Decision Support Systems and the Coordination of Supply Consortium Partners[☆]

Jörn Schönberger and Herbert Kopfer

University of Bremen, Chair of Logistics, Wilhelm-Herbst-Straße 5, 28359 Bremen, Germany, {jsb, kopfer}@uni-bremen.de

Abstract

We propose to enhance decision support systems (DSS) by coordination capabilities to align the decision making of consortium partners for improving the process quality in volatile process environments. In computer simulation experiments, we reveal conceptional shortcomings of traditional DSS if unforeseen process-threatening events like unexpected workloads have to be handled by a consortium. It is demonstrated for a transportation process planning scenario that a process quality increase is achieved if a DSS is extended by components that align the process-related decision making of legally independent supply consortium partners. The central idea proposed to add coordination capabilities to DSS is to temporarily adjust decision models of subordinate consortium partners to the fulfillment degree of consortium objectives in order to establish a coordinated decision making.

1. Introduction

A Decision Support System (DSS) is an information processing system especially dedicated to derive or support the derivation of goal-oriented decisions in complex decision situations. Such a system combines process-related data with analytical decision models in order to enable a computer-based control of value creation processes [1]. DSS are set up according to some commonly agreed design paradigms.

Within this contribution we report a research about the evaluation of existing DSS-concepts for the management of transport processes in a volatile process environment. In such an environment, the planning conditions and requirements vary significantly within a small time span. Processes running in a volatile environment need a frequent update in order to maintain their efficiency.

[☆]This research was supported by the German Research Foundation (DFG) as part of the Collaborative Research Center 637 "Autonomous Cooperating Logistic Processes" (Subproject B7)

The management of processes in a volatile surrounding becomes even more challenging if two or more decision makers are involved (often on different decision levels). Here, the needs and objectives of all participating acteurs have to be considered simultaneously during the process planning. A supply chain consortium is an often found example for such a complicated decision making scenario. The superior coordinator instructs a subordinate service partner to fulfill certain orders. In reaction to this call, the service partner deploys its resources to fulfill the orders by executing processes. In this report, we deal with such a two-decision-maker-situation arising from a supply chain scenario, in which the subordinate service partner offers transportation.

By means of the two-decision-maker situation from supply chain planning, we reveal shortcomings of DSS design paradigms caused by extraordinary events whose processing is not defined in advance. Although previous research [2, 3] has addressed new ideas for handling those events DSS architectures are unable to handle these events efficiently. The main contribution of this article is the proposal of an extension of the three-layer event-handling-concept [4] by a fourth layer that controls even the handling of extraordinary events. We define and evaluate a four layer event handling system for the management of extraordinary situations in the above-mentioned two-decision-maker-situation.

We state the following research hypotheses guiding the research reported here: (i) Two (or more) concurrent process decisions makers cannot be sufficiently supported by a DSS set up according to existing design paradigms (ii) A consideration of several decision makers in the layout of a DSS contributes to overcome some deficiencies arising from conflicting or even contradicting planning goals of interacting process decision makers.

The organization of this paper is as follows. We start with the description of the investigated process decision situation (Section 2). Then, we compile DSS-design principles (Section 3), propose a multi-agent-system-based DSS for the aforementioned decision situation (Section 4), and demonstrate its shortcomings. A conceptual extension of design guidelines for DSS is proposed in Section 5 and applied to extend the DSS from Section 4. Performing computational simulation experiments, we evaluate the extended system (Section 6).

2. A Dynamic Transport Process Planning Problem

We start with the description of the derivation of transport processes from customer demand in a supply consortium (Subsection 2.1). Then we introduce a specific transport process challenge (Subsection 2.2) and discuss the need for a coordinated decision making in a supply consortium (Subsection 2.3). An online optimization model for the investigated scenario is proposed (Subsection 2.4).

2.1. Transport Process Planning in a Supply Chain Scenario

A supply chain consortium is a collaboration of independent companies that set up, maintain and operate a value creation chain contributing their specific

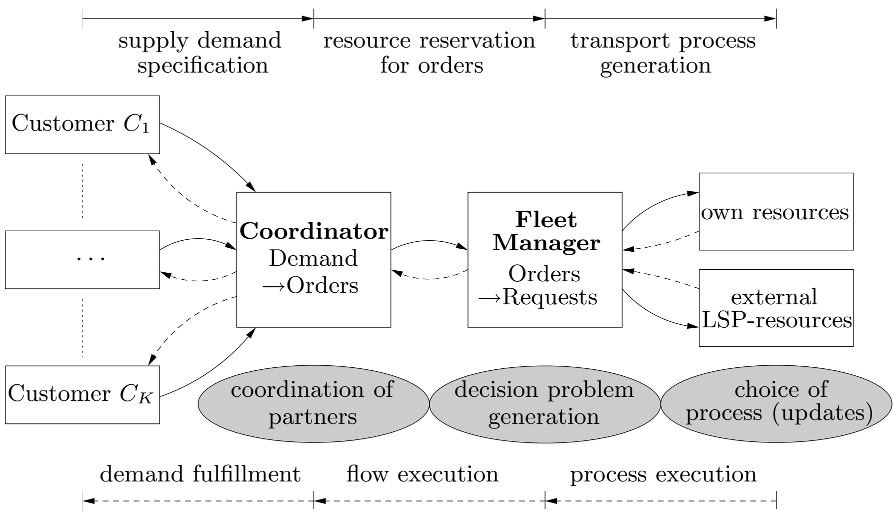

Figure 1: Transport Process Planning Scheme

knowledge and resources to the value creation processes. Due to the spatial scatter of the value creating locations (plants, storages, customers) excessive transport of raw materials, semi-finished or finished goods has to be carried out. A freight forwarding company is the partner within the consortium that is responsible for configuring the necessary transport processes and deploying the accessible transport resources (trucks or external and booked capacities).

The derivation of profitable and reliable transport processes from customer demand is figuratively presented in Figure 1. It is a realization of the hierarchical planning idea [5, 6]. Initially, customers of the consortium specify their supply demand towards the coordinator. The coordinator derives internal orders and reserves capacities at the consortium partners. Especially, the fleet manager receives transportation orders. This order specification implies a decision problem because the fleet manager has to derive transportation requests and to decide about the setup (or update) of transportation processes for fulfilling the requests. A fleet manager decides about the deployment of own trucks or external resources provided by logistic service providers (LSPs). The execution of the transportation processes leads to the fulfillment of the orders given to the fleet manager so that the necessary flow of goods between the value creation locations is executed. This finally contributes to the customer demand fulfillment.

2.2. A Transport Process Planning Challenge in a Supply Chain

We assume that the supply consortium coordinator has agreed a contract with a customer. This customer specifies demand and submits this demand unregularly and at unpredictable times to the coordinator. Immediately after the reception of the demand, the coordinator specifies the consortium orders and injects the generated orders into the order pools of the involved service agents.

The coordinator receives customer demands continuously over time. He generates orders from the customer demands and the resulting requests are to be executed by the transport providing partner. A reception of additional requests triggers a process revision to incorporate the additional requests into the so far followed transport processes. The process-planning problem of the fleet managing agent is therefore a dynamic decision problem [7], which is solved in online fashion, e.g. a process revision is carried out in an event-driven fashion in response to the additionally submitted requests. Consequently, a sequence of concatenated decision problem instances P_i is stated. Each instance is formulated as a static and deterministic mathematical optimization model (all relevant data at the re-planning time t_i are assumed to be known). Solving such a model means to find the most profitable process decisions for the transport operations considering the so far actually known planning data.

A release of one or more additional requests initiates the revision of the so far constructed routes for the own vehicles. If needed some requests are excluded from the routes of the own vehicles and forwarded to an LSP. A re-assignment of requests formerly given away to an LSP to an own vehicle is impossible.

The fleet managing agent receives a certain amount as budget from the coordinator for covering all expenses associated with the transport order fulfillment. From this budget he has to pay his costs for fulfilling the necessary transport orders (travel expenditures and fees to be paid for subcontracted shipments). The difference between the overall budget and the request fulfillment costs remains as surplus at the fleet managing agent.

