalsmeer (VBA) and Amsterdam Airport Schiphol (AAS; the numbered dots indicate some smaller terminals at Schiphol). All connections are tubes that will mainly be constructed underground. The layout contains 2 two-way tracks, namely a long one between AAS and VBA and a short one at AAS (leading to terminal 18). Large AGVs are able to carry cargo units up to a 10 ft. main-deck aircraft pallet with a maximum weight of 3500 kg. The average AGV speed is expected to be 6 m/s on long distances between terminals and between 2 m/s and 4 m/s on the terminals. The terminal docking capacity varies between 1 dock (the smallest AAS-terminal) and 14 docks (rail terminal). The average AGV (un)loading time equals 1 minute for full AGV loads and 3 minutes for consolidated loads. The network should be able to transport about 3.6 million tons of cargo in 2020. Order

patterns can be highly imbalanced in time and show sharp peaks on specific days and hours of the day, depending on the route. Throughput time requirements may vary between orders; some rush orders should be handled in les than 1 hour. To handle this cargo, we need around 300 AGVs in the year 2020. For more details on the system characteristics, we refer to Ebben [2001].

5.2. Simulation modelling in eM-Plant

To test the logistic control framework, we constructed an object oriented simulation library. In this library, we modelled both physical objects like terminals, tracks, docks, parking areas, AGVs and loads as well as information objects (e.g. order and equipment status lists) and control objects as specified in Section 4. We chose eM-Plant (Tecnomatix [2001]) as simulation tool because of its object orientation and flexibility. Of course, other suitable object oriented simulation software is available on the market. Still the construction of an object-oriented library is not as straightforward as theory suggests. The frequent design changes in the OLS-project are a test case for library flexibility. Especially the maintenance of information objects and reliable information flows is critical. Exceptional events may cause inaccuracies in system information, which can easily lead to deadlocks. Obviously, control objects do not function properly if the input information is not fully correct. These problems especially arise as a consequence of equipment failures (see also Section 6).

We used our object oriented simulation library to evaluate a wide range of design options and also as a test bed for our logistic control framework. The focus of our project is on logistic control at network and node level as displayed in Figure 3. To reduce model complexity and simulation run times, we implemented only a simple version of AGV behaviour in our simulation in the sense that AGVs can accelerate and decelerate instantaneously. As a consequence, advanced Traffic Control and Distance Control is less useful. Therefore we use a simplified version, taking into account conflict avoidance for major conflict areas only (crossings, roundabouts, terminals). The number of AGVs that is allowed to drive on such a conflict location simultaneously is bounded, see terminal access control for an example. Further, mutual minimum distances are set at some specific critical locations only (entrance and exit of terminals and two-way tracks). To avoid too optimistic estimates of system performance, detailed terminal simulations have been carried out to estimate a

statistical relation between average AGV speed on terminals and the traffic density (cf. Verbraeck et al. [1998, 2000]). These statistical relations are part of our model to correct the average AGV speed for traffic density.

In order to improve the system performance using look-ahead heuristics, we pass information between control objects as early as possible. For example, the two-way track control agent may use information about expected AGV arrivals, and the consolidation control agent may be informed when transit orders are loaded on their way to the transit terminal. Obviously, this still allows to use basic control rules that do not use early information.

6. Testing the modification flexibility of our logistic control framework

During the OLS project, we faced several changes in requirements to the logistic control system. Major issues were layout changes, equipment failures, AGV energy provisioning and finite terminal capacities (docking, parking area, storage). In this section, we show how these issues can be incorporated in our framework (6.1-6.4). These issues served as a test for the modification flexibility of our framework.

6.1. Layout changes

In the period between 1997 and 2001, layout changes were frequently proposed. Major changes include the trajectory, the number of terminals, the introduction of two-way tube sections and new terminal layouts. Modifications in trajectory and numbers of terminals did not have impact on our control framework, because parameters change only. The introduction of two-way tube sections had consequences for our control framework. First, we needed an additional agent to manage the two-way tube sections (see 4.11). Also, the travel times between terminals became less predictable, because it is not known beforehand if, and how long an AGV has to wait for access to the two-way track sections. Here the local control framework appeared to be powerful, because the existing agents did not need serious modifications. In fact, it was sufficient to modify the latest dispatch times of orders by including an offset for the uncertainty in travel time. Regarding terminal layouts, these required only parameter modifications in some cases (number of docks, parking capacity at the dock, terminal length, etc.). In other cases, capacity issues came up, such as finite storage capacity. This required more effort to modify our framework. We refer to Section 6.4 for details.

