4

Supply Chain Event Management System

Bearzotti Lorena[1], Fernandez Erica[2,3],
Guarnaschelli Armando[2,3], Salomone Enrique[2] and Chiotti Omar[2,3]
[1]*Universidad Nacional Andrés Bello, Campus Los Castaños 7 Norte 1348 Viña del Mar,*
[2]*INGAR- CONICET, Avellaneda 3657 Santa Fe,*
[3]*CIDISI – UTN FRSF, Lavaise 610, Santa Fe,*
[1]*Chile*
[2,3]*Argentina*

1. Introduction

The *Supply Chain Management* (SCM) can be defined as the set of proposals used to efficiently integrate suppliers, manufacturers and warehouses, such that the product is produced and distributed in the right quantity and at the right time, minimizing the total cost and satisfying the required service level (Simchi-Levi et al., 1999). To this aim, enterprises in a Supply Chain (SC) perform collaborative business processes (Soosay et al., 2008). Particularly, collaborative planning processes allow each enterprise to obtain production and/or distribution schedules synchronized with schedules of the other SC members (Derrouiche et al., 2008).

In this chapter, a *schedule* is defined as a set of orders, where each order represents a supply process (production or distribution) that assigns materials to a place, states the required resources, the time period during which each resource is required and its required capacity. The execution of a schedule implies performing the operations defined in the supply process each order represents.

As result of the uncertainty inherent in any supply process (Kleindorfer & Saad, 2005) disruptive events arise. The problems they cause during a schedule execution occur on a daily basis, and affect not only the organization where they are produced but also propagate throughout the SC (Lee et al., 1997; Radjou et al., 2002). That is, these disruptive events may affect the schedules and their synchronization.

In this chapter a *disruptive event* is defined as a significant change in the order specifications or planned values of resource availability. These changes could be: rush or delay in the start or end date of the order, changes in the amount specified by the order, change in the expected future availability of a resource, and change into the current level of a resource regards to its planned value. They can be produced by changes that can take place into the enterprise or outside the enterprise. For example, an equipment breakdown, breakage of materials, change of material specification, weather conditions, traffic congestion, etc.

The occurrence of disruptive events is a fact well known to the planning task, and therefore planning systems generate schedules including buffers (material, resource capacity and time) to be robust and flexible, thus the schedule can be adapted to conditions occurring during implementation (Van Landeghem & Vanmaele, 2002; Adhitya et al., 2007; Wang &

Lin, 2009; Bui et al., 2009; Liu & Min, 2008). The occurrence of disruptive events during the schedule execution requires an adequate response. If the effect cannot be mitigated, an exception occurs. In this work an *exception* is defined as a deviation from the schedule that prevents the fulfillment of one or more orders that requires re-planning.

Supply Chain Event Management (SCEM) is defined as an event-based business process whereby significant disruptive events are recognized in time, reactive actions are quickly triggered, the flow of information and material are adjusted and key employees are immediately notified. The goal of SCEM is to enable the SC to respond to disruptive events avoiding the need to re-plan the operations of the SC. To support this business process, a new generation of event-based information systems, known as SCEM Systems (Masing, 2003; Zimmermann, 2006), has been proposed. This proposal emphasizes the necessity of exception-based management of the SC, supporting short term logistic decisions, avoiding complex cycles of re-planning, aimed at reducing the gap between the planning system and the execution of the schedules generated by it.

In this chapter a proposal to systematically address the problem of disruptive event management in SC is described. Both academic and industrial researchers (Zimmermann, 2006, Radjou et al., 2002) have identified this problem as not being adequately covered by state of the art solutions in the SCEM systems. Moreover, the ability to automatically detect disruptions and repair them locally without affecting coordinated and coexistent schedules within a SC, is recognized to be a major competitive advantage in next generation of SCM systems.

2. SCEM system classification

From the view point of their automation levels, SCEM systems can be classified in the following types:

Monitoring system: Planned as an extension of traditional Tracking and Tracing Systems (Szirbik et al., 2000; Kärkkäinen et al., 2003) they allow the user monitoring planed events to detect disruptive events.

Alarm system: Can systematically detect deviations in the schedule and notify the key employee (Hoffmann et al., 1999; Speyerer & Zeller, 2004; Teuteberg & Schreber, 2005; Zimmermann, 2006).

Decision support system: Can detect deviation and find a solution that minimizes the disturbance impact on the SC. The solution will be proposed to the human decision-maker to make the final decision (Cauvin et al., 2009; Adhitya et al., 2007).

Autonomous corrective system: Able to detect a disruptive event, verify the feasibility of the current schedule or look for a solution to repair the schedule and implement it if one exists.

From the view point of their monitoring strategy, the SCEM systems can be classified in the following types:

Order focus: The monitoring task is centred on the orders. As disruptive event captures any significant change i to the specification of an order, ΔO_i. These include: rush or delay in the start or end date of the order, changes in the carrying amount of the order, cancellation of order, new order.

Order and Resource focus: The monitoring task is centred on the orders and resources associated with an order. As disruptive event captures any significant change in the planned value of the resource j, ΔR_j, which produces significant change i to the specification of an order, ΔO_i. To infer the change in the order specification, a change propagation function is used, which can be represented as in equation (1):

$$\Delta O_i = \mathrm{f}(\Delta R_j, \text{ for all } j), \text{ where}$$
$$\Delta R_j: \text{ change of the resource } j \tag{1}$$

Order, Resource and environment focus: The monitoring task is centred on the orders, resources associated with them and environmental variables. As disruptive event captures any significant change i to the specification of an order, ΔO_i, any significant change j in the planned value of a resource, ΔR_j, and significant change regards to the expected in the environmental variable h, ΔE_h, which may produce significant changes to the specification of an order or a resource. To infer these changes, propagation functions are used, which can be defined as in equation (2):

$$\Delta O_i = \mathrm{f}(\Delta E_h, \text{ for all } h),$$
$$\Delta R_j = \mathrm{f}(\Delta E_h, \text{ for all } h), \tag{2}$$
$$\text{where: } \Delta E_h = \text{change of the environment variable h}$$

A propagation function can be defined as follow:

- *Based on data*: through relevant data fault propagation patterns are detected and inference rules to specify cause-effect relationships are defined based on them.
- *Based on the structure of the supply process*: a deterministic model is defined to capture and propagate disruptive events based on the structure of the supply process.
- *Based on the availability profile*: a deterministic model is defined to capture and propagate disruptive events based on the availability profile of a resource.
- *Based on the structure of the supply process and probabilistic data*: a probabilistic model is defined to capture and propagate disruptive events based on the structure of the supply process and statistical data updated periodically.