A service degree is fixed in the contract agreed between the coordinator and the fleet manager, e.g. a given percentage p^{target} of the customers' transport demand has to be fulfilled within the customer-specified time restrictions. At a certain time t, there are f_t requests whose completion times (already realized or scheduled) fall into the period $[t-500, t+500]$ (moving time window). Among these f_t requests the number of f_t^{punc} requests is completed (or expected to be completed) within the previously agreed time windows. The current process punctuality rate p_t is defined by $p_t := \frac{f_t^{punc}}{f_t}$ (representing the current reliability of the transport system). When the quotient p_t does not fall below the threshold value p^{target} the reliability requirement is met. This quote can be explained as follows: Each demand comprises goods necessary to keep the production processes at the customers' factories running and goods used to build up security stocks. The first kind of goods must be provided in time while the second kind of goods can be delivered later without causing corruption on the running production processes. In a *high quality (HQ) period* the requirement for the least punctuality is fulfilled ($p_t \geq p^{target}$) but in a *low quality (LQ) period* the required punctuality rate is not attained ($p_t < p^{target}$).

For the reason of simplicity, we assume that each transport order is an executable task and it is converted 1:1 into a transportation request and added to the request pool of the fleet managing agent. A request instructs a transport resource to visit a given location during a customer specified time window. Examples in which such a kind of request occurs are related to situations in which a large number of small-sized packages are loaded at the beginning of a day-trip so that packages (like spare parts) can be delivered to a large number of customers without the necessity to re-visit a loading berth. Other applications are the collection of used consumable items (collection of used laser or ink-cartridges during office-hours) and the dispatching of service crews or repairmen [8]. For a summary of further related scenarios we refer to [9]. If the number of additionally released requests temporarily increases so high that it is not possible to serve the additional requests immediately, then a workload peak occurs.

Whenever an additional request corrupts the execution of the so far followed processes a process revision (re-planning) becomes necessary. The arrival times at some customer sites may be postponed to have the possibility to serve one or several additional customers earlier by the same vehicle. Typically, the processes of several transport resources have to be updated simultaneously because a re-assignment of requests is necessary if the originally selected vehicle is not able anymore to fulfill one or more requests in a profitable way. Subsequently arriving requests are handled by updating the existing transportation plan by altering the processes.

2.3. Coordination of the Consortium Partners

The two considered consortium members (agents) aspire different and to a certain extend contradicting planning goals. While the fleet managing agent aims at maximizing his profit by keeping costs as low as possible in each re-planning scenario the coordinator agent targets to achieve the promised least punctuality degree ($p_t \geq p^{target}$) whenever a re-planning is carried out. The strive for minimizing the operational costs restrains the fleet managing agent from investing additional expenditures to increase the punctuality rate if this rate has fallen below the threshold value, i.e. if the LSP charges are quite high. For this reason, the contract between the coordinator and the service agent must contain some specific measures in order to ensure that the service agent's strive for profit maximization does not lead to a negligence of the coordinator's requirements.

Efforts to improve the quality of processes conjointly derived by consortium partners are subsumed under the term *coordination* [10]. Here, coordination of the superior coordinator agent and of the subordinate fleet manager is necessary in order to close an information gap caused by information asymmetry among these two agents [11]. While the fleet manager maintains detailed information about the transport processes and resources the coordinator agent has (global) information about the system workload, expected demand or the current punctuality of supply consortium wide processes. Coordination is achieved by interchanging information between the two agents and using it for adjusting processes to the current system state.

A direct way to ensure the achievement of the desired least punctuality rate is to force the subordinate service providing agent to generate only processes fulfilling the least punctuality requirement (*brute-force coordination*). Every process proposal which does not obey the least punctuality condition and which leads to a lower punctuality will be rejected by the coordinator. The minimization of costs is only addressed as a second rang desire. It is not a mandatory planning requirement. The achievement of the least punctuality is the superior planning goal. We refer to this configuration as the hard condition configuration (HARD-configuration) of the investigated supply consortium scenario.

The least (minimal) punctuality rate p^{target} has been agreed between the coordinator and the fleet manager. An average workload has been assumed while fixing the agreed service level p^{target}. A significant increase in the number of customer sites (workload peak) augments the process costs and therefore lowers the profit of the transport partner. It is quite unfair that the additional expenses are not shared with the coordinator (and thereby among all supply consortium partners). Thus, a strict enforcement of the least punctuality discriminates the transport partner and enforces him to leave the consortium as soon as possible in order to prevent serious financial damage. For this reason, the HARD configuration is neither realistic nor applicable. We use it as reference configuration to provide comparable results for simulation experiments. These reference results enable an estimation of the costs necessary for ensuring the achievement of the service goal.

In a more elaborated coordination scheme, the coordinator must provide incentives to each service providing partner in the supply consortium if the partners act in the sense of the common goals instead of acting only in the sense of their own interest. For each partner, a major motivation to participate in the supply consortium is to maintain or increase the own profit. Vice versa, the attempt of a partner for maximizing its profit enables the supply consortium controller to influence and regulate the behavior of this partner. Here, the fleet manager is promised a higher benefit if it acts in accordance with the common supply consortium wide goals but its profit is reduced if the fleet manager refrains from acting in the sense of the coordinator.

The main idea of the penalty configuration (PEN-configuration) is to monetarily penalize the fleet manager for each request whose on-site fulfillment starts with a delay. Thereby, this partner is motivated to fulfill as much requests as possible on time so that the punctuality rate p^{target} can be guaranteed. If a demand peak occurs then the fleet manager freely decides whether to accept the profit reduction or to spend more efforts to maintain or even increase the service level. Here, the negative impacts of a workload peak are shared between the coordinator (and therefore among all partners in the consortium) and the fleet manager: The latter pays penalties for late arrivals but the supply chain consortium accepts a temporarily reduced punctuality.

2.4. Dispatching Task of the Fleet-Managing Agent

A sequence of transportation plans $TP_0, TP_1, TP_2, \ldots$ is generated reactively at the ex ante unknown update times $t_0, t_1, t_2, \ldots$ and each single transportation plan is executed as long as it is not updated. In order to determine the transportation plan TP_i at time t_i a static decision problem P_i requires a solving.

The problem P_i represents the task of selecting the least cost transportation plan from the set of all transportation plans available at time t_i. Thus, P_i is an optimization problem and the sequence $P_0, P_1, \dots$ is an online optimization problem representing the dynamic decision problem of the subordinate fleet managing agent.

In the following, we propose a mathematical model M_i for each instance P_i of this online optimization problem. A single request r attains consecutively different states that change with ongoing time. Initially, r is **k**nown but it is not yet scheduled (K). Then, r is assigned to an own vehicle (I, short for **i**nternal fulfillment) or subcontracted (E, short for **e**xternalization). If the operation at the corresponding customer site has already been started but not yet been finished the state S (short for **s**tarted request) is assigned to r. The set $R^+(t_i)$ is composed of additional requests released at time t_i. Requests completed after the last transportation plan update at time t_{i-1} are stored in the set $R^C(t_{i-1}, t_i)$. At time t_i, the recent request stock $R(t_i)$ is determined by $R(t_i) := R(t_{i-1}) \cup R^+(t_i) \setminus R^C(t_{i-1}, t_i)$. Each request belongs at each time to exactly one of the sets $R^K(t_i)$, $R^E(t_i)$, $R^I(t_i)$ or $R^S(t_i)$, in which the requests having a common state are collected.

The plan updating problem P_i at time t_i is as follows. Let V denote the set of all own vehicles, $P_v(t_i)$ the set of all paths (sequence of visited sites beginning with the position of the vehicle at time t_i and ending with the central depot) executable by vehicle v in TP_i and let $P(t_i)$ denote the union of the sets $P_v(t_i)$ $(v \in V)$. If the request r is served in a path p then the binary parameter a_{rp} is set to 1, otherwise it is set to 0. A request r, already known at time t_{i-1} that is not subcontracted in TP_{i-1} is served by vehicle v_r. The travel costs associated with path p are denoted as $C^1(p)$. Finally, $C^3(r)$ gives the subcontracting costs of request r.