6.2. Equipment failures and failure management

In the OLS project, equipment failures became an issue after the first versions of our logistic control framework and the corresponding simulation model had been designed. The major failure modes are docks and AGVs. A failed AGV can block other traffic and may cause a dramatic drop in system performance if the failure is not handled properly. Therefore, it is sensible to include dedicated failure managers to deal with such failures.

6.2.1. Dock failure management

A dock failure has only limited impact on the overall system performance. It causes significant delays, but only to a few orders, namely the orders that have been assigned to the dock upon failure. In case of failure, the load being handled at the failed dock will usually be blocked. The issue is mainly proper information maintenance, since the control actions are basic:

- load and unload orders are transferred to dock assignment to find another dock
- the corresponding AGVs are transferred to AGV dispatch for timely dispatch to their new destination

6.2.2. AGV failure management

Various AGV recovery options are possible. One way to remove a failed AGV is to tow it away using a special *recovery vehicle* to some repair station where it can be repaired without blocking other traffic. Such a recovery vehicle should approach the failed AGV from the front, hence it has to drive in the opposite direction over the track system to the failed AGV. This means that the tracks in between should be freed first for the recovery vehicle to do its order. An AGV failure manager controls the AGV failure recovery process, consisting of the following steps, cf. Figure 5:

1. Selecting the appropriate recovery vehicle, for example the recovery vehicle closest to the failed AGV;
2. Selecting the route of the recovery vehicle, usually the shortest route to the failed AGV;
3. Clearing the route of the recovery vehicle: Access to all tracks on the route is blocked (via traffic control);
4. Driving of the recovery vehicle to the failed AGV; as soon as the recovery vehicle has passed a track, the track is freed and other AG Vs are allowed to access the track again (via traffic control);
5. Towing the failed AGV to a repair station (in our application it is always located at some terminal);

6. Handling the load when the failed AGV was loaded: the load is transferred to order acceptance at the terminal where the failed AGV is under repair; the order cannot be refused, so order acceptance merely acts as a gateway to terminal management (consolidation planning and order sequencing);
7. Repairing the failed AGV;
8. Returning the repaired AGV to the system: the AGV is transferred to empty AGV assignment;

Figure 5 below shows the various decisions in a single integrated control block for sake of simplicity. Of course, the various control decisions within this block (1, 2, 3, 4, 6 and 8) can be modelled by separate control objects.

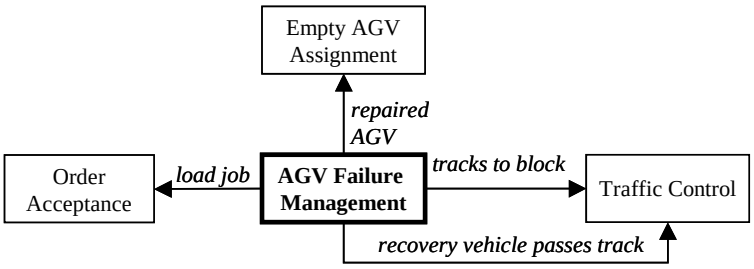


Figure 5. AGV Failure Management related to the other local control objects

This implementation of failure management requires special communication with traffic control, but apart from that it operates independent of the other agents. In our experience, the main pitfall is inaccurate maintenance of information in the case of failures. For example, the failed AGV should temporary be removed from local and global AGV status lists to avoid scheduling an AGV that is actually not available. Because of the wide variety in failure locations, a careful examination of all local and global status lists requiring an update is needed.

6.2.3. Impact on system performance

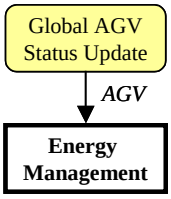
Inclusion of failure behaviour and management in our framework and simulation model offers the opportunity to gain insight in the impact of AGV failure rates on system performance. As an illustration, we show in Table 2 the relation between the on-time service percentage of orders and the mean time between failures (MTBF) of docks and AGVs. Note that the MTBF is measured in active hours (i.e. failures never occur if a resource is not

active). The table illustrates that AGV failures have much more impact than dock failures, but that occasional AGV failures still allow to attain a high overall on-time service percentage.