3. Proposed SCEM System

This chapter presents a SCEM system that from the view point of its automation level it can be classified as autonomous corrective system, and from the view point of its monitoring strategy can be classified as order, resource and environment focused.

Figure 1 graphically represents a model of main components of a system proposed for management a SC. The SCEM system is composed by the Control Subsystem, the Monitoring Subsystem, and the Feasibility Management Subsystem.

In this architecture, the SCEM system receives from the Planning System a *schedule* and notifies it if an *exception* has occurred; from the Execution System receives *execution data* and send it a *solution* (repaired schedule) if it is necessary to mitigate the effect of a disruptive event.

Following, the three main components of the SCEM system, Control Subsystem, Monitoring Subsystem, and Feasibility Management Subsystem are described.

3.1 Control subsystem

This subsystem, graphically represented in Figure 2, is responsible for providing the functionality to control a schedule. Each member (node) of a SC has a Control Subsystem responsible for requesting monitoring function (*Request schedule Monitoring)* to the Monitoring Subsystem providing the access to updated data from the Execution Systems. It also interacts with the Feasibility Management Subsystem requesting the feasibility

verification and repairing (*Request Feasibility Schedule Check*) when a disruptive even is detected and engaging in collaborative repair (*Request Collaboration to other Control System*) when requested. It is responsible for send solution to the Execution System received from the Feasibility Management Subsystem and notifying exception to the Planning System for re-planning.

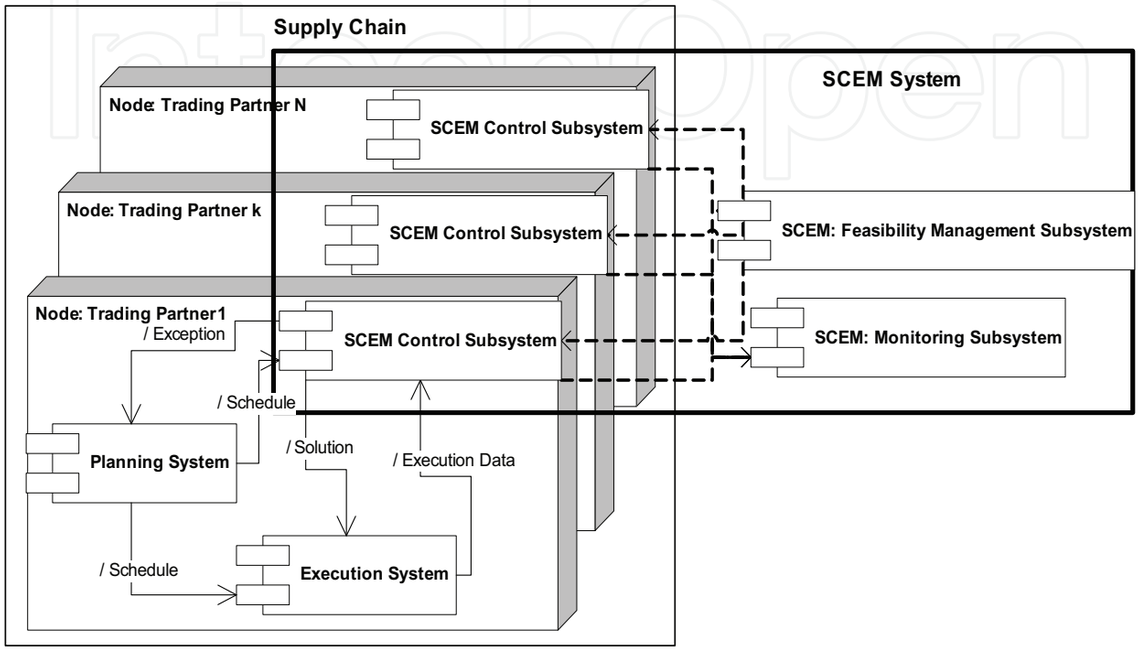


Fig. 1. Model of main components of a SC management system

3.2 Feasibility management subsystem

This Subsystem is responsible of providing functionalities for verifying the feasibility of a schedule when a disruptive event has occurred, and to repair a disrupted schedule requesting the collaboration of other Control Subsystems if it is necessary. To perform these functionalities appropriate decision making models, such as mathematical programming, heuristic programming, etc., are required.

The functional model of the Feasibility Management Subsystem is graphically represented in Figure 3. This Subsystem receives the disruptive event notification, sent by the Control Subsystem, and analyzes the impact of it on a schedule. If the feasibility is damaged, it searches for strategies to repair the schedule.

Sometimes, in order to restore feasibility in a damaged schedule it is necessary to propagate changes towards either customer or providers orders. These changes are feasible only if the customer's and provider's schedules are analyzed together with the damaged schedule, to ensure synchronicity and execution feasibility. That is, the solution to a schedule disruption, if found, must consider all schedule synchronization and should be executable and expressed in a common representation. To this aim, a request for collaboration is sent to the Control Subsystem. The solution to the schedule disruption only introduces changes within planned buffers.

