The world is becoming increasingly linked, integrated, and complex. Globalization, which arrives in waves of increasing and decreasing usage, results in a permanently dynamic environment. Global supply networks, logistics processes, and production facilities try to follow these trends and – if possible – anticipate the volatile demand of the market. This challenging world can no longer be mastered with static, monolithic, and inert information technology (IT) solutions; instead it needs autonomic, adaptive, and agile systems – living systems. To achieve that, new systems not only need faster processors, more communication bandwidth, and modern software tools – more than ever they have to be built following a new design paradigm. As determined by the industrial and business environment, systems have to mirror and implement the real-world distribution of data and responsibility, the market (money)-driven decision basis for all stakeholders in the (real or virtual) market, and the goal orientation of people, which leads to on-demand, loosely coupled, communication with relevant partners (other players, roles) based on reactive or proactive activities.

This chapter provides an insight into the core challenges of today’s dynamics and complexity, briefly describes the ideas and goals of the new concept of software agents, and then presents and discusses industry-proven solutions in real-time environments based on this distributed solution design.

The following examples are discussed in detail in this chapter, covering the solution approach, challenges, and customer demand as well as relevant pros and cons:

- An autonomic machine control system applied to the adaptive control of a modular soldering machine. The particular case is concerned with the
Part C | Automation Design: Theory, Elements, and Methods

creation of a novel modular production machine with an integrated distributed agent control system, which has been sold worldwide since the middle of 2008. The agent model is described in terms of the specific customer requirements and the advantages of the approach.

- A solution to real-time road freight transportation optimization using a commercial multiagent-based system, LS/ATN (living systems adaptive transportation networks), which has been proven through real-world deployment to reduce transportation costs for both small and large fleets. After describing the challenges in this business domain and the real-time optimization approach, we discuss how the platform is currently evolving to accept live data from vehicles in the fleet in order to improve optimization accuracy. A selection of the predominant pervasive technologies available today for enhancing intelligent route optimization is described.

Both examples reflect their specific history and background, which motivated the customer and the developers to apply an autonomous automation approach.

Although software agents are a core principle of the autonomous automation examples in this chapter we only touch this field slightly, as other chapters in this book focus and elaborate on agent-based automation.

### 23.1 Theory

#### 23.1.1 Dig into the Subject

First let us briefly delve deeper into the title of this chapter Real-Time Autonomic Automation, from right to left.

**Automation** always has been—and still is—the basic starting point to relieve workers of routine tasks and increase the utilization of resources, because of production machinery, transportation capacity, and limited material stock. Automation means using any kind of mechanical device or program to carry out a repetitive job faster, more reliably, and with higher quality than human beings can normally achieve. According to Wikipedia [23.1]:

> Automation (ancient Greek: self dictated) . . . is the use of control systems such as computers to control industrial machinery and processes, reducing the need for human intervention.

**Autonomic** is already a term very close or even similar to the term agent-based. In many aspects both describe the same characteristics. Autonomic systems firstly have sensing capabilities to keep in touch constantly with their environment and know what’s going on (belief). Then they know what they want—their purpose—their goals (desire). And finally autonomic systems are able to derive and decide what to do (intention) based on the current situation and predefined goals. Putting together these terms results in belief–desire–intention (BDI), which is a software model developed for programming intelligent agents. To again cite Wikipedia:

> An autonomic system is a system that operates and serves its purpose by managing its own self without external intervention even in case of environmental changes.

The last part of this definition gives the important hint that autonomic systems in particular are able to adapt to changes in the environment (making use of their sensors) in order to decide constantly on the best action according to the current situation.

**Real time** is a critical term, as it is used ambiguously and greatly depends on the environment in which it is applied. The main differentiation made is between hard and soft real-time systems. **Hard** real-time systems guarantee a configured time from an event to system response, since otherwise the whole system will fail, e.g., in brake control for cars or tail plane fin control for a jet fighter. However, hard real-time control could also apply to comparably slow systems as long as the response is guaranteed, such as a heating control where the reaction may take several minutes. **Soft** real-time control means that a system is typically fast enough to react in due time. Deadlines are normally met, but if not kept the system will not fail, but at most lose some quality; for example, in dispatching systems real-time reaction is important, but if a decision (to reshuffle order allocations) is delayed there might be a loss of efficiency and increased costs, but the system does not fail as a whole.

Since distributed systems such as those discussed specifically in this chapter require solutions supporting heterogeneous environments, developers of such sys-
tems are attracted by the platform independency of Java. Since its invention, Java has become increasingly fast, but does not have built-in real-time capabilities. Thus the real-time specification for Java (RTSJ, JSR-001 (Java specification request)) has been developed and implemented by Sun and others as an add-on to standard Java, and offered to the market as real-time Java or the Java real-time system (Java RTS). On their website Sun gives another short and concise definition of real time, which again emphasizes the difference between speed and predictability: “Real-time in the RTSJ context means the ability to reliably and predictably respond to a real-world event. So real-time is more about timing than speed.” Additionally to a real-time programming environment a real-time operating system is needed, like Solaris 10, SUSE Linux Enterprise Real-Time 10 (SP2) or Red Hat Enterprise MRG 1.0.1 Errata releases.

Both examples described in this chapter do not fulfill and do not need to fulfill the hard real-time specification. Why? The second example is a decision support system for transport optimization where the dispatchers can still decide and become active independent of the systems, as with a navigation system. The first example has to fulfill hard real-time requirements, but the system design succeeded in keeping the critical real-time parts within the local controller of each module of the machine and thus outside the Java-based control logic. With this approach the control system where the system is normally fast enough to trigger the actuators and the reaction guarantee is given by the lowest-level controllers, the machine manufacturer could save significant costs by being able to use standard hardware, a standard operating system, and a standard runtime environment (Java SE).

**Real Time through Autonomy**

If hard real-time support will be needed in the future, this does not conflict with the system layout – quite the contrary: autonomous agent-based systems are initially designed to build real-time control systems. The major design principle to enhance real-time capabilities is the natural and elementary separation of responsibilities, and thereby the distribution of tasks. This design for scalability allows making use of all (local) processing power available in a solution and system environment. Keeping local tasks and local decisions entirely local results in high reactivity, independent of the overall system load.

### 23.1.2 Optimization: Linear Programming Versus Software Agents

To make things even more complex the real world has given us not only the challenge of real-time reactivity but also the parallel goal of optimal decisions (or at least decision support). This means that a software system not only has to cope with very complex, normally NP-hard (nondeterministic polynomial-time hard), optimization problems, but also should solve them again and again, building on the second by second changes in the environment. Unfortunately new design paradigms, such as software agents, are always compared with traditional approaches in terms of solution quality, despite the fact that traditional mathematical operation research (OR) methods are not designed to handle real-time events. For this reason the following descriptions summarize the major differences and objectives of both approaches; a more detailed discussion can be found in [23.2].

**Linear Programming**

**Pro.** Current optimizers, which means traditional OR methods such as linear programming, are designed to find the optimum for a problem, independent of the processing duration.

**Con.** However, it is very hard and time consuming to cover the full real-world complexity and map all different aspects into a linear equation system, not to mention changes of requirements, processes or business goals. Despite many traditional optimizers try to achieve real-time response, their basic intention to find the absolute optimum works against it. The optimization interval, even combined with tricks and workarounds, is normally too long to be considered real-time.