AGV MTBF	Dock MTBF --	Dock MTBF 25 hour
--	97.2	97.0
1500 hour	96.2	95.4
1000 hour	95.2	95.3
500 hour	93.2	92.7

Table 2. Effect of the MTBF of AGVs and docks on the overall on-time service percentage

6.3. Energy management



For energy provisioning, various options are available, such as electric wires along the tracks (in the tubes for the OLS) and rechargeable batteries. The latter option requires additional logistics control. In the OLS project, we studied the option to replace batteries at some *battery stations*. Next, these batteries can be recharged off line. This requires various additional decisions, namely *when* and *where* to replace batteries. At various occasions, we have to check the remaining energy level of the batteries. Such a trigger can be added to global AGV status update. For details, we refer to Ebben [2001].

Energy management uses the following information for its decisions:

- the estimated distance that the AGV still can drive with the remaining energy
- the distance to the nearest battery station
- the expected loss of time if the AGV is rerouted via the battery station; this depends on the detour that the AGV should make, but this time should be corrected for waiting time that will be incurred anyway (for example if the AGV is waiting in front of a terminal or two-way track);
- the distance to the next location of a global AGV status update and the distance to the closest battery station from there;

For the decision *when* to replace batteries, we can use three principles:

1) Replace a battery when its charge is not sufficient to perform the next job;
2) Replace the batteries of all AGVs before a peak period; this is only attractive from the point of availability when a fully charged battery is sufficient for this peak; the latter is not true for the OLS case;
3) Replace a battery when an opportunity arises, i.e. when an AGV has to wait anyway during some time that is sufficient to swap the battery:
    a) when an empty AGV is temporarily parked at a terminal waiting for a new load job;
    b) when an AGV has to wait at the entrance of a two-way track or terminal;
    c) when an AGV is on its way to or is at a central parking area;
    d) when an AGV is loading or unloading at a dock; this is only a reasonable option if battery swapping takes less time than (un)loading and the physical activities can be carried out simultaneously;

Opportunity replacement is attractive because it reduces the operational AGV time lost by battery changes but it also leads to more battery swaps, and so the battery swap stations should have higher capacity. Also, opportunity replacement should always be combined with the first control option, because it cannot be guaranteed that an opportunity always arises before the battery is empty. In the OLS case, we combined the options 1, 3b and 3c.

For the decision *where* to replace batteries, several heuristic rules are possible:

1) Nearest battery station;
2) Farthest reachable battery station on the current route;
3) First battery station encountered on the current route;
4) Battery station that leads to minimum delay;

For the OLS case, we chose for option 2 in order to reduce the number of battery swaps. Because of the loose coupling to other control objects, the other heuristics can be implemented without major consequences for the control framework. Note that it is possible that an opportunity arises while driving to the farthest reachable battery station on the current route. In that case, the opportunity has priority.

Energy limitations may influence the behaviour of other agents as well. For example, empty AGV assignment may select an AGV that has not sufficient charge to transport that specific order, whereas no battery stations is

close to the route. Then, it is reasonable to adjust empty AGV assignment by using the remaining energy as a second criterion. If we expect that an AGV has to change its battery on the next route, it is best to assign this AGV to an order to be transported over a route where changing a battery is possible without serious delay.

Using the modifications to our logistic control structure, we are able to analyse the impact of various options for energy provisioning: charge rails and battery swapping using various battery types. In Table 3, we give some key results of the simulation analysis. For all cases, we obtained similar on-time service percentages; the main differences are the resource requirements, i.e. the number of AGVs. Using such results, a financial trade-off of various energy provisioning systems can be made. For details, we refer to Ebben et al [2001].

Battery	Swap time (min.)	Number of battery stations	Number of AGVs
Charge-rails	n.a.	0	360
Lead-acid	5	1	380
	5	3	370
Nickel-cadmium	5	1	390
	5	3	380
Nickel-cadmium quick-charge	5	1	410
	5	3	380
Lead-acid	1	3	360
Nickel-cadmium	1	3	360
Nickel-cadmium quick-charge	1	3	370

Table 3. Key results of simulation analysis of energy systems

6.4. Dealing with finite terminal capacity

The control agents as specified in Section 4 do not take into account capacity restrictions at the destination when scheduling orders. Finite terminal capacity arises from (1) finite docking capacity, (2) finite AGV parking capacity and (3) finite cargo storage capacity. Ignoring these restrictions may cause significant waiting times of AGVs when arriving at the destination terminal.