In order to code the necessary decisions for determining a transportation plan in the representation M_i of P_i, we deploy two families of binary decision variables. Let $x_{pv} = 1$ if and only if path $p \in P(t_i)$ is selected for vehicle $v \in V$ and let $y_r = 1$ if and only if request r is subcontracted.

$$\sum_{p \in P(t_i)} \sum_{v \in \mathcal{V}} C^1(p) x_{pv} + \sum_{r \in R(t_i)} C^3(r) y_r \rightarrow \min \tag{1}$$

$$\sum_{p \in P_v(t_i)} x_{pv} = 1 \qquad \forall v \in \mathcal{V} \tag{2}$$

$$x_{pv} = 0 \qquad \forall v \in \mathcal{V}, p \notin P_v(t_i) \tag{3}$$

$$y_r + \sum_{p \in P(t_i)} \sum_{v \in \mathcal{V}} a_{rp} x_{pv} = 1 \qquad \forall r \in R(t_i) \tag{4}$$

$$y_r = 1 \qquad \forall r \in R^E(t_i) \tag{5}$$

$$\sum_{p \in P_{v(r)}} a_{rp} x_{pv_r} = 1 \qquad \forall r \in R^S(t_i) \tag{6}$$

$$p_t \geq p^{target} \tag{7}$$

$$x_{pv} \in \{0,1\} \; \forall p \in P(t_i), \; y_r \in \{0,1\} \; \forall r \in R(t_i) \tag{8}$$

For the HARD-configuration the process-planning problem is represented by the mathematical optimization model M_i stated in (1)-(8). The costs for TP_i are minimized (1). One route is selected for each vehicle (2) and vehicle v is able to execute the selected path p (3). Each single request known at time t_i is either served by a selected vehicle or forwarded to the LSP (4) but a once subcontracted request cannot be re-inserted into the path of an own vehicle (5). An (S)-labeled request cannot be re-assigned to another vehicle or LSP (6) and overall, the percentage p^{target} of all requests must be scheduled in time (7).

The model (1)-(8) is NP-hard to solve since it represents the traveling salesman problem in a specific parameter setting.

$$\sum_{p \in P(t_i)} \sum_{v \in \mathcal{V}} \left(C^1(p) + C^2(p) \right) x_{pv} + \sum_{r \in R(t_i)} C^3(r) y_r \rightarrow \min \tag{9}$$

In the PEN-configuration the punctuality constraint (7) is skipped and the objective function (1) is replaced by the evaluation function (9) that incorporates the penalty payments $C^2(p)$ for lateness. Penalties associated with p are summed up to $C^2(p)$ from all late customer site visits according to p. According to (9) each late arrival at a customer's site is penalized independently from the fact if $p_t \geq p^{target}$. Doing so, we have a unique quantification of lateness of the scheduled requests. We cannot uniquely identify those late requests (among all late requests) that finally cause the decrease of p_t below p^{target}. Therefore, we use the "more strict" penalization scheme coded into (9). However, the penalties summed up in (9) do not necessarily have to be accounted in complete to the fleet manager's budget.

Parameterizable test cases for the presented online optimization problem are described in detail in [3].

3. Decision Support for Dynamic Transport Process Planning

This section summarizes the state of the art for the development of decision support systems applied to the process management in volatile environments. A compilation of the basic principles of DSS for the management of processes in a dynamic environment introduces this section (Subsection 3.1). Next, we focus on the event handling by DSS (Subsection 3.2). Finally, we connect the paradigm of rolling horizon planning paradigm with DSS (Subsection 3.3).

3.1. Automated Transport Process Control by Decision Support Systems

The purpose of using a decision support system is to assist or even automate the setup and update of processes. According to [12], a DSS consists of three parts: the data base management system (DBMS), the model-base management system (MBMS) and the dialog generation and management system (DGMS). All data previously collected and/or necessary for deriving appropriate decisions are managed by the DBMS. The MBMS hosts formalized representations of the decision tasks (decision models) to be used for deriving decisions in a specific data setting. Tools for enabling the interaction of the DSS with its users are contained and managed by the DGMS.

Figure 2 outlines the usage of a DSS for contributing to a transport process revision. The gray-shaded area represents the environment in which a previously generated process is running. A process revision cycle is initiated by an event occurring in the environment and disturbing the planned execution of the current process instance (1). Immediately after the detection of the disturbing event, an update of the planning data hosted in the DBMS is triggered (2) and the necessary data revisions are established (3). After the necessity of a process revision has been determined by an analysis of the altered data, the MBMS component of the DSS is requested to instantiate a new suitable decision model (4). The MBMS selects an appropriate decision model type and parameterizes it using the stored planning data (5). The complete decision model is forwarded to a decision making algorithm and one or more proposals for an update of the process are derived using computational methods from operations research and/or artificial intelligence (6). These proposals are handed over to the DGMS that prepares the presentation of the proposals towards the responsible decision making agent (7). One proposal (8) is then implemented and executed (9) until the next process disturbing event is detected.

The components and actions (1) to (5) in Figure 2 form the **model building phase** in a process revision cycle. At the end of the model building phase, a formalized representation (the decision model) of the current decision task is available. At the second phase of an update cycle, the components and activities (5)-(9) are grouped to the **model solving phase** in which an implementable solution of the set up decision model is identified, selected and implemented. The MBMS (5) connects the two phases. Thus, the management of the formal decision model plays a central role for the process control.

The model solving phase is investigated in numerous research groups that want to speed up the solving of complex decision models. Two striking facts

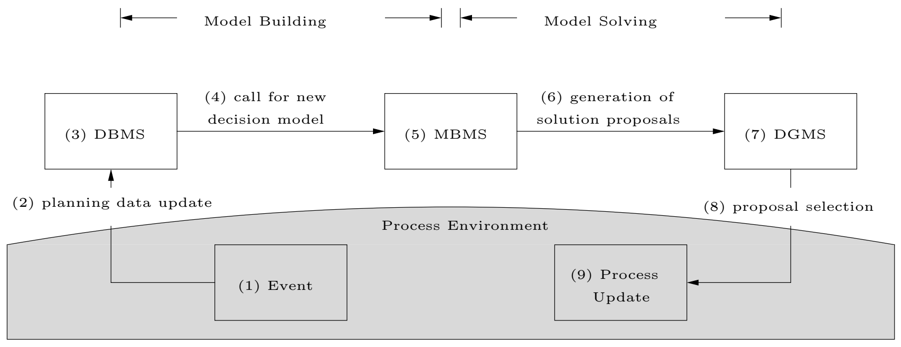

Figure 2: Process Revision Cycle in a Decision Support System

are extracted from the scientific literature about the model building subsystem of DSS for the operational management of transport processes in dynamic environments:

- Only one decision model type is maintained in the DSS. The type depends upon the kind of requests to be served, the number of depots and the kind of transport resources.
- There are fixed rules that describe how a new decision model instance is parameterized with the updated planning data. These rules remain unchanged throughout the complete running time of the DSS. They are not changed over time or in response to an unexpected event.

3.2. Event-Handling in DSS for Dynamic Transport Dispatching

[4] propose a generic three-layer architecture for DSS used to handle dynamically emerging events (cf. Figure 3) in the administration of transport processes. Adjacent layers communicate by exchanging messages. If a layer is not able to handle an event, it informs the next higher layer and asks for support from there. A response to a received message is replied to the lower layer after the message has been processed.

The lowest layer is the *interface layer* which hosts the *event characterizer* and the *instructor*. If an unexpected event is detected then an **event message** is sent to the event characterizer that checks whether the processes must be revised or not. In the latter case the instructor sends an **all-clear implementation message** back to the field teams (drivers and/or electronic on-board-units), which are currently waiting for an answer from the DSS. A typical event handled by the interface layer is the completion of a request. In the former case, the event leads to a process corruption and a **call for revision** of the process is sent to the next layer (*choice layer*).

In the choice layer, the *verbalizer* receives the call and analyzes the process corruption caused by an event. The *selector* tries to remedy the corruption.