**Result.** The optimum can be found, but too late. The world might already have changed dramatically. The optimum was only valid for the past, when the optimization started. The difference between the calculated and the current optimum is large, and the accumulated error between the optimal curve and the found (to be optimal) curve increases significantly over time.

**Software Agents**

**Con.** The distributed negotiation approach with its bottom-up search space expansion is not designed to find the optimum.
Pro. Rather, it permanently strives for the optimum, as the optimization interval is very short and the approach maps the real-world complexity in all its details and can be customized to nearly any specific need without touching the core optimization principle.

Result. A close-to-optimal result is found every second or faster. The difference from the theoretical optimum is relatively small and the total error over time is kept to a minimum.

Conclusion
The optimum should not be seen and understood as a single point in time, but instead as the difference between the continuously calculated optimization curve and the real-world volatile optimum. This short excursion to real-time optimization and its application in automated systems is properly summarized in the following two statements: How does it help knowing what would have been the optimum one hour ago? Or: Better be roughly right instead of precisely wrong.

23.1.3 Classification of Agent-Based Solutions

To understand and correctly apply agent-based solutions it is important to follow a clear classification of software systems. Based on the short agent definition as a sense–decide–act loop it is straightforward to classify agent-based solutions (Fig. 23.1) depending on the existence of real-world or artificial interfaces for the sensors and actuators.

Simulation System
If you input only simulated, recorded historical data or forecasted data into an agent solution and only use the system output for analysis but not for direct decisions, then it is a simulation system. This applies, e.g., if real-time captured order data for a dispatching systems are fed into the dispatching system again, mostly for the purpose of verifying the system configuration (i.e., the cost model, see below). The result of the simulation is stored within databases, data warehouses or presented in business graphs, but not used directly to control or

Fig. 23.1 Solution classification: depending on the existence of real-world or simulated sensors and actuators you can distinguish between three system types: simulation, decision support or control system
trigger any actions. It is important to understand that the core of the system – the optimization and decision algorithm – is not simulated, but instead the sensor input and actuator behavior, and hence the real world. Very often it needs even more effort to create a realistic simulation of the world model than only implementing the sensor and actuator interfaces.

**Decision Support System**
If the input is directly linked to real-world sensors (e.g., a telematics system) and the output is only used to support and inform the dispatcher, then we talk about a decision support system. A navigation system is a good example of a decision support system, as its sensors, the GPS (global positioning system) antenna, is directly linked to the real world, the GPS satellites. On the output side, the actions resulting from the best route are not directly executed; the navigation system does not turn the steering wheel. Instead it only suggests to the driver what he/she should do. However, the driver has the final decision.

**Control System**
If the sensors are real and the output directly executes decisions without human interaction, then it is a closed control loop and thus a control system. One typical representative is the antilock brake system of a car. It automatically – fully autonomic – releases the brake if needed without having the driver in the decision loop. This autonomic behavior is based on at least two different conflicting goals: first to reduce the speed of the car and second to keep the wheels turning.

In each mode of operation the core agent solution is the same; only the real-world interfaces differ. The transportation optimization system described in this chapter is mainly used as a decision support system, but is also used to analyze history and forecasts as a simulation. The discussed machine control system is, as the name implies, a control system, where the results of the decision algorithm directly influence, e.g., the drive speed of transportation belts, heating power, and pump strength.

**23.1.4 Self-Management**
The ever-accelerating complexity and dynamics of IT systems makes their administration and optimization with only human resources no longer feasible or at times even impossible. Hence, it is reasonable and necessary to equip IT systems with capabilities that increasingly allow them to administrate, monitor, and maintain themselves. These self-management properties of so-called autonomic solutions according to [23.3] are:

- **Self-configuration**: The system automatically changes its operating parameters to adapt to mutable external conditions, some of which may even be unpredictable at the time of a system’s development.
- **Self-optimization**: The system continuously assesses its own performance, explores possible courses of actions that would result in performance improvements, and adopts the ones that are most promising.
- **Self-healing**: The system has abilities to recover from certain unfavorable conditions that may result in malfunctions. It autonomously attempts to determine compensation actions and performs them.
- **Self-protection**: The system detects threats against its functioning and takes preventive and corrective measures to ensure correct operation.

This group of properties are often also referred to as self-properties or self-* properties.

**23.2 Application Example: Modular Production Machine Control**

**23.2.1 Motivation**
Efforts to increase the flexibility of production lines are a pivotal trend in manufacturing. Whole assembly lines as well as individual machines are increasingly subdivided into modules in order to adapt precisely and just in time to constantly changing specifications (quasi make-to-order). An instance of this trend can be also found in microproduction.

However, the centralized, hardwired design of traditional control software imposes limits on dealing successfully with unpredictability. It is thus necessary to choose a novel approach to the development of control software, such that it is capable of managing
modular machines dynamically and with minimal manual intervention while automatically maximizing the throughput and thereby optimize investment into production resources.

This allows a production line to adapt continuously to changing boundary conditions and order specifications. Such a control system stands out due to its superior flexibility and adaptivity, and drives the automation and optimization of modern production lines further, while at the same time embracing the increasing complexity and dynamics of its environment.

An innovative offering in this area is a key differentiator for all vendors and users of modular production lines. Whitestein’s product living systems autonomic machine control (LS/AMC) makes use of these principles and applies them in the modular machine control market.

The particular case discussed is a concrete industrial application that entered live production in mid-2008 and is being offered as a solution to the general market.

### 23.2.2 Case Environment

The particular application case of the LS/AMC control system is a modular soldering machine wherein each module is governed by an independent local agent controller. Coordination of the individual module operational parameters and the transition of boards from one module to another are the key control aspects.

#### Machine Setup

After many years of successful soldering using a conventional monolithic machine, the project team decided to prepare for the future by initiating a redesign of the centrally controlled machine (Fig. 23.2) as a novel modular approach employing distributed control (Fig. 23.3).

The modular setup of the new design not only allows configuration of the machine according to the customer’s needs, but also has a separate local control within each module. The single drive for the one and only conveyor belt also has been replaced by one conveyor and one drive per module. This gives broad processing flexibility, as the target market for this machine typically requires changing production programs in real time.

A typical machine setup is composed of a feeder, a fluxer, one to three heaters, a soldering wave module, and a cooler module.

#### Sensors and Actuators

Each machine module has several sensors and actuators connected to the local controller board. There are digital switch sensors such as end-of-belt, zero-position, emergency-stop, and liquid level as well as linear sensors including temperature and encoder of step motors. Actuators comprise motors, pumps, and heaters as well as fans and signal lights. Overall a small standard configuration with five modules already contains around 40 sensors and 50 actuators, which have to be managed and coordinated.

#### Customer Requirements

The goal of using agent technology in this project was to minimize the complexity of development, operation, and maintenance of machines, without reducing
the degrees of freedom for future application scenarios. Specifically, this implies:

**Autonomic Equipment Adaptation.** The control software of a modern production line must autonomically adapt to the ideal equipment configuration for each order. This effectively eliminates the need for manual reconfiguration. It also ensures that future enhancements of the system remain possible with only minor outlay.

**Dynamically Varying Solder Programs.** Typically this machine is used for batch-size-one tasks, which means that each and every board is processed with different soldering parameters and the boards are processed in parallel, i.e., pipelined.