We can extend the logistic control framework by introducing some new integrated decision units. We extend the set of control objects at terminal level with two additional objects providing coordination *between* terminals:

- *Capacity Management*, an agent that receives requests to handle unload and transit orders and that decides upon the timing and order in which these requests can be granted;
- *Order Release*, an agent that selects orders in the order provided by Order Sequencing and communicates with Capacity Management on the order and timing of order release;

This kind of coordination requires that we should introduce information exchange between terminals.

6.4.1. Capacity Management

To take into account terminal capacities for order scheduling, we need to maintain information on resource utilisation (docks, parking lanes, storage), see Figure 6. Given known orders to be processed, we can derive *resource utilisation profiles* showing the estimated utilization of docks, parking lanes and cargo buffers as function of time. These profiles are updated by the relevant status updates. Also, AGV Dispatch provides information when an AGV will arrive at a dock for (un)loading. Finally, the capacity management agent provides information to the profiles upon acceptance of an unload request.

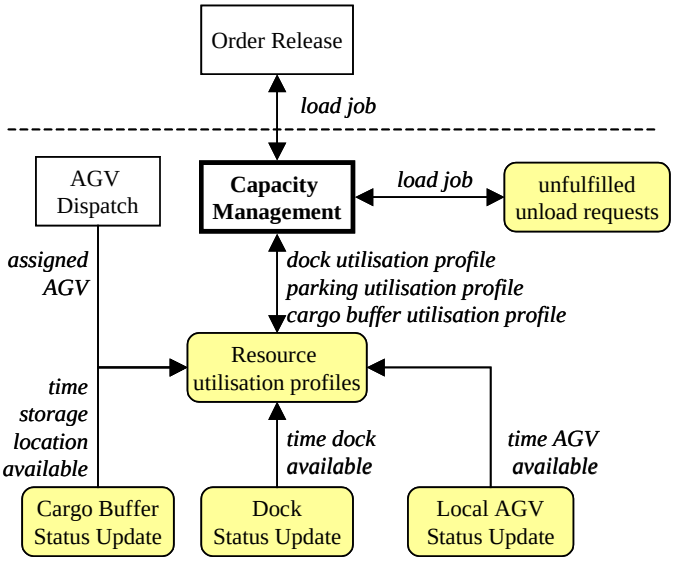


Figure 6. Embedding Capacity Management

The agent combines the resource utilisation profiles with information on unfulfilled unload requests, i.e. previous requests from other terminals (by order release, see below) that have been rejected so far. A simple

heuristic is as follows. An unload request is accepted if it has the minimum latest dispatch time (LDT) of all orders on the list with unfulfilled unload request. Note that we do *not* set priorities on due times, because orders with an earlier due time but a later LDT still have more slack to be handled. Next, we search the earliest instance at which this unload request can be handled by examining subsequently the cargo storage profile, the docking profile and the parking profile. That is, we check capacities backwards in the logistic chain. Capacity management grants permission for the selected order to be started with a given earliest arrival time at the terminal. This information is sent to the order release agent at the terminal where the order should be started.

6.4.2. Order Release

Finite terminal capacity prevents that orders are processed as determined by the order sequencing agent. Therefore, the order release agent is introduced that determines order release times based on information from order sequencing and permission granted by capacity management at the destination terminals, see Figure 7.