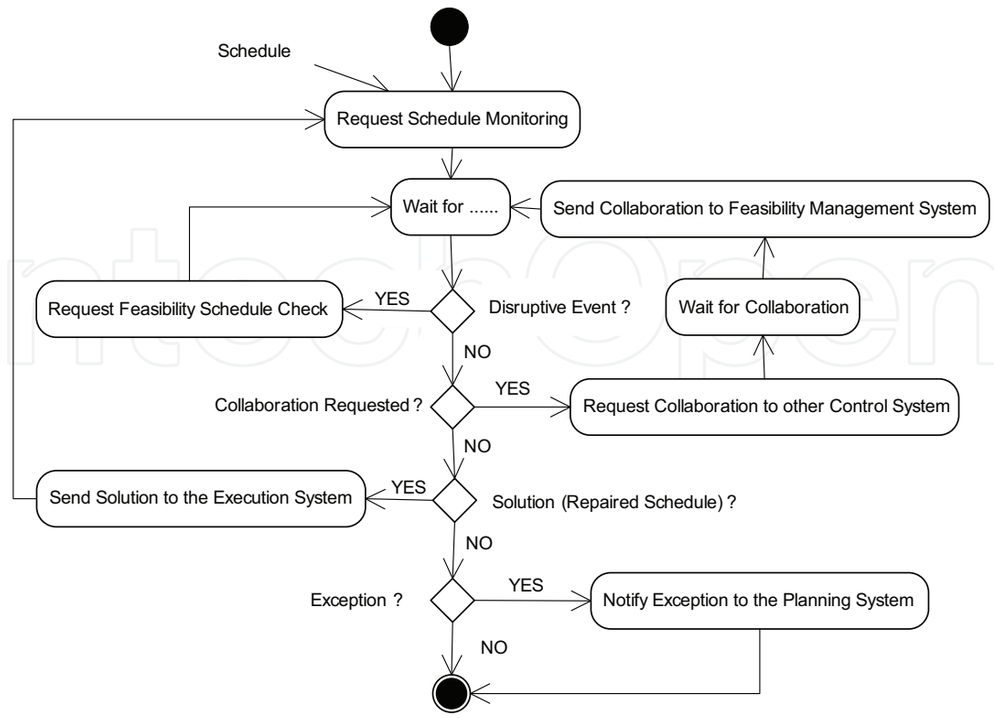


Fig. 2. Functional model of the control subsystem

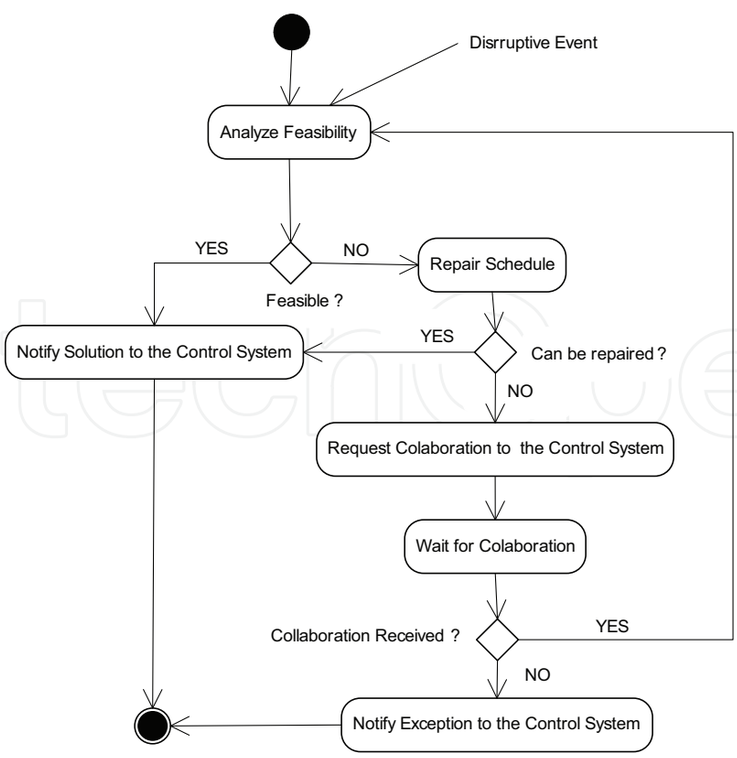


Fig. 3. Functional model of the feasibility manager subsystem

The Feasibility Management Subsystem performs these functions using a model based on a reference model for disruptive event management able to describe on going execution schedules of any kind, systematically capturing the planned buffers on critical resources and orders in a way suitable to perform feasibility analysis and repair processes. The reference model is described following.

3.2.1 Reference model for management of disruptive events

A reference model for management of disruptive event has been presented in (Guarnaschelli et al., 2010). It is relevant to emphasize that an instance of this reference model is a self-contained description of the feasibility of an execution schedule that can be automatically transformed into a Constraint Satisfaction Problem (CSP). This CSP, checks the feasibility of the execution schedule, and also gives the possibility of identifying slacks in order to device a repair mechanism with minimum modifications, as the one presented here. Following the two main modeling views of this reference model are described.

3.2.1.1 The supply process orders

Whenever a disruptive event occurs it is important to track its origin and delimit its propagation, to be able to attenuate its effects or even eliminating them. To do this any ongoing execution schedule, a *Schedule* and the possible sources and propagation paths of a disruptive event, is described as a net of *Resources* and *Supply Process Orders* linking them. This representation is proposed not only because it can show the origin and propagation of a disruptive event, but also allows monitoring and controlling disruptive events and possible exceptions at their origin, communicating disruptions (repair solutions) and exceptions to the proper receptor. The receptor is the Control Subsystem, which has control of the resources and involved supply process orders. Using this representation the disruption propagation and impact can be assessed as different SCs and different business partners in a single SC are represented as supply process orders and resources having different Control Subsystems.

Figure 4 presents an UML class diagram representation of a general supply process. Linked supply processes define a net of resources and supply process orders. In the class diagram, a supply process is defined, through a *SupplyProcessOrder*, which is composed by a set of *DimensionRequirements* imposed to every *FeasibilityDimension* of the resources assigned for the execution of the supply process order. When two supply process orders belong to different business partners, or even to different SCs their relation is captured by the association class *RelatedSPO*, which implies relationships between the two supply process orders, such as same *orderQuantity*, same timing, etc.

Some supply process orders, can be cancelled in favor of the feasibility of execution of other supply process orders, and there can be special supply process orders called spare, that are only executed in case of emergency, for example an supply process order using a 3PL (third party logistics provider).