**Dynamic Performance Optimization.** The capability to optimize capacities dynamically with changing configurations and target values is a top priority. This ensures maximum throughput and minimizes idle capacities and quality failures.

**Seamless Integration Capability.** At the macrolevel it is required that the control software for modular production lines such as this offer standard interfaces to integrate into a total production control system.

**Intuitive User Interface.** Not least, such an advanced solution also needs to provide an intuitive user interface, which automatically adapts to the actual machine setup (Fig. 23.4). It offers simple controls for the machine operator, extended functionalities for specialists and technicians, and comprehensive remote maintenance capabilities via the Web.

### 23.2.3 Solution Design

**Existing Solutions for Agent-Based Control**

Whitestein Technologies has applied agent-based distributed control in many related domains throughout recent years. Before describing the path from monolithic to modular agent-based control we give three examples from other areas where distributed optimization is applied. The following examples all make use of multilateral negotiation algorithms to continuously seek optimal solutions.

**Production Scheduling.** The resources in a production environment including personnel, machines, and materials are represented by software agents that use negotiation algorithms (e.g., auctions) to offer and sell their capacity to bidding orders, which are also represented by agents. One of the prominent industry examples is described in [23.4].

**Road Logistics.** To automate the creation of dispatching plans for transportation logistics systems each resource (vehicle) is represented by an agent, which coordinates and exchanges loads with others by making use of bilateral negotiations [23.5]. (See also the next application example.)

**Supply Networks.** All the players in a supply network continuously need to coordinate their demand forecasts and capacity availability. Agents can assist in this time-consuming and time-sensitive task perfectly. Monitoring agents along the supply chain fire an alarm and trigger activities if reality deviates too much from the plan [23.6].

**From Monolithic to Modular Control**

As in all previous examples, the LSI/AMC-based soldering machine solution uses modular control principles because each module not only needs coordination with neighboring modules but also needs local, autonomic control to optimize the overall process; for example, the heater module must maintain the temperature within tolerance limits irrespective of environmental changes caused by a board running through the module or a user opening a lid. Each module must thus combine its local control tasks with overall process coordination.
Modular control also means that each module holds its own production schedule and is able to give a production forecast in a backward-chain manner to enable the feeder module to estimate when best to start a new board. The module agents combine this production planning part with the real-time control when a board physically appears and when target temperatures are reached in reality.

**Agent Model**

Besides an agent type per physically available module type, the agent model (Fig. 23.5) comprises one agent per order (printed circuit board (PCB) to be soldered) and some administrative agents for user management, configuration management, and client communication. To be precise, an agent in the agent model is an agent type, analogous to a class in object orientation. In a running application an agent is an instance of an agent type and thus correspondent to an object, as an instance of a class. The agent types used within this solution are:

- The configuration agent is responsible for detecting the attached modules of a concrete customer machine configuration via the CANopen (CAN: controller area network) bus. It then instantiates the corresponding module agents, where, depending on the detection, several agents of one type might be started, e.g., if two or more heater modules are used.
- One module agent type per physical module type, as there are:
  - the feeder module
  - the fluxer module
  - the heater module
  - the wave modules, one for the oil and one for the nitrogen version
  - the cooler module.

Each of these module agents control their module, e.g., heat up the tin, ensure the needed tin level or keep the temperature stable, and they communicate with

**Fig. 23.5** The agent model in AML (agent modeling language)
neighboring modules for the preliminary and real-time scheduling of the soldering process. New module types will be developed and added to the configuration as needed, e.g., a lift module to bring a board back to the first module of the machine.

The feeder agent has the additional task of instantiating the order agent when it detects a new board and the user presses the start button. The order agent then prepares its processing schedule by talking to each module agent, and then supervises and logs its soldering process in detail.

The client-proxy agent collects and holds all information needed to keep the connected client(s) up to date.

Besides the logging agent and the user-management agent there are more administrative agents, which are not shown in the agent model diagram for reasons of clarity.

The following are the core features of the implemented agent model.

**Autonomic Module Control.** Every machine module is represented by a specifically adapted software agent that optimizes the module’s operations and capacity utilization.

**Superordinate Coordination.** Through permanent bilateral negotiation and coordination between neighboring modules (i.e., of their software agents) the system constantly reaches a state of superordinate coordination. This eliminates the need for a central control instance.

**Self-Managing Orders.** As for every module (resource), software agents are also responsible for the control of each production unit (order). They self-manage the order’s progress through the machine(s) autonomously and ensure that all requirements relating to (cost-)efficiency, speed, and quality are optimally satisfied.

**Distributed Communication.** The decentralized approach based on bilateral communication allows for virtually unlimited scaling possibilities, while at the same time increasing robustness against malfunctions and various external influences.

**Standards Compliance.** At the controller level the software provides full support for the CANopen industry-standard machine control and communication interface.

**Interaction Model**

One of the core principles of this solution is to dynamically create one agent per module detected on the CANbus and establish a communication link to the two neighbor agents. Consequently there is no global communication among the agents but only the one on the left and the right side. This bilateral communication model is very lean but still powerful enough to drive the backward scheduling and real-time synchronization between the modules.

Here is one example of this synchronization task. As each board (production job) and each module has different processing parameters the conveyor belts typically run at different speeds. To ensure clean handover from one to the next module, LS/AMC has implemented a communication protocol (Fig. 23.6) following a notify-and-pull principle, where the sender stops and notifies the receiver and, as soon as it is ready, the receiver sets the receiving speed and then grants the sender permission to send at this speed.

23.2.4 Advantages and Benefits

The following advantages are only qualitative. Detailed metrics are not yet available and proven comparisons with other (monolithic) approaches have not been conducted, as this is an ongoing project in its final deployment phase. However, during the course of the development we experienced many of the advantages in real life, and even unexpected ones.

Especially we found the modular design to be extremely helpful in a project like this with moving targets over more than 2 years. The moving target was caused by the learning curve while designing the machine – the hardware itself. Even though sensors, actuators, and their behavior changed every week, the core of the solution has been stable and unchanged since its initial design.

We received more feedback from real life just before the publishing of this Handbook. The machine has been extended for a new customer by two lift modules, two more transportation modules, and a barcode reader. The agent-based design of the solution has shown that it can schedule and optimize the throughput and performance of the machine without any change of the algorithm. The additionally instantiated module agents naturally latched into the processing chain. They coordinated with the older module agents to control the soldering process as expected.

At least some of the following – theoretical obvious – advantages have thus been materialized.
**Flexibility**
The modular and distributed architecture of the LS/AMCs agent system allows for easy addition of new modules, without causing fundamental changes to the existing system architecture. The introduction of new kinds of machine modules only requires the development of a new module agent, which can be integrated into the current system with minimal effort.

**Autonomic Adaptivity**
New modules or modules that are failing or in need of maintenance can be exchanged while the system is running. Moreover, thanks to the LS/AMC distributed system architecture and intrinsic feedback-based adaptivity, machine control is updated autonomically at runtime without requiring restart of the control software.

**Maintainability**
Compared with traditional procedural or purely object-oriented approaches, the agent-oriented design of LS/AMC offers the advantage of intuitively mapping the real-world production line and order structure one-to-one. This makes the system better to understand and use, increases its durability, and improves its maintainability. An agent system also supports the easy and targeted customization of logging routines at the process level. This ensures the availability of more helpful and efficient methods of error monitoring and analysis.