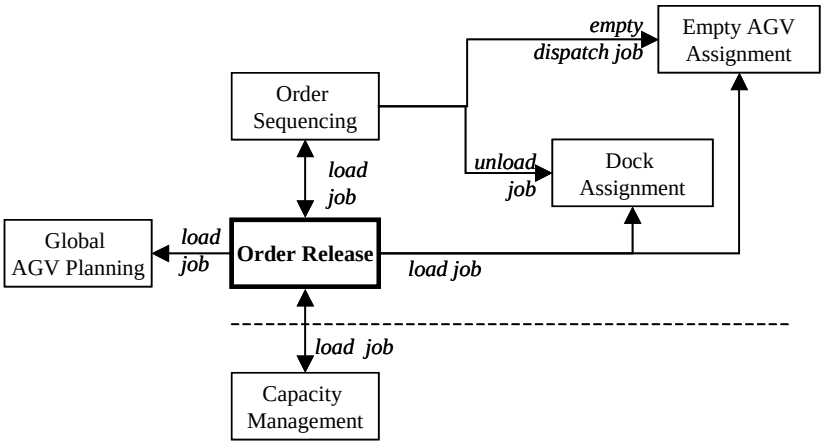


Figure 7. Embedding Order Release

As a consequence, order sequencing has to be modified. Unload orders and empty dispatch orders are still passed to the empty AGV assignment agent and the dock assignment agent respectively, but load orders are passed to the order release agent. Based on the communication with the capacity management agent at the destination, order release determines the earliest starting time and passes the order with this attribute to empty

AGV assignment and dock assignment. Also, the communication with global AGV planning has to be altered. When executing order acceptance and consolidation planning, it is not known yet when a load order will start, and hence it is also not known yet when an empty AGV is needed. Therefore, this trigger is broken and replaced by a trigger from order release to global AGV planning. Note that early information on planned order release is required to prevent delay, because global AGV planning usually needs some time to supply empty AGVs from other areas in the network in the case of a local shortage.

6.4.3. Numerical illustration

In the OLS system, the rail terminal has limited storage capacity. As a consequence, orders to the rail terminal have to be postponed sometimes. We investigated the effect of introducing these capacity restrictions. The control rules were adapted to take these limitations into account (cf. Ebben, 2001). In Table 4 we see that this capacity restriction results in a need of 20 additional AGVs.

Number of AGVs	With capacity restriction	Without capacity restriction
150	53.2	84.8
160	75.2	99.1
170	84.4	99.8
180	98.8	99.9

Table 4. On-time service percentages with and without a storage capacity restriction on the rail terminal

7. Testing integration flexibility

In our local control framework, a control agent is assigned to every single decision. The framework provides some flexibility to improve the logistic performance by integrating local control objects without altering the information interface. In this section, we give an example.

7.1. Integral Vehicle Planning

We discuss the integration of global vehicle planning and order sequencing. If these two agents are separated, the distribution of vehicles is not fully controlled at network level. As the order and timing of orders to be dispatched is controlled at terminal level, it is not sure how the vehicles will be distributed over time if no empty

dispatch orders are issued by global vehicle planning. In a local control context, this problem can be reduced by issuing empty vehicle dispatch orders ignoring future arrivals of full AGVs. At terminal level, empty dispatch orders may be deleted if a loaded AGV is dispatched to the same destination in time.

This can obviously be improved if the sequence of load orders at each terminal can be controlled at network level, i.e. by integrating global vehicle planning and order sequencing of all terminals in the network. We developed a serial scheduling algorithm for such an *Integral Vehicle Planning* agent, cf. Van der Heijden et al. [2002]. Input to the combined object is an aggregate order list and AGV status list, output is an order list per terminal, consisting of load orders, unload orders and empty dispatch orders. Because both the input and the output are the same as the combined output of the underlying local control objects, it smoothly cooperates with the rest of the logistic control framework. That is, the assignment of loads to empty AGVs and to docks, the dispatch of loads and AGVs etc. can still be controlled as before.

Still, the serial scheduling method does not take into account (1) uncertainty in travel times caused by unpredictable delays at two-way tubes and (2) coordination between successive routes for orders that have to pass one or more intermediate terminals between departure and destination terminal, see Section 2. To account for delays at the two-way tracks, the global AGV planning agent and the two-way track control agent should be integrated. Although this is possible, it complicates the logistic control.

7.2. Numerical illustration

We developed various AGV planning methods, each having its own span of control and information requirements. Next to the local control rule as discussed in Section 4, we also studied a simple first-come first-served (FCFS) approach and an integrated serial scheduling method as mentioned above. All these methods were tested with the simulation library using three different order patterns corresponding to a Monday a Tuesday and a Friday, see Table 5 for the key results. We refer to Van der Heijden et al [2002] for more details and some other control variants.