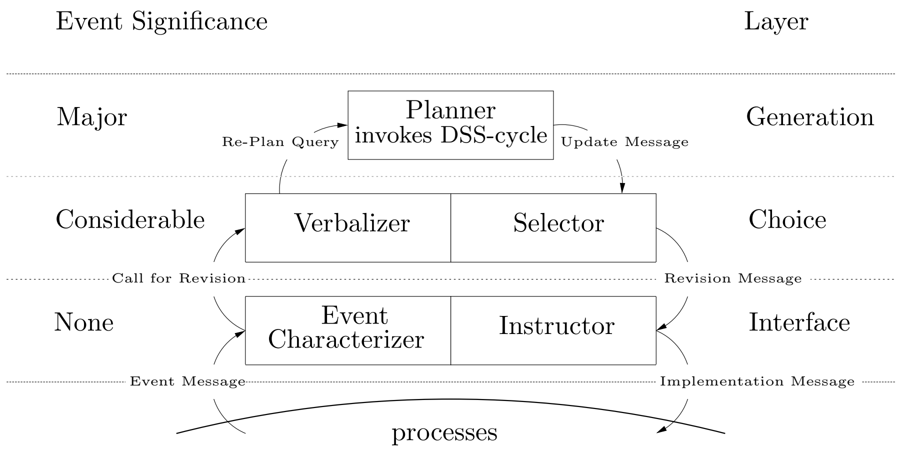

Figure 3: Three-layer model for event handling dynamic in transport process management

If it succeeds then a **revision message** is given down to the interface layer that broadcasts the relevant implementation message to the vehicles. A typical example for an event to be handled by the choice layer is an occurring traffic congestion that endangers the compliance of announced arrival times. Here, the selector defines a new path for the affected vehicle(s) without shifting the arrival time at a customer site out of the time window or re-assigning requests to other vehicles. If the corrupting event causes a major corruption that cannot be solved in the second layer then a global re-planning is triggered by sending a **re-plan query** from the choice layer into the *generation layer*.

The generation layer only hosts the *planner* which derives process update proposals. An additionally arriving request is an example for an event that requires the call of the planner. Its update proposals are fed back in an **update message** to the selector in the choice layer. If the event is handled by the choice layer then the application of local process revisions leads to a process recovery. In case that an event cannot be handled by the choice layer the process revision cycle (1)-(9) given in Figure 2 is started in the generation layer. A new decision model using up-to-date planning data is instantiated and solved. The solution of the model is sent back to the choice layer in an update message.

3.3. Rolling Horizon Disposition and Dispatching

The update cycle (1)-(9) reported in Subsection 3.1 provides the infrastructural framework for the automatic process update if major process corruptions have been detected. The basic principle to inject the recently acquired process data into the running process is rolling horizon planning. A sequence of plans $S_0, S_1, \dots$ (S_i is a solution of the decision model M_i of problem instance P_i) is generated. At time t_i the plan S_i is derived and its realization is initiated. It is continued with the execution of S_i until, at time t_{i+1}, additional data is revealed. The plan S_i is replaced by S_{i+1} and S_{i+1} is executed until it is corrupted at time t_{i+2} by events providing additional data and so on. The solution update is carried out if a pre-specified time point is reached [13, 14, 15, 16] or if one or more certain events take place [17, 18, 19]. In the first case, the update of S_i is called *time-triggered* but in the second case the revision is *event-triggered*. Two general concepts for the update of S_i to S_{i+1} are distinguished. *Rule-based* updating follows the hypothesis that a few basic reasoning rules are valid and that it is possible to inductively reason the behavior in all other cases not explicitly stated in the basic rules [20]. The a-priori-route-concept [21, 22] is an example of rule-based updating. Additionally, update rules like MST-algorithms [19] or cheapest insertion approaches [17] are representative examples for rule-based reasoning. A deductive reasoning is carried out in *model-based* update. Here, the set of all possible update alternatives is implicitly described by a formalized problem description (the decision model M_i) and a structured scanning of the set of alternatives leads to the desired solution. Examples of model-based approaches include the linear programming based optimization of a traveling salesman's route and the solving of a mixed-integer linear programming model of the capacitated vehicle routing problem. Typically, mathematical optimization models are selected as decision problem representation.

4. A DSS for Online Model-Based Process Planning

We report the development and evaluation of an event-triggered automatic model-based transport process update system. In a real-time setting event-triggered rolling horizon disposition comes along with some disadvantages (substantial workload slows the systems response time down; if dynamism is low then processing time is not used for further plan improvements). Since we are not aiming at real-time decision making but want to investigate online decision making features, these shortcomings can be ignored here.

At first, the layout of the planning system is outlined (Subsection 4.1). Next, the experimental setup of computational simulation experiments is given (Subsection 4.2). Process performance indicators are defined (Subsection 4.3). Simulation experiment results are presented and discussed in Subsection 4.4.

4.1. Planning System Layout

We use the mathematical optimization model (1)-(8) as process model maintained by our DSS. The application of the coordination scheme HARD requires the elimination of delayed arrivals at customer sites. In case that a solution of the internal process model exhibits a punctuality rate p_{t_i} less than p^{target}, the solution is repaired using the procedure REPAIR() shown in Figure 4.

As discussed above there are application obstacles associated with the HARD coordination idea discussed in Section 2 that impede the application of HARD to align the requirements of the superior principal and the requirements of the subordinate agent. Therefore, we also model the PEN-coordination approach. In the PEN-configuration, the constraint (7) is skipped and objective function (9) replaces (1).

1. All requests r which are fulfilled unpunctually are collected in the set S_1. If S_1 is empty then goto (4).
2. For each request $r \in S_1$ the savings $s_1(r)$ are calculated. Here, $s_1(r)$ is defined as the difference between the travel costs saved by deleting r in its tour and the subcontracting costs F_r. The requests contained in S_1 are sorted in decreasing order according to $s_1(r)$.
3. Finally, the fulfillment mode of the first request in the sorted list (the one with the highest savings) is switched to subcontraction and the request is deleted from the list. Goto (1)
4. The repair has been completed.

Figure 4: procedure REPAIR()

In order to generate and evaluate tentative model solutions we apply the Memetic Algorithm (MA) comprehensively presented in [3]. The MA realizes a hybrid search strategy consisting of a global genetic search space sampling and a local 2-opt improvement procedure for solving the scheduling model instances $M_0, M_1, \ldots$ of the online decision problem introduced in Subsection 2.

The proposed DSS has been implemented as a simulation of a multi-agent-system on a standard PC (Intel Core2 Duo CPU, T7500 Processor 2.2 GHz, 2GB RAM, Windows XP SP3). It has been programmed in Visual C++.

4.2. Configuration

The minimum punctuality (the reference signal) to be achieved throughout the complete simulation is set to $p^{target} = 0.8$.

The suitability of the PEN- and the HARD-technique for integrating the planning goals of the coordinator and the fleet manager is investigated within several numerical simulations. An experiment $(\alpha, tech)$ is defined by selecting one of the six values for the LSP-tariff levels $\alpha \in \{1, 1.25, 1.5, 1.75, 2, 3\}$ and combining it with one of the two coordination techniques $tech \in \{HARD, PEN\}$. If the tariff level α equals 1 then both request fulfillment nodes self-entry and subcontraction have the same costs. If α is increased then the LSP-incorporation becomes more expensive compared to the usage of own vehicles (even if time window violations are penalized).

We simulate each scenario $(\alpha, P, tech, \omega)$ in three independent runs $\omega \in \{1, 2, 3\}$ starting with differently seeded initial populations. From the four Solomon instances $P \in \{R103, R104, R107, R108\}$ we derive the request set $R^+(t_i)$ for each instance $P(t_i)$. Overall, there are 6×2=12 experiments $(\alpha, tech)$ leading to 12×4×3=144 simulated scenarios $(\alpha, P, tech, \omega)$.

For the PEN-coordination strategy, we define the following penalization function $C^3(r)$. If the request r is scheduled in time, the penalty $C^3(r)$ is zero for the associated single customer site, $C^3(r)$ increases proportionally up to 25 monetary units for a delay of 100 time units. Further delays do not lead to additional penalties. Initial experiments with different penalization schemes have revealed that *infinite right shifts* of requests do not occur because there are no spatially isolated request locations in the scenarios.

The processing time for each simulated scenario is about 6 minutes in average independently from the applied coordination technique HARD or PEN.

4.3. Observed Indicators

The punctuality rate recorded at time t within the scenario $(\alpha, P, tech, \omega)$ is denoted as $p_t(\alpha, P, tech, \omega)$. Let $p_t(\alpha, tech) := \frac{1}{12}\sum_{\omega=1}^{3}\sum_{P\in\mathcal{P}} p_t(\alpha, P, tech, \omega)$ denote the average punctuality observed at time t for the parameter combination $(\alpha, tech)$.