**Simulation**
Complex simulation scenarios are easy to develop with LS/AMC, since a realistic mirror of a production line is more straightforward to simulate than an abstract model. Many different machine states and process flows can be recreated quickly and realistically. This significantly reduces the cost of quality control and improves personnel training and product demonstrations.

**Goal Orientation**
The software agents employed in this solution explicitly represent their behavior using partially conflicting logical goals. Order agents, for example, pursue the minimization of throughput time, and module agents have the goal of optimizing the modules’ resource consumption. With LS/AMC these goals do not block one another but rather dynamically coordinate toward achieving optimal overall performance.
Real-Time Autonomic Automation

23.3 Application Example: Dynamic Transportation Optimization

23.3.1 Motivation

Across Europe and worldwide, road freight transportation is a demanding high-pressure environment. Competition is fierce, margins are slender, and coordination is both distributed and often intensely complex. As a result many companies are seeking methods to control costs by enhancing their traditional dispatching methods with technology capable of intelligent, real-time freight capacity and route optimization. The former ensures that transport capacity is maximally used, while the latter ensures that trucks take the most efficient calculated route between order pickups and deliveries. These are tractable, yet complex, optimization problems because plans can effectively become obsolete the moment a truck leaves the loading dock due to unforeseen real-world events. It thus becomes mission-critical to assist human dispatchers with the computational tools to quickly replan capacity and routing.

A considerable volume of research exists concerning the domain of automatic planning and scheduling, but many real-world scheduling problems, and especially that of transportation logistics, remain difficult to solve. In particular, this domain demands schedule optimization for every vehicle in a transportation fleet where pickup and delivery of customer orders is distributed across multiple geographic locations, while satisfying time-window constraints on pickup and delivery per location.

Living systems adaptive transportation networks (LS/ATN) is a novel software agent-based resource management and decision support system designed to address this highly dynamic and complex domain in commercial settings. It makes use of agent cooperation algorithms to derive truck schedules that optimize the use of available resources, leading to significant cost savings. The solution is designed to support, rather than replace, the day-to-day activities of human dispatchers. The agent design chosen for optimization directly reflects the manner in which logistics companies actively manage the complexity of this domain. The global business is divided into regional business entities, which are usually dispatched via distributed dispatching centers. Interacting software agents represent this distribution.

While one of the largest customers of LS/ATN has demonstrated a reduction of 11.7% in costs compared with the manual dispatching solution, we typically guarantee a reduction of at least 4–6%. This improvement...
is significant for transportation companies with large numbers of orders to manage, significant costs, and small profit margins.

The achievements made thus far have been attained using only traditional manual communication (mobile phone) between the driver and dispatcher. Using this data LS/ATN generates global dispatching suggestions and improves the communication among the distributed dispatching centers. Incorporating sensor data on, for example, traffic conditions and vehicle status allows more accurate continuous estimation of vehicle estimated time of arrival (ETA), thus presenting yet further opportunities for cost savings and reduced fuel consumption. One key to this is the integration of real-time track-and-trace data feeds from en route vehicles, which act as feedback measures to an optimizer engine. This allows continuous adaptation and regeneration of dynamic route plans based on the real-world environment.

Close integration with key pervasive technologies such as GPS and reliable multinet communication offers the capability of enhancing core system intelligence with fast, timely, and accurate measures of the live environment [23,8]. Continuous transmission of vehicle state and location information provides live feedback metrics for the optimization platform, allowing human dispatchers to improve the efficiency of entire fleets. This flexibility enables logistics providers to react quickly to new customer requirements, altering transport routes at very short notice in order to accommodate unexpected events and new orders.

There can be little doubt that the future of freight transportation in Europe and beyond lies with the widespread adoption of pervasive technologies and intelligent transportation systems. One of the few questions remaining is simply how rapidly firms will adapt.

The remainder of this chapter examines the business domain characterizing the identified problems and then presents an industry-proven solution to these problems, LS/ATN (Fig. 23.7). It has been developed in close collaboration with worldwide logistics providers such as DHL, and has been proven through real-world deployment to reduce transportation costs through the optimized route solving for both small and large truck fleets. The primary aspects of our agent-based solution approach are discussed, followed by the presentation of benefits and savings, which are then continued with emerging options for incorporating state-of-the-art mobile technologies and pervasive computing into the solution.

### 23.3.2 Business Domain

Today most logistics companies use computational tools, collectively known as transport management sys-

Fig. 23.7 Details of a route in the LS/ATN dispatcher control center, as suggested by an optimizer agent
tems (TMS), such as Transportation Planner from i2 Logistics, AxsFreight from Transaxiom, Cargobase, Elit, and Transflow, to plan their transportation network from a strategic level all the way through to sub-daily route schedules. However, many TMS are unable to handle unexpected events adequately and generate plan alterations in real time. When dealing with large numbers of distributed customers, limited fleet size, last-minute changes to orders, or unexpected unavailability of vehicles due to traffic jams, breakdowns or accidents, static planning systems suffer from limited effectiveness. Significant human effort is required to manually adapt plans and control their execution.

In addition, vehicles can be of different types and capacities, are usually available at different locations, and drivers must observe regulated drive-time restrictions. To cope with all this, new intelligent approaches to route planning are emerging that are capable of continuously determining optimal routes in response to transportation requests arriving simultaneously from many customers. The key challenge lies in allocating a finite number of vehicles of varying capacity and available at different locations such that transportation time and costs are minimized, while the number of on-time pickups and deliveries, and therefore customer satisfaction, is maximized.

Road Freight Transportation
Road freight transportation is a very heterogeneous business environment serving a wide variety of customers with many different types of transportation, each configurable in many ways. In addition, large companies add the challenge of different business structures regarding processes, culture, and information technology.

One of the most significant challenges is the permanent handling of unexpected events such as traffic jams or other reasons for delays and new, changed or canceled customer orders. While new orders are an expected component of everyday business, their precise characteristics and appearance time are highly variable. A good solution must address the decentralized responsibilities of dispatchers working across the world with potentially overlapping geographical responsibilities, and supporting individual strategies and local approaches to dispatching.

To survive in an environment of significant cost pressure with margins of only 1–3%, logistics providers must address how to structure strong interaction between regional or organizational logistics networks and effectively manage the increasing complexity.

Core Challenge
The ongoing challenge for a logistics dispatcher is to find the best balance between:
- His reaction speed (time, effectiveness)
- The quality of a solution (schedule)
- The cost (efficiency) of a solution.

A comprehensive solution not only requires a core real-time optimization algorithm, but also a cooperative process bringing together all involved people.

Load Constraints
In a linear programming approach first of all you have to cover and configure the following load constraints:
- Precedence (pickup before delivery)
- Pairing (pickup and delivery by the same truck)
- Capacity limitation (dependent on truck type)
- Weight limitation (dependent on truck type)
- Order–truck compatibility (type, equipment)
- Order–order compatibility (dangerous goods)
- Last-in first-out (LIFO) loading of orders (optional).

Additionally it is important at least to take into account the following time constraints:
- Order-dependent load and unload durations
- Earliest and latest pickup
- Earliest and latest delivery
- Opening hours for pickup and delivery
- Legal drive-time restrictions
- Maximum allowed tour duration
- Lead time for ordering spot market trucks.