Day	AGVs	Overall service level		
		FCFS	Coordinated local control	Integrated control (serial scheduling)
Monday	120	97.1	99.8	100
Tuesday	185	89.9	99.1	98.9
Friday	165	93.2	99.7	99.6

Table 5. On-time service percentages for several variants of global vehicle planning

In Table 5, we see that coordination is required to achieve acceptable service levels. FCFS clearly yields an inferior performance. The integrated planning approach only outperforms a coordinated local rule with respect to service levels in difficult cases. An advantage of the integrated planning approach, which holds for all cases, is that outside the peak less empty kilometers are required to achieve comparable service levels.

7.3. Discussion

In the example given, integration of control objects appears to fit very well in our logistic control framework. However, our experience is that this is not always that easy. The main bottleneck for integration flexibility is the information infrastructure: integral planning usually requires more information (input) and leads to different planning overviews (output) than local control agents that use some simple look-ahead heuristics. This means that the information infrastructure should be prepared for control agent integration later on. This is not an easy task. In other words, we found that our framework has some integration flexibility indeed, but that this is much more difficult to achieve than the modification flexibility as described in the previous section.

Apart from complexity issues, one may wonder whether it is feasible in practise to integrate control agents, because various organizations may be responsibility for the various decisions. For example, order scheduling is a task at terminal level (freight forwarders, logistic service providers) whereas global vehicle planning is probably part of the business of the AGV operation company. Therefore, integration of control objects is plausible at terminal level (e.g. integrated order sequencing, order release, empty AGV assignment, dock Assignment, AGV dispatch and load dispatch), but less at the interface between terminal and network level.

8. Evaluation and conclusions

As mentioned in the introduction, we had four research goals. Here we will discuss our results on these issues.

First, we aimed to *develop a well structured framework for logistic control*. We have presented this agent-based framework in Section 3. Instead of pricing mechanisms, we used another well-known principle to organize cooperation between local agents: goal seeking look-ahead heuristics based on environmental state awareness. We focused on such information exchanging local control objects without explicit pricing strategy for the following reasons. First, a significant number of decisions have to be taken within the same unit (e.g. all planning and scheduling decisions at the same terminal). We feel that bidding and negotiations are not natural requirements for a distributed control concept if agents operate within the same economic units striving for the same target performance. Then, other ways of cooperation using information exchange are simpler because a pricing strategy is not needed. Second, our focus was definitely also on the integral performance of collaborating heuristic-based logistic control agents. For example, we wish to avoid empty repositioning of AGVs in view of effective capacity utilization and energy efficiency. Because of the complex problem structure that we deal with, developing and tuning of bidding procedures is a research challenge in itself. Heuristics are much more straightforward related to the overall goal.

Second, we defined a set of *goal seeking heuristics* for the logistic control agents in Section 4. We implemented these heuristics in an object oriented simulation model and tested it in a case study: the design of an automated transportation network around Schiphol Airport, Amsterdam. Several redesign phases were required to arrive at the flexible framework and the corresponding goal seeking heuristics as described in this paper. Although our framework seems relatively simple at first sight considering the variety of decisions to be taken, it is difficult to construct a flexible framework of loosely coupled modules that still yields a good logistics performance. We found that we had to make many several adjustments to our approach during the subsequent research phases. The object oriented design approach surely contributes to the construction of a flexible logistic control framework and its implementation in a simulation model. We believe that this would not have been possible if we would have started with a heavy integrated scheduling system from the beginning.

Third, we wished to *test the modification flexibility* of our framework, i.e. the ability to adjust to a modified system structure and to modified processes. The OLS project provided an excellent test case. During the system design phase in a project with many uncertainties like OLS, it is hard to foresee modifications and extensions to the logistic system, its control and its information requirements. Therefore, it is hard to anticipate. Still, we managed to implement modifications relatively quickly and efficiently (see Section 6 for some examples). Our experience is that the modification flexibility of our logistic control framework with local agents is high.