In a typical SC, a disruptive event can cause different kind of losses, amongst them service level diminishment of the node causing the exception and in general for the whole SC. But it might not affect the totality of the ongoing execution schedule of the related nodes, but instead a set O of *Supply Process Orders* (SPO). Every $SPO \in O$ is related to a set of resources required for its fulfillment (*assignedResources* set) belonging to any of the affected nodes. Whatever the disruptive event, if the information required is on hand, it is possible to trace its origin to the unavailability of one of the related resources or to a change in one supply process order.

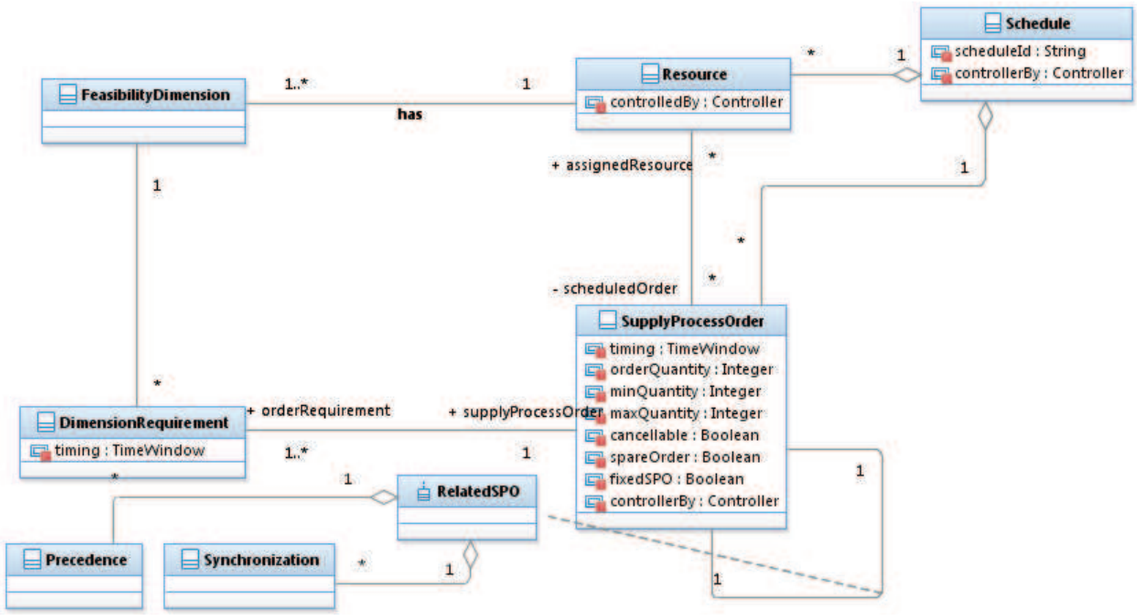


Fig. 4. UML class diagram of a general supply process order

3.2.1.2 The resources

In order to assess the availability of a resource, the feasibility of its schedule and to evaluate the effects of disruptive events, it is necessary to describe the resources. A possible attempt is to classify resources by types as in (Hoffmann et al., 1999), but this has the drawback that resources in a SC can be quite diverse, a more generic and extensible characterization is needed.

As the purpose is to model the availability of resources and the feasibility of its scheduled supply process orders, the characterization of resources by introducing the concept of feasibility dimensions is proposed (Figure 5).

A *FeasibilityDimension* is a characteristic of a resource describing its capability to fulfill a requirement from a supply process order. The availability of the resource is therefore conditioned by them and every requirement for the resource should be expressible in terms of its feasibility dimensions.

Two types of feasibility dimensions are defined, *CapacitatedDimension* and *StateBased Dimension*. Each one of them has an *AvailabilityProfile* that describes its intrinsic and planned availability in a given *Horizon*.

Every *SupplyProcessOrder* has a set of requirements over the resources assigned for its fulfillment. These requirements should be expressed according to the availability of each resource, which is expressed in feasibility dimensions. The concept of *DimensionRequirement* to define all the possible uses of resources is introduced, in correspondence with each type of feasibility dimension. Therefore there are two types of requirements *StateBased Requirement* and *CapacityRequirement*.

3.3 Monitoring subsystem

This Subsystem is responsible of providing functionalities for monitoring a schedule. The execution of a schedule implies performing the operations defined in the supply process