In order to study the impacts of the demand peak on the punctuality, we calculate the deviation of $p_t(\alpha, P, tech, \omega)$ from the reference value $p_{1000}(\alpha, P, tech, \omega)$ for all times in the observation time interval $[1000, 5000]$ by

$$p_t(\alpha, P, tech, \omega)/p_{1000}(\alpha, P, tech, \omega) - 1. \tag{10}$$

The largest past-peak deviation from the reference value is then calculated by $\frac{\min_{t\geq 1500}\{p_t(\alpha,P,tech,\omega)\}}{p_{1000}(\alpha,P,tech,\omega)} - 1$. Now, the average $\delta(\alpha, tech)$ of the largest observed deviation from the reference values for the parameter combination $(\alpha, tech)$ is defined as shown in (11).

$$\delta(\alpha, tech) := \frac{1}{12}\sum_{\omega=1}^{3}\sum_{P\in\mathcal{P}}\left(\frac{\min_{t\geq 1500}\{p_t(\alpha, P, tech, \omega)\}}{p_{1000}(\alpha, P, tech, \omega)} - 1\right). \tag{11}$$

Let $T^{below}_{\alpha,tech}$ denote the first time in which $p_t(\alpha, tech)$ falls below p^{target} and $T^{heal}_{\alpha,tech} := \min\{t \in [1000, 5000] \mid \nexists\, l \in [t, 5000], p_l < p^{target}\}$ referring to the time in which an HQ state is finally re-achieved by $p_t(\alpha, tech)$. We define

$$\pi(\alpha, tech) := \frac{T^{heal}_{\alpha,tech} - T^{below}_{\alpha,tech}}{4000} \tag{12}$$

determining the percentage of LQ periods within the observation interval $[1000, 5000]$.

Throughout the simulation time, we have recorded the percentage of subcontracted requests in $q_t(\alpha, P, tech, \omega)$. These values have been summarized in

$$q_t(\alpha, tech) := \frac{1}{12}\sum_{\omega=1}^{3}\sum_{P\in\mathcal{P}} q_t(\alpha, P, tech, \omega) \tag{13}$$

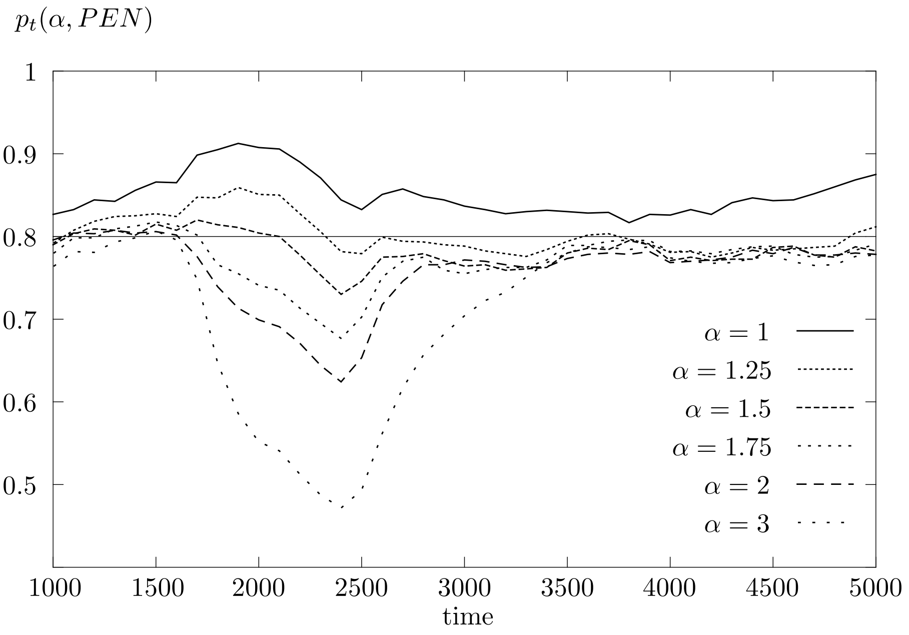

Figure 5: Development of the punctuality $p_t(\alpha, PEN)$

for each setting $(\alpha, tech)$. The observed maximal subcontraction rate is defined by $\sigma(\alpha, tech)$ and calculated according to (14). It indicates the exploitation of the subcontraction fulfillment mode.

$$\sigma(\alpha, tech) := \max_{t \geq 1500} q_t(\alpha, tech). \tag{14}$$

4.4. Results

In contrast to the results already published in [2] for scenarios with $\alpha = 3$, we here analyze the impacts of variations of α starting from $\alpha = 1$ up to $\alpha = 3$ with respect to the previously defined performance indicators. Especially, we are interested in the growth of the gap between the indicator values observed in the PEN- and the indicator values observed in the HARD-experiments in case that α increases.

The PEN-approach can guarantee to keep the threshold of 80% for the average punctuality only for $\alpha = 1$ (Figure 5). The punctuality even increases just after the demand peak is over ($t \geq 1800$) because the routes of the own vehicles are compiled from a larger number of available requests. So, a higher number of matching requests is found. If the tariff level α exceeds 1.25, the LSP-incorporation becomes more and more unattractive. Its actual costs are getting higher than the travel costs plus penalty payments. As a consequence, the usage of own trucks is preferred although it does not come up with an on-time service.

Table 1: Maximal punctuality deviation $\delta(\alpha, tech)$

	$tech$	α					
		1	1.25	1.5	1.75	2	3
$\delta(\alpha, tech)$	$HARD$	3.6%	0%	5.3%	4.4%	4.2%	3.5%
	PEN	-1.0%	-1.8%	-7.8%	-13.6%	-22.0%	-38.8%
$\pi(\alpha, tech)$	$HARD$	0%	0%	0%	0%	0%	0%
	PEN	0%	60.0%	70.0%	97.5%	82.5%	97.5%
$\sigma(\alpha, tech)$	$HARD$	22.2%	14.9%	10.0%	8.0%	7.2%	8.0%
	PEN	21.4%	15.5%	10.0%	5.8%	5.1%	4.1%

Overall, the HARD-configuration outperforms the PEN-configuration. We first analyze the maximal decrease of the punctuality rate p_t. The evolution of this parameter is completely different for the two controller configurations. If PEN is used then the maximal decreasing rate of p_t falls from -1.0% ($\alpha = 1$) down to -38.8% ($\alpha = 3$) as it can be seen in Tab. 1. In contrast, we observe that a tariff level increase hardly influences the punctuality if the consortium is organized according to the rules of HARD. Here, no significant decrease of p_t is detected. The slight increase of the punctuality during the acute peak management is caused by the fact that the increased number of available requests enables an improved compilation of routes without being endangered to achieve a too low punctuality.

We now draw our attention to the appearance of LQ-situations. Clearly, it is $\pi(\alpha, HARD) = 0$ for all investigated tariff levels α. In addition, the HARD-configuration is able to slightly enlarge the punctuality compared to the referential value at time $t = 1000$. The additional knapsack-constraint enables the memetic search to evaluate different separations of the request portfolio into self-fulfilled and subcontracted requests. Since the constraint (7) ensures that at least 80% of the requests are in time, no penalty costs contradict the route composition. Due to the parameter sensitivity of the punctuality $p_t(\alpha, PEN)$ to tariff level increases, the percentage $\pi(\alpha, PEN)$ of LQ-situations increases from $\pi(1, PEN) = 0$ up to $\pi(3, PEN) = 97.5\%$. Therefore, a short-time demand peak has a long lasting negative impact on the punctuality of the service.

The maximal rate σ of subcontracted requests decreases with increasing tariff level α. For comparable freight tariffs ($\alpha \leq 1.5$) both configurations behave similarly with respect to the subcontraction of requests. The maximal externalization rate $\sigma(\alpha, tech)$ is nearly the same in both cases for each $\alpha \leq 1.5$. However, if the tariff levels climb further then $\sigma(\alpha, HARD)$ remains stable at $\approx 8\%$ while $\sigma(\alpha, PEN)$ falls further down to $\approx 4\%$.