Fourth, we *tested the integration flexibility* of our framework, i.e. the ability to combine the control tasks of various agents into one overall agent. We succeeded to include some integral control agents in our framework, see Section 7 for an example. However, we found that the information and communication structure can be a bottleneck for the integration of local control agents. It is necessary, but also hard to foresee which integration will be suitable in the future. Therefore, we found that serious modifications of the information and communication structure were needed to facilitate integration. Therefore, we feel that a local control framework is more practical. Besides, there is evidence that integrated control and advanced algorithms do not necessarily yield significant performance improvements, see Van der Heijden et al [2002] for global vehicle planning and Ebben et al [2000] for two-way track control. Future developments will show whether our framework is flexible enough to cope with other modifications and extensions. At the moment, one may think of developing intelligent order acceptance procedures and integrating control objects at the terminals.

References

1. Booch, G. (1994), *Object-oriented analysis and design, with applications*, Addison-Wesley, Reading.
2. Damen, J.T.W. (2001), "Service-controlled agile logistics", *Logistics Information Management* 14 no 3, 185-195.
3. De Koster, R, and J.R. van der Meer (1998), "Centralized versus decentralized control of internal transport, a case study", in: *Advances in Distribution Logistics*, B. Fleischmann et al. (eds), Springer Verlag, Berlin, 403-420.

4. Ebben, M.J.R. (2001), *Logistic control in automated transportation networks*, Ph.D. Thesis, Twente University Press, the Netherlands.
5. Ebben, M.J.R., D.J. van der Zee and M.C. van der Heijden (2001), "Dynamic traffic control in one-lane, two-direction AGV systems", working paper, University of Twente, The Netherlands (submitted for publication)
6. Espinasse, B., L. Cloutier, and P. Lefrancois (1998), "Globalization of manufacturing in the digital communications era of the 21st century: Innovation, agility and the virtual enterprise". *Proceedings of the tenth international IFIP international conference PROLAMAT 98*, Trento, Italy.
7. Evers, J. and D.G. Lindeijer (1999), *The agile traffic-control and engineering system: TRACES*, working paper, TRAIL, Delft University of Technology, the Netherlands.
8. Evers, J.J.M., L. Loeve, and D.G. Lindeijer (2000), "The service-oriented agile logistic control and engineering system: SERVICES", *Logistics Information Management*, 13 no. 2, 77-90.
9. Heijden, M.C. van der, M.J.R. Ebben, N. Gademann and A. van Harten (2002), "Scheduling vehicles in transportation networks", to appear in *OR Spektrum*.
10. Heijden, M.C. van der, M.J.R. Ebben, and A. van Harten (2001), "Waiting times at periodically switched one-way traffic lanes: A periodic, two-queue polling system with random setup times.", *Probability in the Engineering and Informational Sciences* 15, 495-517.
11. Jennings, N.R. and M.J. Wooldridge (1998), *Agent technology: foundations, applications, and markets*, Springer, Berlin.
12. Jennings, N.R. (2000), "On agent-based software engineering", *Artificial Intelligence* 117, 277-296.
13. Kephart, J.O., J.E. Hanson and A.R. Greenwald (2000), "Dynamic pricing by software agents", *Computer networks* 32, 731-752.
14. Qinghe, H., A. Kumar and Z. Shuang (2001), "A bidding decision model in multiagent supply chain planning", *International Journal of Production Research* 39 no. 15, 3291-3301.
15. Tecnomatix (2001), *eM-Plant reference manual* 5.5, Tecnomatix Technologies.

16. Verbraeck A., Y.A. Saanen, and E. Valentin (1998), “Logistic Modeling and Simulation of Automated Guided Vehicles”, in: A. Bargiela, E. Kerckhoffs (Eds.), *Simulation Technology: Science and Art*, SCS, San Diego CA, 514-519.
17. Verbraeck, A. E.C. Valentin and Y.A. Saanen (2000), “Simulation as a real-time logistic control system: AGV control with Simple++”, in: M. Rabe et al (Eds.), *Simulation in production and control*, Berlin, p. 245-255.
18. Wellman, M.P., W. E. Walsh, P.R. Wurman and J.K. MacKie-Mason (2001), “Auction protocols for decentralized scheduling”, *Games and Economic Behaviour* 35, 271-303.
19. Zee, D.J. van der (1997), *Simulation as a tool for logistics management*, Ph.D. Thesis, University of Twente, the Netherlands.
20. Zeigler, B.P. (1990), *Object-oriented simulation with hierarchical, modular models – Intelligent agents and endomorphic systems*, Academic Press, San Diego.