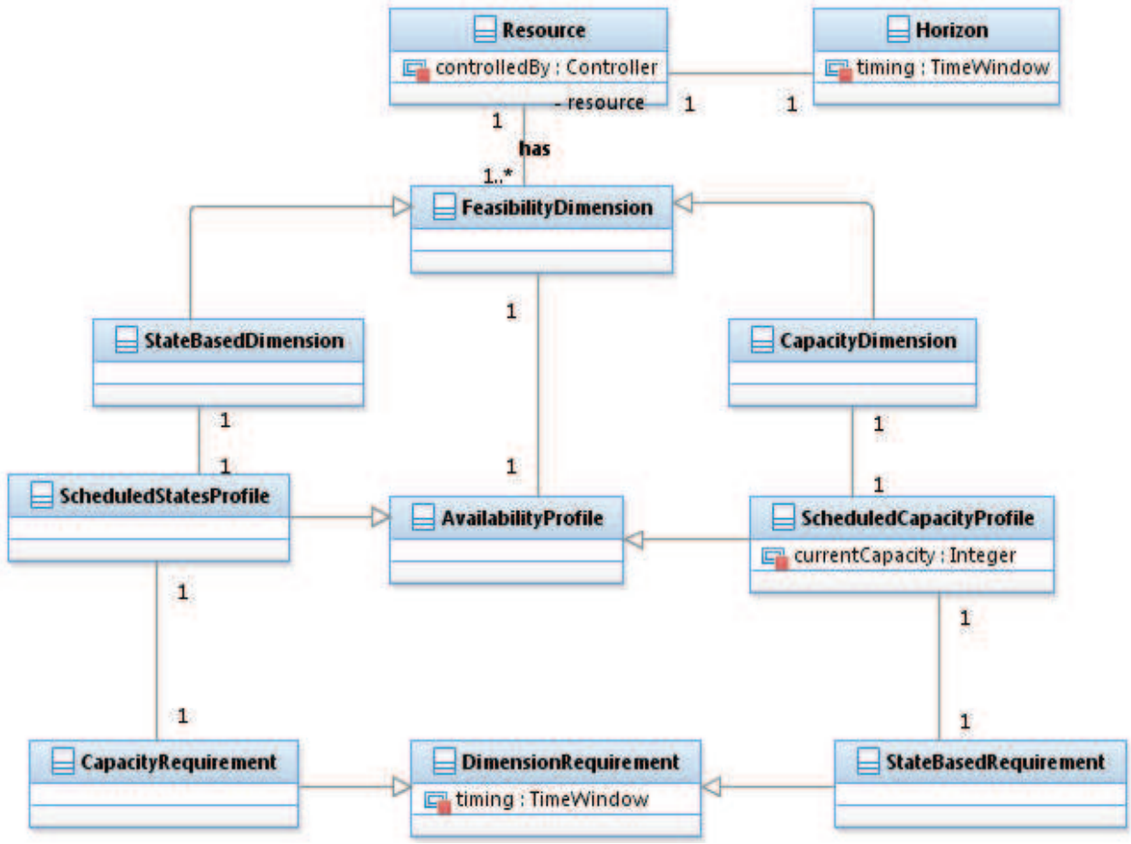


Fig. 5. Feasibility dimension of a resource

each order represents and is feasible if both the order requirements and the availability of the resources take their planned values. The monitoring function implies detecting relevant disruptive events in real time (Knickle & Kemmeler, 2002). To perform this functionality four main monitoring activities have to be carried out during the execution of a schedule, which are described following.

Monitoring changes on expected future availability of a resource: The objective is to capture significant changes of the planed value of the future availability of a resource.

Monitoring order progress: The objective is to monitor on going execution orders to proactively predict if a disruptive event affects the order expected completion. This implies to capture significant changes on any variable measuring the order progress or having a predictive relationship with this progress. Typically, they are variables in the execution environment that are used as predictors of potential disruptions.

Monitoring current status of resource feasibility: The objective is to capture significant changes on the current value of any attribute of a resource that is critical to grant its feasibility.

Monitoring current status of resource feasibility: The objective is to capture significant changes on the current value of any attribute of a resource that is critical to grant its feasibility.

Monitoring order specification changes: The objective is to capture independent changes of the order specification values, i.e., start time, quantity or end time. By independent, we mean original modifications to the order specification, not as derived consequence of adjusting the order in response to other disruptive events.

The functional model of the Monitoring Subsystem is graphically represented in Figure 6. The Monitoring Subsystem has the ability of systematically generating the structure for capturing changes that can take place into the enterprise or outside the enterprise and may affect a supply process order or a resource. Using a cause-effect model propagates these changes to infer advancement or delay in the start or end date of the order, changes in the amount specified by the order, change into the availability of a resource, or change into the current level of a resource regards to its planned value, and analyzes if these changes can produce a disruptive event. The cause-effect model is developed based on the supply process structure and statistical data, and can be modeled using Bayesian Network, Petri Net, decision tree, etc. In this way, the Monitoring Subsystem can proactively notify the Control Subsystem a disruptive event affecting an order or a resource has occurred.

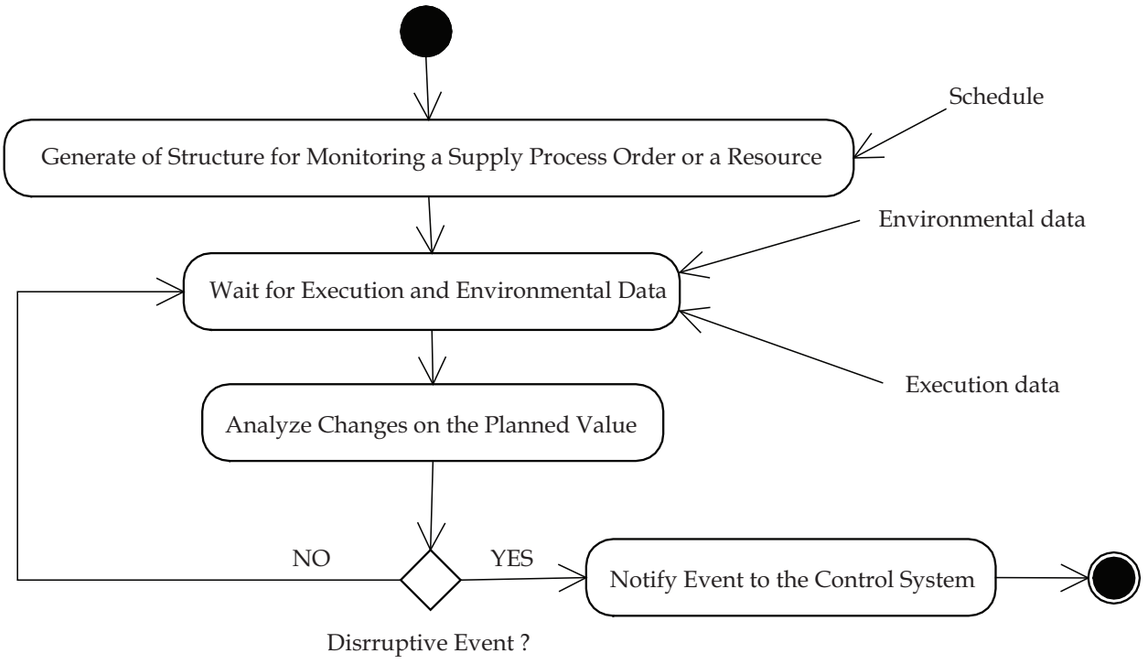


Fig. 6. Functional model of the monitoring subsystem

3.3.1 Reference model for monitoring orders and resources

A reference model for monitoring orders has been presented in (Fernández et al., 2010). In this chapter the reference model is extended for monitoring orders and resources. An instance of this reference model is a self-contained description of the monitoring structure of an order or a resource, which can be automatically transformed into a particular cause-effect model. The UML class diagram in Figure 7 presents the monitoring reference model which has a monitoring network structure based on a *cause_effect* relationship among *Variables*. These variables represent *AttributeVariable* (resource or order specifications) or *Environment Variable* affecting a resource or a supply process order specification.