A reason for the bad performance of the PEN-configuration is its non-observance of the reliable subcontracting services if its costs are significantly higher than the sum of self-fulfillment costs and penalty payments. For $\alpha = 1$ the maximal percentage of subcontracted requests is $\sigma(1, PEN) = 21.4\%$ but for large tariff levels, this portion is significantly reduced down to $\sigma(3, PEN) = 4.1\%$. In the same situation, the HARD resource allocation strategy identifies the externalization as the preferred mode for nearly the double number of requests (8.0%).

5. Enhanced DSS with Image Modification

In this section, we propose a structural extension of the process control system introduced in the previous section. The motivation is to equip a DSS to manage even crisis situations like workloads in which state-of-the-art systems are not able to maintain a sufficiently high process quality. *Image Modification (IM)* [23] is used to modify the maintained process model used to update existing models. The objective function or the constraints of the maintained model are altered and changes of the model are made after having evaluated the current process performance. From the perspective of the supply chain IM enables the alignment of the decision goals of the supply consortium coordinator and of the behavior of the fleet manager. Thus, IM enables the coordination of the coordinator agent and of the fleet managing agent in the consortium.

In case that an event disturbs a process, the three-layer-model introduced in Subsection 3.2 is invoked. The type of the DSS response depends on the severeness (or impacts) of a disturbing event. A considerable event is managed without incorporating the process model but according to some pre-given process update rules (e.g. a delayed arrival at a customer side without deferments of subsequent arrivals). Only an event of major significance leads to the re-call of the process planner (the process controller) with the aim to generate a process control signal which enables the update of the corrupted process (e.g. the arrival of additional requests that require a re-compilation of the tours and re-sequencing with the routes). However, not all events can be successfully managed by the proposed event handling as we have seen by means of the example of a load peak in the previously reported simulation experiments. Here, the planner is not able to return a process update message that implies a process update coming along with a sufficiently high punctuality even if PEN is applied. If such a *threatening* event is detected IM intervenes into the regular DSS-process-management cycle (1)-(9) shown in Figure 2 and IM varies the internal process model of the process with the goal to enable the generation of a feasible process update.

In order to enable the DSS to handle such a threatening event and thereby invoking an IM intervention, we extend the three-layer model of Séguin et al. (cf. Subsection 3.2) and add a fourth event handling layer. This additional layer (*coordination layer*, cf. Figure 6) is invoked by the third layer if a threatening event has appeared that cannot be handled in the current configuration (e.g. solving the mathematical process model) of the planning system. The third layer is extended by an *error signal generator* that compares the actual system state (p_t) with the desired state ($p_t \geq p^{target}$) and maps the detected deviation into a numerical value (the **error signal**). This value is forwarded into the fourth

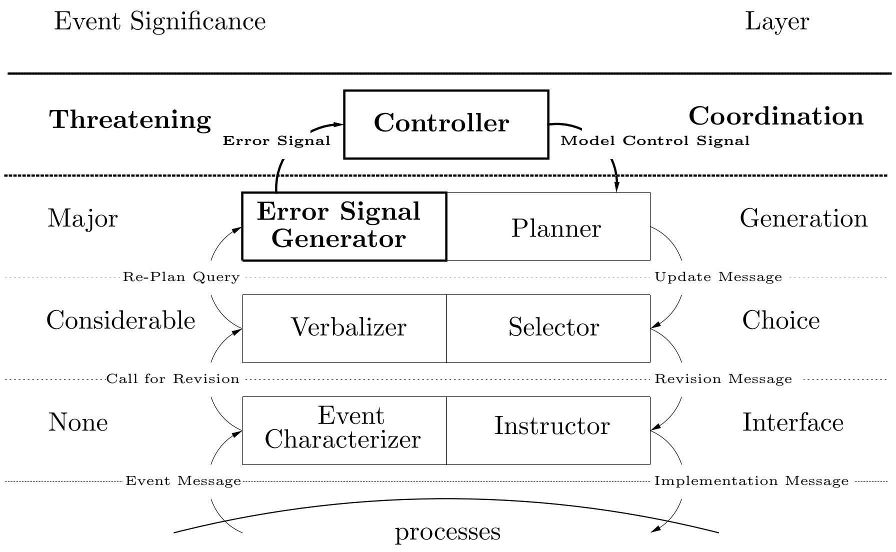

Figure 6: Four-layer event handling model for a process management system (the proposed extensions are shown in bold)

layer in which a model controller transforms the error signal into decision model modification instructions (**model control signal**). This signal is sent back into the generation layer. The planner is reconfigured by replacing the so far used process optimization model by the re-parameterized process optimization model. Then, the required process re-planning is carried out using the re-adjusted model (the *planner*-component). The application of IM enables the DSS to automatically select a suitable process update model out of the MBMS.

For the considered online decision situation the error signal is the deviation $p_t^{error} := -min\{p_t - (p^{target} + 0.1), 0\}$ of the current process punctuality p_t from the reference punctuality p^{target}. In order to prevent a reduction of p_t below p^{target} an error is already determined if the punctuality falls below $p^{target} + 0.1$. An *intervention strategy* determines how the so far used decision model is updated with respect to p_t^{error}.

$$h(p_t^{error}) := \begin{cases} 0, & p_t^{error} \leq 0 \\ 1, & p_t^{error} \geq 0.2 \\ 5 \cdot p_t^{error}, & 0 < p_t^{error} < 0.2 \end{cases} \tag{15}$$

An **intensity function** h scales the error signal into the interval $[0;1]$. The h-value determines the applied percentage of the possible interventions. We use the piece-wise linear intensity function given in (15) that is zero if $p_t \geq p^{target} + 0.1$ (no error, no intervention). If $p_t \leq p^{target} - 0.1$ then the h-value is 1 which means that all possible model modifications are applied. If p_t decreases from $p^{target} + 0.1$ down to $p^{target} - 0.1$ then the h-values increase proportionally from 0 up to 1, so that an appropriate portion of the available process decision model modifications is applied.

Existing constraints of the mathematical update model are temporarily sharpened or relaxed according to the current h-value. To apply the **Constraint Set ADaptation (CSAD)** intervention strategy the constraint (5) has been replaced by the constraint (16). The set $R^{intervention}(h(p_t^{error}))$ is adjusted before the next re-planning is performed. From the set R_t^+ of recently arrived but so far unscheduled requests the portion $h(p_t^{error})$ is put into $R^{intervention}(h(p_t^{error}))$ according to a so-called request selection rule SEL: $R^{intervention}(h(p_t^{error})) := H(h(p_t^{error}, R_t^+, SEL)$. A detailed description and evaluation of the CSAD-intervention is given in [2].

$$y_r = 1 \qquad \forall r \in R^E(t_i) \cup R^{intervention}(h(p_t^{error})) \tag{16}$$

Another possibility to adjust the maintained optimization model is to vary the objective function of the process update model. The objective function (17) replaces (1). This objective function does not determine the actually incurred costs of the generated transportation plan but the fictitious costs, which are actually accounted to the budget available for the fleet manager within the consortium [3]. The service agent aims at minimizing this amount in order to maximize its own remaining profit.

$$\lambda_t \cdot \sum_{p \in P(t)} \sum_{v \in \mathcal{V}} \left(C^1(p) + C^2(p)\right) x_{pv} \quad + \quad \cdot \sum_{r \in R(t)} C^3(r) y_r \rightarrow \min \tag{17}$$

The parameter λ_t is adjusted to p_t^{error} by setting $\lambda_t := 1 + h(p_t^{error}) \cdot \alpha$. If the error signal is zero then λ_t equals one. As soon as p_t^{error} starts growing, the parameter λ_t is increased until it finally reaches the value $1 + \alpha$ which makes the subcontraction of additional requests more attractive then the integration of additional requests into existing routes. Since subcontracted requests are fulfilled timely, a further decrease of p_t is prevented.

Since the variation of λ_t changes the search trajectory of an exact or heuristic model solver through the search space (described by the constraint set), we call the feed-back triggered modification of (17) **Search Direction ADaptation (SDAD)**. A detailed description of the application of the SDAD-based model controller can be found in [3].