Problem Classification
One approach to tackle this optimization problem is by considering it as a multiple pick up and delivery problem with time windows (mPDPTW) [23.9], which concerns the computation of the optimal set of routes for a fleet of vehicles in order to satisfy a collection of transportation orders while complying with available time windows at customer locations. To solve the real-world challenge to an acceptable degree it is necessary to add another two aspects: first the capability to react in real time, and second to deal with time constraints in a flexible manner, using penalty costs to decide between a new vehicle or being late. This results in the even more complex multiple pick up and delivery problem with soft time windows in real time (R/T mPDPSTW) [23.10–12].

Thus, in addition to a pickup and delivery location, each order includes the time windows within which the order must be picked up and delivered. Vehicles are
dispatched from selected starting locations and routes are computed such that each request can be successfully transferred from origin to destination. The goal of R/T mPDPSTW is to provide feasible schedules that satisfy the time window constraints for each vehicle to deliver to a set of customers with known demands on minimum-cost vehicle routes. Another aspect is the capability to suggest charter trucks (dynamically add resources) when appropriate, i.e., when charter trucks are cheaper than the company’s own existing or fixed-contract trucks.

Further Challenges
A further significant challenge is managing opening hours, meaning to support multiple time windows during a day (e.g., lunch breaks). One of the major topics outside the optimization core problem is the ability to combine global dispatching suggestions automatically created by the system with local individual dispatcher decisions. Not forgetting the difficulty of combining continuous planning (perpetual with rolling horizon) with discrete decisions, track and trace, and billing processes. Then there is also the recurrent decision to transport direct or indirect (via a hub or depot) and to consider the limited docking or handling capacity at a hub.

Finally customer requests to parallelize the optimization of the three main resources, truck/tractor, trailer/swap body, and driver(s), must also be handled. Each may take a different route due to the pulling unit (truck/tractor), with drivers also potentially changing during a tour.

23.3.3 Solution Concept

The centralized, batch-oriented nature of traditional IT systems imposes intrinsic limits on dealing successfully with unpredictability and dynamic change. Multiagent systems are not restricted in this way because collaborating agents quickly adapt to changing circumstances and operational constraints. For real-time route optimization, it is simply not feasible to rerun a batch optimizer to adjust a transport plan every time a new event is received. Reality has shown that events such as order changes occur, on average, 1.3 times per order. Distributed, collaborating software processes, i.e., agents, can however work together by partitioning the optimization problem and following the bottom-up approach, thereby solving the optimization in near-real time.

Software Agents
To solve the domain challenges described above it is necessary and advantageous to apply a new software design concept: software agents. This technology offers an ideal approach to allow real-time system response and assessment in a distributed heterogeneous environment. Software agents are grounded by the notion of communication between independent active objects, each of which may have its own goal objectives and role assignments. These capabilities inherently mirror typical business structures and processes. Technically, software agents operate using sense–decide–act loops, which can be either purely reactive or proactively goal oriented.

In the transportation business domain an agent could be a packet, a pallet, a truck, a driver, an order or a dispatcher. They follow a reverse, bottom-up optimization principle with decentralized solution discovery and escalation strategies: first a dispatcher mentally optimizes within his domain of responsibility (e.g., 20 trucks), then in steps expands the search space to his office, his subsidiary, the region, the country, and finally tries to improve a solution globally.

Bilateral Order Trade
As mentioned, the agent design principle is based on communication and interaction among autonomous objects mirroring the real world. This optimization model closely follows cooperation in reality, where all trucks are driven and managed by self-employed drivers (and truck owners). They first accept each new order they get from any customer and then start to search, and negotiate with, other truck drivers in order to exchange or transfer orders looking for a win–win situation for both sides. This is triggered by each order event, where an order exchange also counts as an event. Each truck negotiates with other trucks in sequence with a tight restriction to bilateral order trades. However multiple trades can take place in parallel, always between a pair of trucks. This solution design allows fully distributed and parallel solution discovery, which scales very well and allows individual goals and strategies per truck (agent).

Agent Model and Strategy
To solve the R/T mPDPSTW problem dynamically, the LS/ATN transportation optimizer [23.13], used by DHL throughout Europe, segments and distributes the problem across a population of goal-directed software agents. Each agent represents a dispatcher, who manages one or more vehicles (resources). This is slightly different to exactly one agent per one truck, but the
principle is the same, even closer to reality where a dispatcher manages more than one truck. The reason was technical performance optimization while keeping the core principle of bilateral negotiations.

The system is completely event driven: a new order, a changed order, a delay or a successful order exchange triggers a local activity. The dispatcher of the affected vehicle becomes active and tries to optimize by negotiating with neighbor trucks, trying to exchange or move loads and by checking and calculating all reasonable combinations and selecting the cheapest. The global optimum is striven for through a kind of snowball effect, which stops when there is no more optimization found. A threshold savings value, which avoids an order exchange for too little saving, reduces plan perturbation.

To find the optimal allocation the agents work on a strict cost basis. Each possible route is checked against a configurable, individual, and fully detailed cost model. This market-based approach, the money to be spent, is the common denominator to make the multiple conflicting goals comparable, which are:

- Reduction of empty driven distance
- Reduction of waiting times
- Increase of capacity utilization.

For R/T mPDPSTW optimization, an agent represents each geographical region, or business unit, with freight movement modeled as information flow between the agents (Fig. 23.8). Incoming transportation requests are distributed by an AgentRegionBroker (not shown) to the AgentRegionManager governing the region containing the pickup location. The number of such agents depends on the customer’s setup of (regional) business units and varies between 6 and 60 for current deployments. In the larger case, 10,000 vehicles and up to 40,000 order requests are processed daily. This implies that no more than a few seconds are available to reoptimize a transportation plan when, for example, a new order must be integrated. Each AgentRegionManager generates a transportation plan specifying which orders to combine into which routes and which vehicles should be assigned to those routes. Agents exchange information using a negotiation protocol to insert transportation requests sequentially, while continually verifying vehicle availability, capacity, and costs.

While the optimization function is 100% cost based, other objectives must be satisfied in parallel when calculating routes. Some of these constraints are compulsory (hard), such as capacity and weight limitations of the vehicle, customer opening hours, that pickup date is before delivery date, and that pickup and delivery are performed by the same vehicle. Other soft constraints can be violated with a cost penalty, such as missing the latest possible pickup time or delivery time.

Experiments and First Findings

In the course of the software development we evaluated the effect of certain key parameters. One of these is the number of orders being negotiated and transferred between trucks ($k$). Our experiments showed that the runtime increases linearly with increasing $k$, but the

---

**Fig. 23.8** Illustration of freight transportation in Europe partitioned into six regions, each with its own agent region manager. Blue circles represent major transport hubs and red lines indicate example routes connecting hubs.

**Fig. 23.9** Optimization results with increasing $k$ (number of orders exchanged)
costs decrease only marginally (Fig. 23.9). $k = 0$ means that there is no order exchange taking place, but only the first-time allocation.

Another experiment is the effect of the maximum allowed delay for soft time windows. The graph (Fig. 23.10) shows that a maximum delay above 2 h decreases the quality (number of broken constraints increase) while not reducing costs significantly.