The monitoring structure has a set of milestones. Each *Milestone* defines a point where a set of variables will be observed. Each *Variable* of the monitoring structure has one *State* that can be: *ObservedState* or *EstimatedState*. When the state is *ObservedState*, the *Variable* is observed and its value is given. When the state is *EstimatedState*, the *Variable* value is estimated from

the value of other variables using the *cause_effect* relationship network. To perform this task, the *MonitoringStructure is_analysed_by* a *MonitoringStructureAnalyzer*. A variable has an observation policy. An *ObservationPolicy* defines the mode, the recurrence and the updating time of the observed variable.

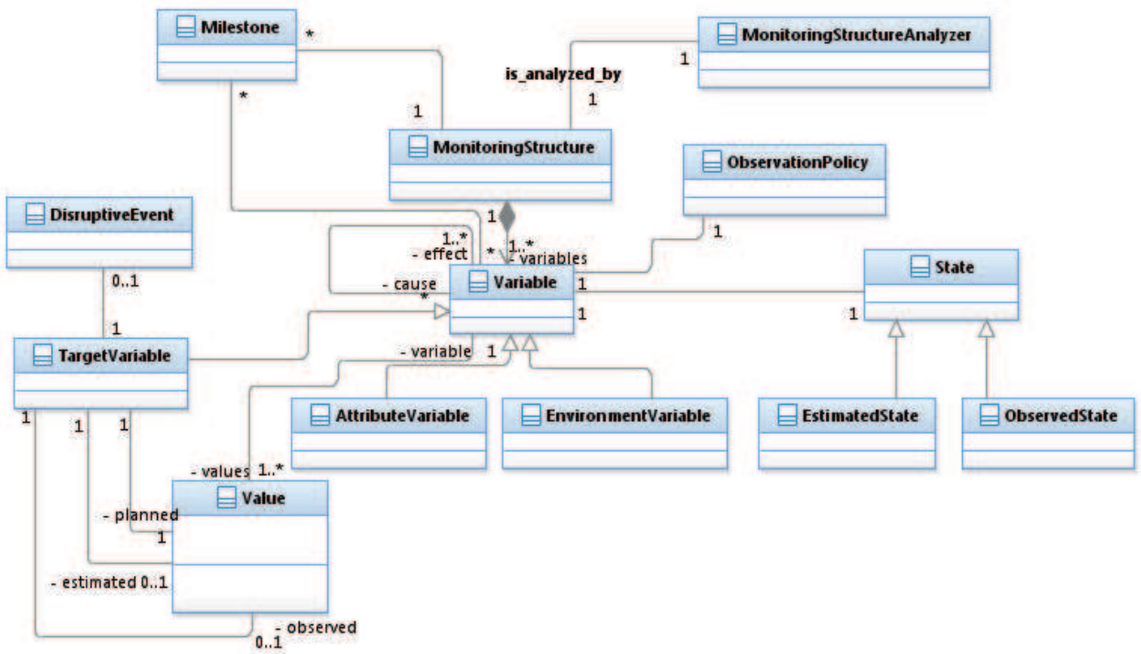


Fig. 7. Reference model for monitoring orders and resources

The *TargetVariable* has a planned *Value* that is an order specification (for example, amount, end time, etc.) or a resource parameter (for example, geographical positions planned, production rate planned of the machine, etc.). Also, the target variable must have an estimated *Value* which is defined by a *MonitoringStructureAnalyzer* or an observed *Value* which is a given value of a variable in the monitoring structure.

The *TargetVariable* is used to evaluate if a disruptive event can occur. This is done by evaluating conditions between the estimated/observed value and the planned one. When the disruption condition is verified, a *DisruptiveEvent* is reported.

3.3.1.1 Monitoring changes on expected future availability of a resource

Figure 8 presents the UML class diagram associated with this activity. Every resource has a scheduling horizon during which it will be monitored. Each *AvailabilityProfile has_assigned* a *MonitoringStructure.*

The *ScheduledCapacityProfile* is defined by an ordered set of *AvailableCapacityItem*. Each *AvailableCapacityItem* has two *TimeMilestone* (*itemStartTime* and *scheduleStart* attributes) that define a time period where the capacity bounds will be observed to evaluate modifications of its planned values.

The monitoring structure has a set of *TargetVariable*. This is, for each *AvailableCapacityItem* there are two target variables (one for each capacity bound).

The Monitor is responsible for observing the capacity bounds at each milestone associated with a *AvailableCapacityItem.* It gets the observed value of each capacity bound and inserts

them to the *MonitoringStructure*. Each target variable (*minTargetVariable* and *maxTargetVariable*) has a planned *Value* that is the planned maximum or minimum capacity bound. The Monitor uses each target variable and comparing its planned and observed values to evaluate a possible *DisruptiveEvent*.