Beside the limitation of the adaptation of the objective function or the constraint set a third adaptation opportunity comprises the simultaneous variation of the objective function and of the constraint set. Before the next process update is carried out, both the constraint set as well as the objective function are adjusted to p_t^{error}. We refer to this hybrid adaptation strategy as **(SDCS)**.

The pseudo-code of the SDCS-based process update function is presented in Figure 7. Both the so far used process decision model and the so far implemented process are provided for the update (1). The process update time is fetched (2) and the additionally arrived requests are collected (3). In step (4) the current process punctuality is calculated. Next, the error signal is determined (5). The

```
1. **function** update_process_SDCS($M^{old}$, $Proc^{old}$);
2. $t$ := fetch_current_time();
3. $R_t^+$ := get_new_requests();
4. $p_t$:= get_current_process_punctuality($Proc^{old}$);
5. $p_t^{error} := -min\{p_t - p^{target}, 0\}$;
6. $(\lambda_t, R_t^{intervention}) := (1 + h(p_t^{error}) \cdot \alpha, H(h(p_t^{error}), R_t^+, SEL))$;
7. $M^{new}$:=update_model($M^{old}$, $\lambda_t$, $R_t^{intervention}$);
8. $Proc^{new}$:=solve($M^{new}$);
9. **return**($M^{new}$, $Proc^{new}$);
10. **end**;
```

Figure 7: Pseudo-code representation of the process update function using SDCS

model modifications are fixed in step (6) and the new decision model is defined (7). Now, the process update is generated by solving the new decision model (8) and the new process decision model as well as the new process is returned to the DSS-cycle (9).

The application of IM demonstrates the superiority of the coordinator over the fleet manager. However, this relationship can be beneficiary for both partners on the longer perspective. Although the interventions are myopically negative for the fleet manager (profit reduction), they ensure that the performance of the overall consortium is protected (high process reliability). The fleet manager benefits from this protection because it is part of the consortium. If another subordinate partner of the consortium is affected from coordinator interventions (like a production partner) then the fleet manager benefits from coordinator interventions if they prevent the process collapse.

6. Experimental Evaluation of the Extended DSS

We have repeated the experiments reported in Section 4. Now we apply the three model adjustment strategies SDAD, CSAD and SDCS using the LSP-tariff level $\alpha = 3$. In these scenarios, a workload peak impacts most severely as shown in Table 1. Again, the averaged scenario processing time is 6 minutes.

The development of the punctuality p_t of all five investigated strategies is printed in Figure 8. Two details are striking. At first, $p_t(SDCS)$ never leaves the system development corridor. Only a slight decrease of $p_t(SDCS)$ is observed after the peak of incoming demand/incoming requests is over. Secondly, SDCS outperforms not only PEN but also SDAD and CSAD with respect to the maintained punctuality. From these results, we conclude that the simultaneous application of both model adjustment strategies is more suitable than the application of one of the two strategies CSAD or SDAD.

All three adaptive strategies (CSAD, SDAD, SDCS) decide a significantly higher externalization quote q_t than the static strategies (HARD, PEN) dur-

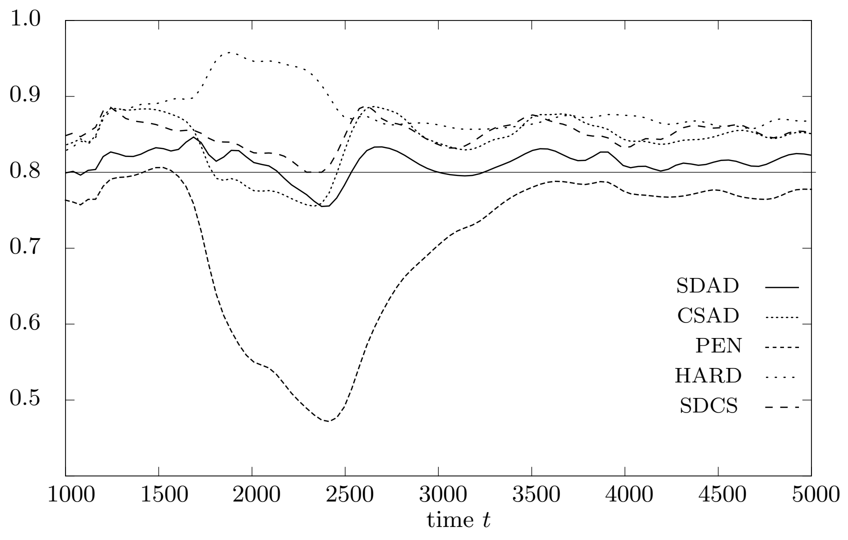

Figure 8: Development of the punctuality $p_t(exp)$

ing the experiments. SDCS is able to identify a higher percentage $q_t(SDCS)$ of requests to be given away to an LSP immediately after the demand peak is over than SDAD does (Figure 9). However, compared to CSAD, the percentage of requests selected by the coordinator for being subcontracted in the aforementioned period is lower ($q_t(SDCS) \leq q_t(CSAD)$). This means that the analytical capabilities of SDCS to identify portfolio incompatible requests are improved compared to CSAD. However, compared to SDAD, the strategy SDCS demonstrates a stricter and more assertive behavior: SDCS identifies more incompatible requests than SDAD, so that a higher percentage of requests are subcontracted if SDCS is applied.

An offline process quality and reliability assessment of SDCS uncovers that the hybrid adaptive strategy SDCS outperforms PEN and the two "individual" adaptive strategies SDAD and CSAD. On the other hand, it turns out that the quality gap between the HARD strategy and the hybrid adaptive strategy is closer than the gap between HARD and SDAD or HARD and CSAD respectively.

From the results compiled in Table 2 we learn that the relative decrease $\delta(SDCS)$ of $p_t(SDCS)$ is significantly smaller than the relative punctuality rate decrease $\delta(PEN)$. In addition, the relative decrease of the punctuality in the SDCS-experiments is less than the relative punctuality rate decrease $\delta(CSAD)$. The relative decrease $\delta(SDAD)$ is only slightly less than $\delta(SDCS)$.

Only HARD and SDCS ensures that no LQ-situations occur. All three other strategies (PEN, SDAD and CSAD) are not able to guarantee to keep the punctuality rate p_t above 80% throughout the complete simulation experiments.

Finally, we state that SDCS leads to the highest maximal externalization

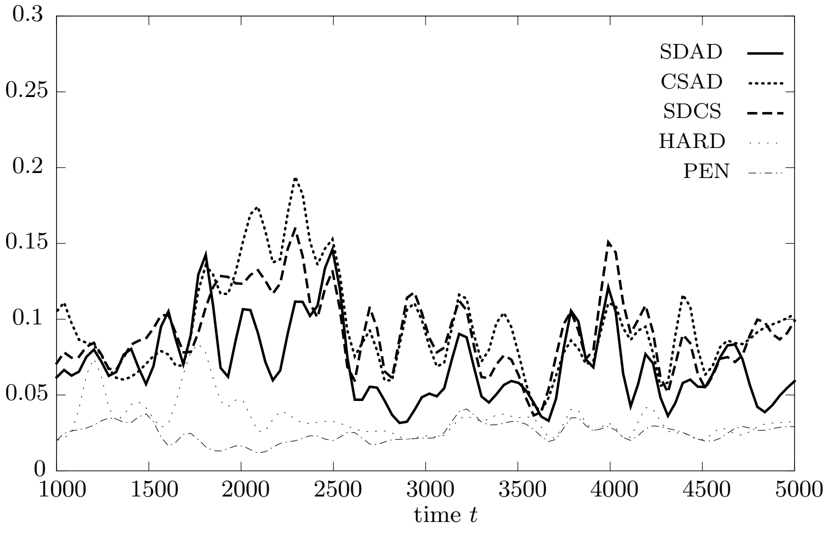

Figure 9: Percentage $q_t(exp)$ of externalized requests in the schedule generated at time t

quote $q_t(\cdot)$ among all five applied strategies. Although SDCS combines SDAD and CSAD, the observed maximal externalization quote $\sigma(SDCS)$ does not fall between the quotes $\sigma(SDAD)$ and $\sigma(CSAD)$ but it is significantly higher than $\sigma(SDAD)$ and even higher than $\sigma(CSAD)$.