The Decision Support Process of LS/ATN

To integrate the globally generated optimization recommendations with the distributed dispatchers performing manual optimization we identify the following decision support process (Fig. 23.11): Orders arriving from an external system (ERP (enterprise resource planning), TMS (transportation management system) or other) flow into the core agent system, which generates dispatching recommendations in the form of a globally optimal matching proposal of orders to trucks. There is no time limit into the future; LS/ATN has an endless planning horizon. A certain lead time before the orders are to be checked and released, the system transfers them to the to-do board of the responsible dispatchers. They approve and adjust the suggested plan if needed prior to fixing the tour and purchasing the required transportation capacity. This automatically sends a confirmation to the subcontracted carrier and can issue a message to the truck driver, if equipped accordingly. The released routes then switch to tracking mode where the agents take over responsibility for monitoring incoming tracking messages, verifying whether they indicate an existing or upcoming time violation. The dispatcher is informed only if needed. As a final step the dispatcher releases a finished tour for billing.

This ideal flow covers the standard case, but reality often intercedes to force alterations in real time. In any situation a dispatcher can put a tour or an order into his manual dispatcher board and adjust the plan. This might be needed if, for example, an actual order differs from the booking only when loading it at the customer site.

### 23.3.4 Benefits and Savings

**Higher Service Level at Reduced Cost**

LS/ATN agent-based optimization guarantees a higher service level in terms of results quality. The high solution quality corresponds to a reduced number of
violated constraints. The system allows the desired level of service quality to be fine-tuned. Figure 23.12 presents results obtained from LS/ATN relative to the manual dispatching solution of very experienced dispatchers (manual).

The first proposed solution (LS/ATN 1), using relaxed soft constraints as the manual dispatchers do, provides a reduction of 8.3% in driven kilometers at the same service level with no more than 25% violated constraints.

The second solution (LS/ATN 2), configured with higher penalty costs for late delivery, shows a reduction in driven kilometers of 0.8% relative to the manual dispatching solutions, while providing a significant higher service level: only 2.5% violated constraints with more than 6 h delay.

The third solution (LS/ATN 3), not allowing any time window violation, shows an increase of only 1.7% in terms of driven kilometers, while meeting all the constraints to 100%.

In this analysis we compared the system results with real-world (manual) results, as the customers regard these metrics as optimal. Furthermore, there is the cost and resource problem if one would like to compare the results with the real optimum. This would require setup and development of a parallel solution based on linear programming, which is – according to our experience – not capable of covering all detailed requirements and constraints. Customers do not pay for such a comparison, as it is far too expensive and, even if one could develop a parallel solution fulfilling all requirements in the same way, there is no guarantee that the (one-time) optimal solution will end in due time.

**Significantly Increased Process Efficiency**

Through the use of automatic optimization a lower process cost is achieved. This is due to automatic handling of plan deviations and evaluation of solution options in real time. Moreover, through automation, the communication costs in terms of dispatcher’s time and material is reduced. Better customer support can be guaranteed through fast, comprehensive, and up-to-date information about order execution. Automation also allows processing of a higher number of orders than with manual dispatching only. This is an important issue as the volume of data to be managed is constantly increasing.

![Fig. 23.12](image) Improvements obtained with LS/ATN over manual dispatching. Higher service level at reduced cost. Saved driven kilometers are compared with the manual figure.

![Fig. 23.13](image) Significant cost savings through optimized capacity utilization.
Significant Savings through Optimized Capacity Utilization

Cost savings cannot only be achieved by avoiding empty trips and reducing driven kilometers; an important aspect of cost reduction is the optimal use and allocation of the company’s own and chartered trucks to a mixture of one-way, back-, and round-trips. Without the ability to reduce the driven kilometers significantly there is still a saving potential of up to 7%, as shown in Fig. 23.13.

Savings Potential in Numbers

A partial dataset from our major customer, DHL Freight, contains around 3500 real business transportation requests. In terms of the optimization results, obtained by comparing the solution of manual dispatching of these requests against processing the same orders with LS/ATN, a total 11.7% cost saving was achieved, where 4.2% of the cost savings stem from an equal reduction in driven kilometers. An additional achievement is that the number of vehicles used is 25.5% lower compared with the manual solution. The cost savings would be even higher if fixed costs for the vehicles were included, which is not the case in the charter business, but possibly in other transportation settings.

Combined with other real-world comparisons we can estimate an overall transportation cost saving of 5–10%, which are variable costs (subcontractor payment) and thus have an immediate effect. Fixed cost savings of 50–100% resulting from process and communication improvements are long-term effects, which only pay back when the resources are reallocated.

23.3.5 Emerging Trends: Pervasive Technologies

Although capacity and route optimization tools are proven to produce significant reductions in operating costs, many in the transportation industry are acutely aware that one key and often missing component of the optimization strategy is the provision of real-time feedback from en route vehicles. The objective is an intelligent transportation management system with every vehicle providing up-to-date information of progress through a pickup/delivery schedule and with onboard sensors detecting, for example, when freight is loaded and unloaded, and whether its condition (e.g., temperature) is within tolerance limits.

The intelligent transportation management systems model [23.14] developed within the transportation industry is grounded in the principle of vehicle tracking and incorporation of real-time information into the transportation management process using available pervasive technologies. The emerging approaches to realizing this model involve various combinations of pervasive technologies, some of which are all highlighted in the following section. This section highlights some of the most relevant technologies in use today, or in the early phases of adoption. LS/ATN is able to make use of data sourced from, manipulated by, or transmitted by any of these technologies to enhance the route optimization process.

Global Positioning System (GPS)

Automatic vehicle location (positional awareness) uses GPS signals for real-time persistent location monitoring of vehicles. Both human dispatchers and route planners such as LS/ATN can then track vehicles continuously as they move between pickup and delivery locations. Active GPS systems allow automatic location identification of a mobile vehicle; at selected time intervals the mobile unit sends out its latitude and longitude, as well its speed and other technical information. Passive GPS uses the onboard units (OBU) to log location and other GPS information for later upload. Accuracy can vary, typically between 2 and 20 m, according to the availability of enhancement technologies such as the wide-area augmentation system (WAAS), available in the USA. The European Galileo system will augment GPS to provide open-use accuracies in the region of 4–8 m within the European region.

The adoption of GPS is growing quickly as the technology becomes commoditized, but some transportation companies remain reliant on legacy equipment for measuring vehicle location. Some of the alternatives to GPS in use today include dead-reckoning, which uses a magnetic compass and wheel odometers to track distance and direction from a known starting point, and the long-range navigational (LORANC) system, which determines a vehicle’s location using in-vehicle receivers and processors that measure the angles of synchronized radio pulses transmitted from at least two towers with predetermined position. Another system in use by some transportation companies is cellphone signal triangulation, which estimates vehicle location by movement between coverage cells. This only offers accuracy typically in the region of 50–350 m, but is a cheap and readily available means of determining location.

Onboard Units (OBU)

An OBU, otherwise known as a black box, is a vehicle-mounted module with a processor and local memory...
that is capable of integrating other onboard technologies such as load-status sensors, digital tachographs, toll collection units, onboard and fleet management systems, and remote communications facilities. The majority of OBUs in use today, such as the VDO FM Onboard series from Siemens, the CarrierWeb logistics platform, and EFAS from Delphi Grundig, are typically used to record vehicle location, calculate toll charges, and store vehicle-specific information such as identity, class, weight, and configuration. Some emerging OBUs will have increased processing capabilities allowing them to correlate and preprocess collected data locally prior to transmission. This offers the possibility of more computational intelligence installed within the vehicle, enabling in situ diagnostics and dynamic coordination with the remote planning optimizer such that the vehicle becomes an active participant in the planning process, rather than simply a passive provider and recipient of data.