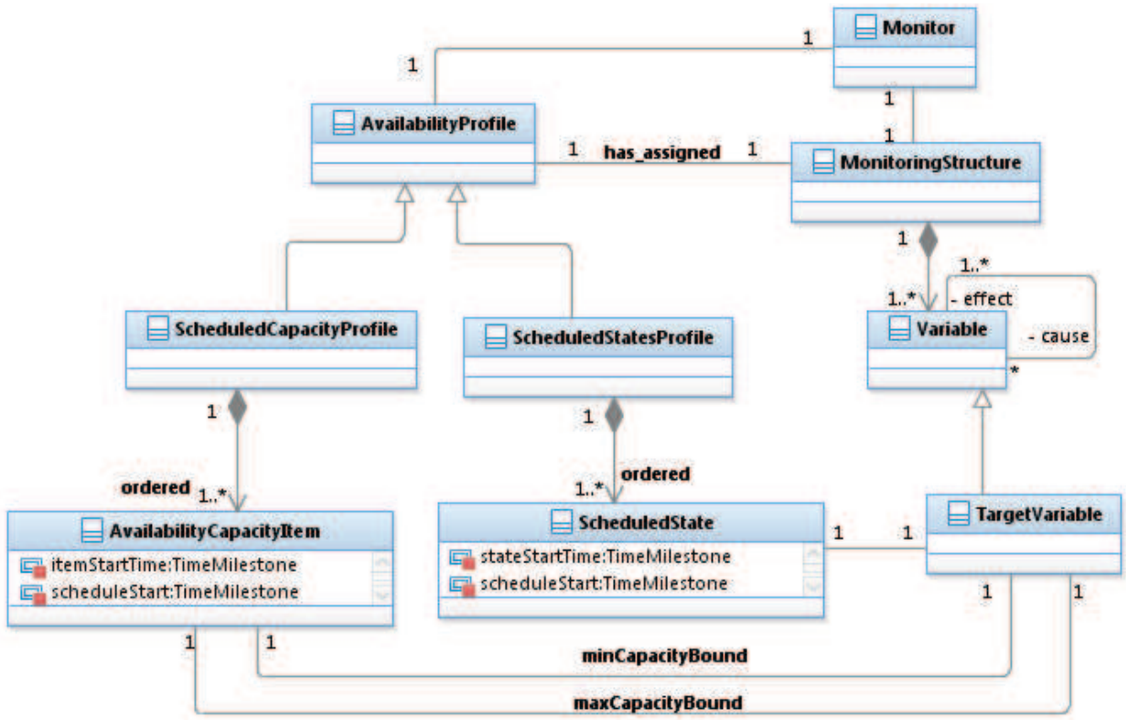


Fig. 8. UML class diagram for the Monitoring future availability of a resource

The *ScheduledStatesProfile* is defined by an ordered set of states, *ScheduledState*. Each *ScheduledState* has two *TimeMilestone* (*stateStartTime* and *scheduleStart* attributes).
The monitoring structure associated with *ScheduledStateProfile* has a target variable for each *ScheduledState*. The Monitor is responsible for observing the specified state at each milestone. It gets the observed value of each *ScheduledState* and inserts it to the *MonitoringStructure*. The target variable has a planned *Value* that is a parameter of the state. It uses the target variable and comparing its planned and observed values to evaluate a possible *DisruptiveEvent*.
For both dimensions, once the schedule start milestone is activated, the monitor will capture any change in the planned availability values and when the change is assessed as significant, according to a threshold, the disruptive event is concluded.

3.3.1.2 Monitoring order progress

Figure 9 presents the UML class diagram associated with this activity. The monitoring structure associated with this activity in general will depend on the type of process since complex cause-effect relationship among variables may be introduced to improve the predictive capabilities of a disruptive event.
Each *SupplyProcessOrder* that is part of a schedule has a *SupplyProcess*. Each *SupplyProcess has_a_set_of* milestones defining its *MonitoringStructure*. A *Milestone* can be a *TimeMilestone* (absolute time or related to another milestone) or a *StateMilestone* (state to be reached by the

supply process). Each *Milestone* has a set of *Variables* associated that allows representing *Environment* variables affecting the supply process or resources *Attribute*s.

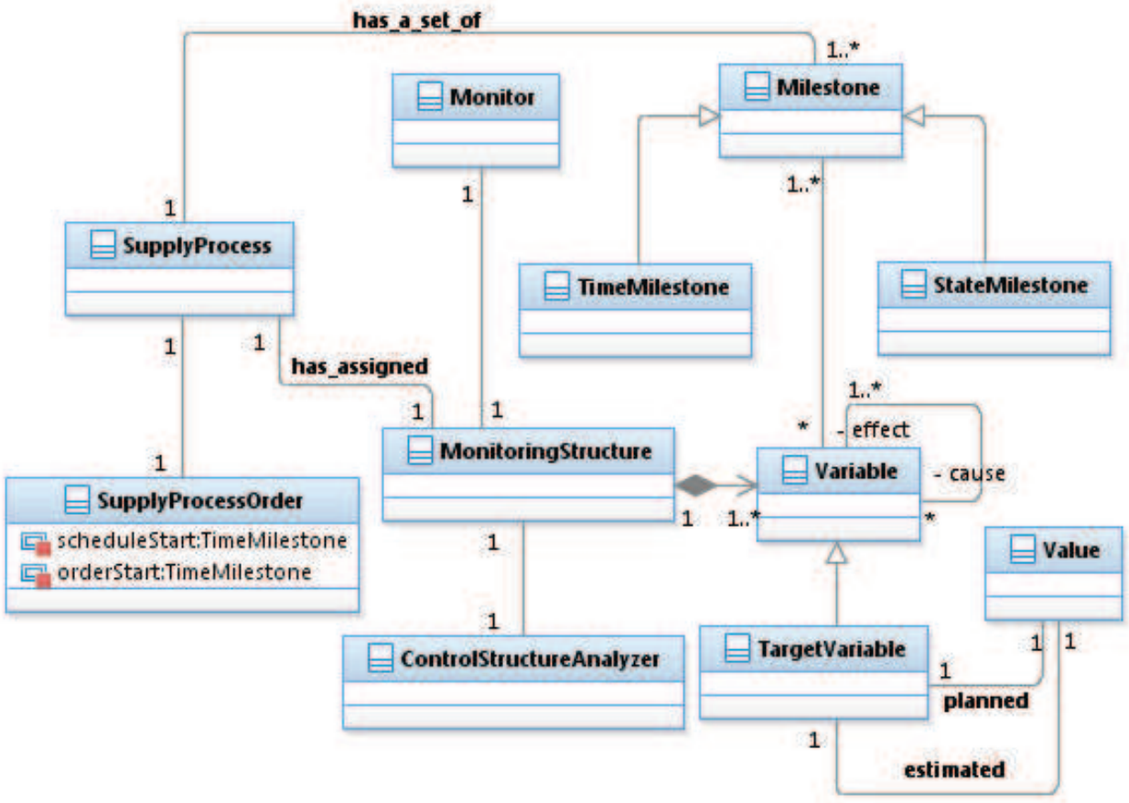


Fig. 9. UML class diagram of the *Monitoring order progress*

Each *SupplyProcess has_assigned* a *MonitoringStructure* based on a *cause_effect* relationship among *Variables* associated with all milestones, which allows predicting if a disruptive event at the last milestone can occur.