Table 2: Value of the indicators for the offline process quality performance

	exp				
	HARD	PEN	SDAD	CSAD	SDCS
$\delta(exp)$	3.5%	-38.8%	-5.6%	-9.5%	-5.79%
$\pi(exp)$	0%	97.5%	16.7%	15.0%	0%
$\sigma(exp)$	8.0%	4.1%	14.5%	15.2%	15.9%

Table 3 consolidates the achieved cumulated costs (c_{5000}), the cost increase compared to the HARD-experiments (γ_{5000}) and the contributions of the three cost drivers, which are travel costs ($\bar{c}_{5000}^{travel}$), penalty payments ($\bar{c}_{5000}^{pen}$) and LSP-charges ($\bar{c}_{5000}^{ext}$). The reduced marginal costs of a request finally leads to a decrease of the cumulated request fulfillment costs $c_{5000}(SDCS)$ of SDCS compared to the costs observed for the CSAD-configured scenarios. A reduction from $c_{5000}(CSAD) = 82696.6$ to $c_{5000}(SDCS) = 78350.8$ money units is observed. However, the application of SDAD leads to significantly less costs than the application of SDCS.

SDCS exhibits the least portion of travel costs c_{5000}^{travel} among all five strategies. Only 37.6% of the total sum of expenditures are used to cover the distance-related expenses of the own fleet. At the same time, SDCS causes the least penalty expenditure percentages among the investigated approaches (2.7%). On the other hand, SDCS comes along with the highest portion of LSP-charges. In the experiments, 59.7% of the total costs are used to entrust LSPs with the fulfillment of incompatible requests. This value is maximal among the five adaptive strategies.

Table 3: Cumulated costs and contributions of cost drivers

	exp				
	HARD	PEN	SDAD	CSAD	SDCS
$c_{5000}(exp)$	56301.5	55748.3	64225.6	82696.6	78350.8
$\gamma_{5000}(exp)$	–	-1.0%	14.1%	46.9%	39.2%
$\bar{c}_{5000}^{travel}(exp)$	81.1%	84.5%	44.4%	40.7%	37.6%
$\bar{c}_{5000}^{pen}(exp)$	3.6%	12.0%	3.8%	3.6%	2.7%
$\bar{c}_{5000}^{ext}(exp)$	15.4%	3.5%	51.8%	55.7%	59.7%

7. Conclusions

We have presented recent research about DSS-improvements for processes running in a volatile environment that are managed by two decision making units (forming a supply consortium): a superior coordinator and a subordinate fleet managing agent. The research hypotheses stated in the report's introduction have been verified.

Within computational simulation experiments we have first approached the limits of model-based DSS applied in a volatile process environment: a collapse of the process punctuality caused by a short but severe workload peak is provoked. These simulation results have revealed that the interests of the superior supply chain coordinator cannot be sufficiently integrated into the DSS. The reasons for this bad performance of the DSS have been analyzed and it has turned out that coordination between the acting agents is not possible using a state-of-the-art DSS. It has been revealed that significant changes of the planning situation (the temporal scarceness of resources) have not been reflected into the formalized problem representation (process model) maintained by the DSS because the superior coordinator that knows about the varied planning situation has not been part of the DSS.

To remedy this shortcoming of a DSS we have proposed to improve the MBMS-component of the DSS by adding a component that emulates the behavior of the superior consortium coordinator. This component adjusts the maintained formal process model to a varied process environment so that unexpected planning situation variations can now be reflected into the process model. We have defined and evaluated the proposed extension for the investigated dynamic vehicle routing problem. Strategies to adjust the objective function and/or the constraint set of the maintained decision model have been set up and tested. We have been able to show that the enhancement of the DSS by coordination issues leads to a significant increase of the process quality even after the appearance of a workload peak. Thereby, we have verified our second research hypothesis.

Future research regarding enhanced DSS will follow two directions. On the one hand, we will analyze the applicability of multi-dimensional performance metrics for the transport scenario. Here, the major challenge is the consolidation of concurring or even contracting feedback signals into an applicable intervention directive. On the other hand, we will transfer the idea of integrating coordination-issues into DSS to other application areas in order to evaluate the general applicability of the proposed concept.

References

[1] K.C. Laudon and J.P. Laudon. *Management Information Systems*. Pearson - Prentice Hall, 9 edition, 2006.

[2] J. Schönberger. Adaptive demand peak management in online transport process planning. *OR Spectrum*, 32(3):831–859, 2010.

[3] J. Schönberger and H. Kopfer. Online Decision Making and Automatic Decision Model Adaptation. *Computers & Operations Research*, 36(6):1740–1750, 2009.

[4] R. Séguin, J.-Y. Potvin, M. Gendreau, T.G. Crainic, and P. Marcotte. Real-time decision problems: an operational research perspective. *Journal of the Operational Research Society*, 48(2):162–174, 1997.

[5] C. Schneeweiß. Elemente einer Theorie hierarchischer Planung. *OR Spektrum*, 16:161–168, 1994.

[6] A.C. Hax and H.C. Meal. Hierarchical integration of production planning and scheduling. In M. Geisler, editor, *Logistics*, volume 1 of *TIMS Studies in the management sciences*, pages 53–69. MIT, 1975.

[7] B. Brehmer. Dynamic decision making: Human control of complex systems. *Acta Psychologica*, 81:211–241, 1992.

[8] O.G.B. Madsen, K. Tosti, and J. Vælds. A heuristic method for dispatching repair men. *Annals of Operations Research*, 61:213–226, 1995.

[9] P. Beullens, D. Van Oudheusden, and L. Van Wassenhove. Collection and Vehicle Routing Issues in Reverse Logistics. In *Reverse logistics: quantitative models for closed-loop supply chains*, pages 95–134. Springer, 2004.

[10] H. Stadtler. A framework for collaborative planning and state-of-the-art. *OR Spectrum*, 31:5–30, 2009.

[11] S. Chopra and P. Meindl. *Supply Chain Managment*. Pearson Education, 2007.

[12] R.H. Sprague and E.D. Carlson. *Building effective decision support systems.* Prentice-Hall, 1982.

[13] H.K. Rajagopalan, C. Saydam, and J. Xiao. A multiperiod set covering location model for dynamic redeployment of ambulances. *Computers & Operations Research*, 35(3):814–826, 2008.

[14] D. Saéz, C. E. Cortéz, and A. Núñes. Hybrid adaptive predictive control for the multi-vehicle dynamic pick-up and delivery problem based on genetic algorithms and fuzzy clustering. *Computers & Operations Research*, 35(11):3412–3438, 2008.

[15] A. Erera, B. Karacık, and Martin Savelsbergh. A dynamic driver management scheme for less-than-truckload carriers. *Computers & Operations Research*, 35:3397–3411, 2008.

[16] S. Mitrović-Minić, R. Krishnamurti, and G. Laporte. Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B*, 38:635–655, 2004.

[17] B. Fleischmann, S. Gnutzmann, and E. Sandvoß. Dynamic Vehicle Routing Based on Online Traffic Information. *Transportation Science*, 38(4):420–433, 2004.

[18] K. Lund, O.B.G. Madsen, and J.M. Rygaard. Vehicle Routing Problems with Varying Degrees of Dynamism. Technical Report IMM-REP-1996-1, Institute of Mathematical Modelling (IMM), Technical University of Denmark, Lyngby, 1996.

[19] G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, and M. Talamo. Serving Requests with On-line Routing. In E.M. Schmidt and S. Skyum, editors, *Proceedings of 4th Scandinavian Workshop on Algorithm Theory*, pages 37–48. Springer, 1994.

[20] H. Lindstaedt. Problemlösen und Verstehen bei ökonomischen Agenten - eine Gegenüberstellung ökonomischer und kognitionspsychologischer Modelle regelbasierten Entscheidens. *NeuroPsychoEconomics*, 2:30–43, 2007.

[21] H. Tang and E. Miller-Hooks. Solving a generalized traveling salesperson problem with stochastic customers. *Computers & Operations Research*, 34:1963–1987, 2007.

[22] Y.-H. Liu. A hybrid scatter search for the probabilistic traveling salesman problem. *Computers & Operations Research*, 34:2949–2963, 2007.

[23] C Bierwirth. *Adaptive Search and the Management of Logistics System.* Kluwer, 2000.