Vehicle data, in its most common form, relates to the state of the vehicle itself, including, for example, tire pressure, engine condition, and emissions data. Automatic acquisition of this data by onboard sensors and its transmission to a remote system has been available within the automotive industry from some years and is now gaining substantial interest in the freight transportation business. The OBU gathers information from sensors with embedded processors capable of detecting unusual or deviant conditions, and informs a central control center if a problem is detected. Sensors also measure the status of a shipment while en route, such as detecting whether the internal temperature of refrigerated containers is within acceptable tolerance limits or whether a door is open or closed.

RFID

Many assets, including freight containers, swap-bodies, and transport vehicles, are now being fitted with transponders not only to identify themselves, but also to detect shipment contents and maintain real-time inventories. In the latter case, units are equipped with radiofrequency identification (RFID) readers tuned to detect RFID tags within the confined range of the container. Some tags, such as the Intermec Intelligat with an operating range of 4 m, are specifically designed for pallet and container tracking, where tags are attached to every item and automatically scanned whenever cargo is loaded or unloaded. The live inventory serves as both local information for the driver and as real-time feedback to the TMS, which uses it for record keeping and as input to the real-time route planner.

In addition, e-Seals, whether electronic or mechanical, are now often placed on shipments or structures to detect unauthorized entry and send remote alerts via the OBU. E-Seals on a container door can also store information about the container, the declaration of its contents, and its intended route through the system. They document when the seal was opened and, in combination with digital certificates and signatures, identify whether the people accessing the container are authorized to do so.

Mobile Communications

Electronic communication is the key enabler of pervasive technologies. In transportation the most basic form in use is the short message service (SMS), which is commonly used to communicate job status such as whether a driver has delivered an order. Technology is already in place to automatically process SMSs and input the data into the route planner.

Also now in relatively widespread use is dedicated short-range communications (DSRC) operating in the short-range 5.8–5.9 GHz microwave band for use between vehicles and roadside transponders. Its primary use in Europe and Japan is for electronic toll collection. DSRC is also used for applications such as verifying whether a passing vehicle has a correctly operating OBU.

Currently, the technology with the greatest utility is machine-to-machine (M2M) [23.15] communication, which is the collective term for enabling direct connectivity between machines (e.g., a vehicle’s OBU and the remote planning engine) using widespread wireless technologies. Legacy second-generation (2G) infrastructure is most commonly used as third-generation (3G) technologies enter the mainstream for day-to-day human telecommunications. M2M is quickly emerging as a principle enabler of networked embedded intelligence, the cornerstone of pervasive computing. It can eliminate the barriers of distance, time, and location, and as prices for the use of 2G continue to drop due to continued rollout of 3G technologies, many transportation companies are taking advantage and adopting M2M as their primary means of electronic communication.

Emerging solutions take M2M to another level by enabling always-on and highly reliable communication through automatic selection of connection technology, e.g., general packet radio service (GPRS), enhanced data rates for GSM evolution (EDGE), universal mobile telecommunications system (UMTS), satellite services, and WiFi according to availability. The LS/ATN route optimizer, for example, can be augmented with a re-
mote connection agent module [23.16] installed in vehicles that offers seamless M2M over cellular technologies, wireless local-area network (LAN), and even short-range ad hoc connections if available. The selection of a particular communication technology can be made either manually or automatically, depending on several metrics including location, connection availability, transmission cost, and service type or task; for example, a fleet operator may prefer the use of satellite to communicate directly with a driver, but then a combination of cellular technologies for remote monitoring, trailer tracking, and diagnostics. Low-cost GPRS might be selected to download position coordinates from an onboard GPS; whereas, a higher-bandwidth (and cost) option such as UMTS/WCDMA (wideband code division multiple access) might be preferred for an over-the-air update to the OBU or onboard sensors.

The position information of a single vehicle is used to adjust dispatching plans immediately in the case of deviations (described below in more detail). If the number and density of vehicles in a region is high enough, this floating vehicle data may be integrated into a map containing real-time traffic flow information [23.17].

Existing: state-of-the-art real-time dispatching combined with traditional track & trace and communication

Future: M2M real-time track & trace and re-scheduling combined with instant driver update

Opportunities of Using Pervasive Technologies
Transportation route optimizers can take advantage of real-time data sourced from vehicles equipped with pervasive technologies by incorporating information relating to vehicle location, state, and activity into their planning processes.

Figure 23.14 shows the major difference and step from the current deployment of the agent-based real-time dispatching system, only making use of traditional communication and track-and-trace capabilities with many manual human activities involved, toward a full real-time control loop leaving the dispatcher in a purely supervisor role. In the future, the sensor and actuator interfaces will be increasingly automated while the decision core system is already in place.

Pervasive communication provides a permanent bilateral link between the vehicles and the dispatching system. Onboard preprocessing is available to calculate continuously the estimated time of arrival (ETA) at the next node, which is periodically sent to the server in order to check immediately the impact on the dispatching plan. Taking speed and other local knowledge into account the local preprocessor is able to deduce traffic conditions and forward this (only temporarily valuable)
knowledge to other vehicles via the dispatching system. The combination of sensed speed with the current location can trigger an automatic status message if the truck is waiting for loading/unloading at a customer site or is idle in a traffic jam. A similar functionality is so-called geo-fencing, which issues a status message when entering or leaving a destination. A further automation is the local communication between truck and a smart container for docking and undocking messages.

All of the above simplify and speed up execution of intelligent information services integrated with route optimization and the derivation of schedules can directly use both information relating to vehicles movements as they proceed through delivery schedules and feedback from RFID transponders notifying when orders have been added to or removed. This real-time component implies that time windows can be more finely tuned according to current events, resulting in alternative schedules that can either compensate for delays or take advantage of time saved. Preliminary results with a prototype demonstrate that employing real-time data in the optimization process can further reduce transportation operating costs by up to 3% beyond the 5–10% achieved from the standard optimization process described earlier, depending on the particular case and system configuration.

### 23.3.6 Future Developments and Open Issues

There remain many scientific and practical challenges related to the design and use of real-time dispatching system. A selection of these that we consider relevant to LS/ATN and for consideration by the community at large are described below.

A major challenge is the effective handling of intercompany, interregion, and intermodal transportation. Transportation intrinsically involves multiple carriers operating both within and across sectors (i.e., road, air, and shipping) and across geographical boundaries. Each carrier has its own, often proprietary, systems that do not necessarily integrate easily with one another. Addressing this integration problem is a significant engineering issue to be faced as the technologies addressed in this chapter come into more widespread use. The integration of transportation planners into supply chain and production systems is also important. As previously mentioned, freight is now often delivered directly to manufacturing plants without passing through transitional storage. Integration of these systems thus becomes a priority when shaping dynamic supply chains, and supply networks.

**OBUs** in use today typically consist of a simple processor, memory, and communication interfaces. Installed software is often designed solely for reading data from sensors and transmitting it to the TMS. One method of improving on this design is the integration of autonomous software controller into the OBU to assist with the manipulation and coordination of collected onboard data. Example uses include assisting in the selection of M2M connection type in a multi-provider environment according to the type and volume of data to be transmitted and caching data locally if connections are temporarily unavailable. The controller can be further extended with a software agent that extends the distributed intelligence offered by the route optimizer. This agent essentially acts as a remote extension of the optimization platform, allowing the agent to act as a proxy representative of the vehicle itself within the context of route scheduling. Vehicles can thus become active participants in the planning process, forming a network overlay of communicating data processors.

Further research is required on so-called smart freight containers capable of announcing their presence and even negotiating with external devices; for example, a simple OBU fixed to a container will allow it to communicate with vehicles, customs checks, and equipment at freight consolidation centers. Many major transportation companies use such centers, distributed at strategic locations, with the primary goal of consolidating freight onto as few vehicles as possible to maximize use of available capacity. With the installation of RFID readers, incoming freight with RFID tags can be traced as it moves through a facility, providing TMS optimizers with complete coverage of freight location throughout its entire lifecycle within the business chain.

In addition, external factors also favor early adoption of pervasive technologies, such as the ongoing escalation of fuel prices, new regulations for pollution reduction, and constant increases in demand for fast, high-volume freight shipping. This is recognized in the European Union white paper *European Transport Policy for 2010* [23.18] which discusses the use of intelligent information services integrated with route
planning systems and mobile communications to provide real-time, intelligent end-to-end freight and vehicle tracking and tracing.

There can be little doubt that the adoption of intelligent transportation planners capable of using real-time data sourced from pervasive technologies such as those discussed in this chapter is a major objective of many freight transportation operators both in Europe and other areas of the world. With these techniques now widely recognized as an important means of reducing operating costs, many companies are already well advanced on the path to adoption.

23.4 How to Design Agent-Oriented Solutions for Autonomic Automation

In the past decade a lot of work has been done on agent-oriented analysis and design. The works presented in [23.19] and [23.20] are only two examples, but very good starting points to dig into the agent world. More details about agent organization, agent platforms, tools and development can be found in [23.7, 21–23].

Other chapters in this Handbook also cover agent-based solutions, the following gives just a brief outline on how to start thinking in an agent-oriented way in the form of a questionnaire. A detailed discussion would be far beyond the scope of this Handbook.

Questions to structure the overall solution:

- What are the processes?
- Who/what drives the processes?
- Which roles do the process drivers play?

Questions to ask for each agent/role:

- What is the responsibility of the agent?
- Which goals does the agent aim at?
- What is the strategy to reach the goals?
- What knowledge does the agent need to follow this strategy?
- With which agents does he need to communicate?
- Which sensors and actuators are needed or available?

These questions have been discussed and answered to design and implement the two application examples contained in this chapter.

Two main aspects of an agent-based solution should be considered in order to analyze and prove the quality of the design. These aspects cannot be given as a concrete metric, but should be understood as general indicators relative to the size and type of the intended system:

- **Local knowledge**: A good agent solution has been achieved (or is possible) if the local knowledge needed by an agent to achieve its goals can be kept to a minimum. If a solution requires an agent to hold a large amount of data, or – in an extreme case – each agent needs to know *everything*, either the design should be rethought or an agent-based solution is not appropriate. Consider, for example, a sales representative (agent) for a car manufacturer, whose goal is to sell as many cars at the highest price possible. For that he does not need to know all the details about car production or supply chain organization. He only needs some extract of the whole business knowledge.

- **Communication**: Although message exchange and service-oriented architectures can accompany agent-based ideas, a good solution keeps communication to a minimum and makes careful use of resource *bandwidth*. If a design requires too much messaging among the participants then the role, and therefore the goal assignment, is not distinct enough. Agent-oriented design means to define a good level of responsibility and assign it to a software entity, which allows it to pursue its goals and to decide actions based on local knowledge. Cooperation with the environment is needed to sense what’s going on but should not be needed to draw conclusions.

23.5 Emerging Trends and Challenges

23.5.1 Virtual Production and the Digital Factory

The automation industry, which includes at least all machine manufacturers, has the vision that all components of a production facility will be accompanied by a full digital description in a standardized format. Besides easy and straightforward integration into factory simulation tools, the goal is also to let modules carry their own electronic description to enable them to plug
in and integrate automatically into a production system in the sense of self-configuration. Testing and putting into operation would become as easy as attaching a new mouse to a computer. Even though it is obvious that this will not work for all machines and that it is very hard to achieve, it is a very worthwhile goal to work towards. If the modules additionally come along with their own agents, they can also dynamically negotiate with the environment in which they are placed and (self-)optimize their activities.

**23.5.2 Modularization**

There is a clear trend and motivation in the industry to modularize machines and production facilities, yielding many advantages.

The customer (user) of a machine gets much greater flexibility, as he can order, configure, and dynamically adapt his production lines according to market needs. A common keyword in this regard is the selling argument grow as you need. On the maintenance side more cost reductions are possible as fewer spare parts are needed for modules built on the same framework, and if a defective module needs to be replaced this is normally easier than replacing a whole machine. Some production machines even offer a shopping-cart-like system, where a component can be exchanged without a screwdriver.

The manufacturer has smaller components to produce, which needs less space, at least for each production cell. In the same way quality tests become easier and faster because only a single module has to be tested. Last but not least, smaller modules are easier and cheaper to transport and deliver. The increased number of common parts leads to cheaper production because fewer tools are needed, and less space for many different parts and higher purchasing discounts can be achieved.

Overall, modularization is a win–win concept for all parties.

**23.5.3 More RFID, More Sensors, Data Flooding**

As RFID technology increasingly finds its way into industrial usage and other sensors based on video cameras, induction loops or microwaves we increase the amount of generated data day by day. Many companies are hungry for data, but have not clearly defined what to do with this new data flood. Admittedly the data and its accuracy have a high value, but one has to be aware that all this new data keeps its high value for only a very limited time. In other words: you have to process and gain the value from fresh data immediately. Because of the huge volumes involved, this can only be done by automated processes, which handle sensor input where it appears and drive activities without too much data transfer through the network. One can therefore conclude that RFID and other sensors increase the need for software agent concepts.

**23.5.4 Pervasive Technologies**

**Limitations: Onboard Agents**

As discussed in Sect. 23.5.1, one vision is to equip machine components and modules directly with their self-* logic (Sect. 23.1.4) and software representative. However, since the processing power of most controller boards is still not sufficient, and more importantly since such a huge variety of controller boards exist, it is not the first goal to support all of these directly. Instead it is much more convenient, faster, and not to forget cheaper to let the agent logic run on dedicated computers and just implement interfaces to the different controllers. This approach is not at odds with the general distributed solution design. It is just a special deployment decision. The very specific controller boards are not loaded with additional computing tasks, but only used as interfaces to the attached sensors and actuators. The software agents are deployed to one or more standard computers installed in the field as needed. A solution architect could theoretically use one dedicated personal computer (PC) per agent (per controller), which would directly reflect the distributiveness of the solution, but – again for cost reasons – several controllers, located in the same module, machine, area, room or building, can easily host the agents of many attached controllers. The agents still somehow work locally, close to the physical installation and thus the overall solutions provides redundancy, reduced latency, and near-real-time responsiveness. Step by step, as controller boards become more powerful in the coming years, the agents can be run directly on the board. This will be a smooth transition without changing the solution’s core algorithms and allows one to take advantage of autonomous concepts already today.
References