Each *Variable* associated with a *Milestone* has one *State* that can be: *ObservedState* or *EstimatedState*. The state of a variable can be changed on the same milestone. After the value of a variable whose state was estimated is known (evidence), the variable changes to observed state. The branch of the monitoring structure (cause_effect relationship sub-net) that predicted its value is no longer necessary and can be eliminated.

The *Monitor* is responsible for observing the variables at each *Milestone*. It starts with the initial milestone, gets the value of each observed *Variable* and inserts them to the *MonitoringStructure*. The *MonitoringStructureAnalyzer* using the *MonitoringStructure* (cause_effect relations net) evaluates the impact of these variable values on current and next milestones until the last milestone. Particularly, it defines an estimated *Value* for the *TargetVariable*. The *TargetVariable* has a planned Value that is an order parameter (for example, amount planned, end time planned, etc.). Following, to predict if a disruptive event can occur, the *Monitor* uses the *TargetVariable* comparing its planned and estimated values. Based on a decision criterion, it predicts if a disruptive event can occur, if so, it reports the *DisruptiveEvent* and the monitoring process ends; if not, the monitoring process follows. To this aim, the *Monitor* defines the next *Milestone* where the variables have to be

observed. The monitoring structure is initially defined for each supply process, but it is dynamically explored each time a milestone is reached. That is, the *Monitor*, depending on the results generated by the *MonitoringStructureAnalyzer*, can extend its monitoring strategy to another milestone including other observed variables, eliminating those that are not necessary or exploring different branches of the control structure.
Unless a specific structure is designed for the process, a default structure is generated having two milestones: supply process order start and supply process order end, two target variables: corresponding to the order end time planned and order amount planned, and one observed variable indicating a measure of the order progress. This measure is used to infer estimated values for the target variables and anticipate a disruptive event.

3.3.1.3 Monitoring current status of resource feasibility

Figure 10 presents the UML class diagram associated with this activity. The monitoring structure associated with this activity defines a target variable for each feasibility dimension associated with a resource. For each *AvailableCapacityItem* and *ScheduledState* there is a target variable which has a planned value that represents an expected value of the corresponding feasibility dimension of the resource. The planned value is calculated through the projected profile for each feasibility dimension. This is, for each supply process order there is a set of requirements over the resources assigned for its fulfillment. These requirements are expressed according to the resource's availability, which is expressed in feasibility dimensions.

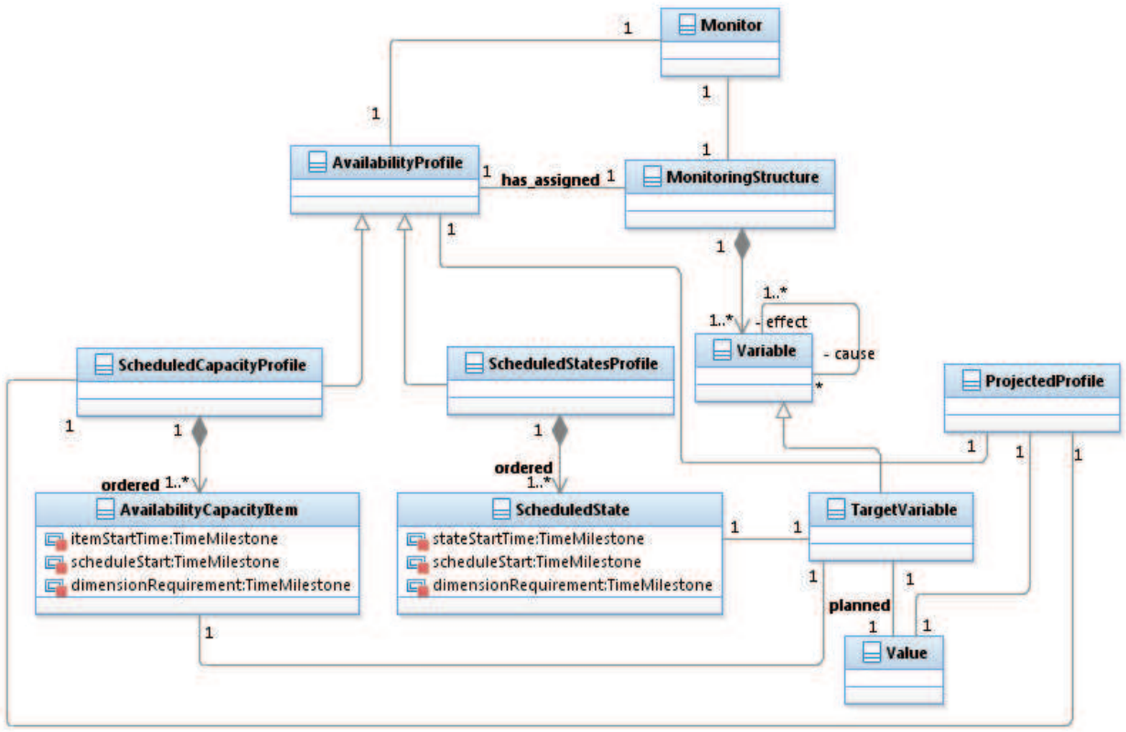


Fig. 10. UML class diagram of the Monitoring current status of resource feasibility

The *Monitor* has a milestone for each dimension requirement where the current value is observed and compared against the planned value to evaluate the occurrence of a disruptive event in that dimension.

3.3.1.4 Monitoring order specification changes

Figure 11 presents the UML class diagram associated with this activity. The monitoring structure associated with this activity defines three *TargetVariables* for each *SupplyProcess Order*. Each Variable has a planned value and an observed value. The planned value corresponds to an order specification (start time, end time or quantity). By default, the Monitor has two milestones to evaluate if a disruptive event may occur in an order. These are: schedule start, order start. Once the schedule start milestone is activated, the monitor will capture any change in the planned values for the order target variables (i.e., changes in the order specification) and when the change is assessed as significant, according to a threshold, the disruptive event is concluded. When the order start milestone is activated, the actual start time will be observed and compared with the planned value to evaluate a possible disruptive event. After this milestone, this activity finishes.
Following the principles of the model driven architecture (Mellor et al., 2004), through a model-to-model transformation, from an instance of the monitoring reference model, a monitoring model can be automatically derived.

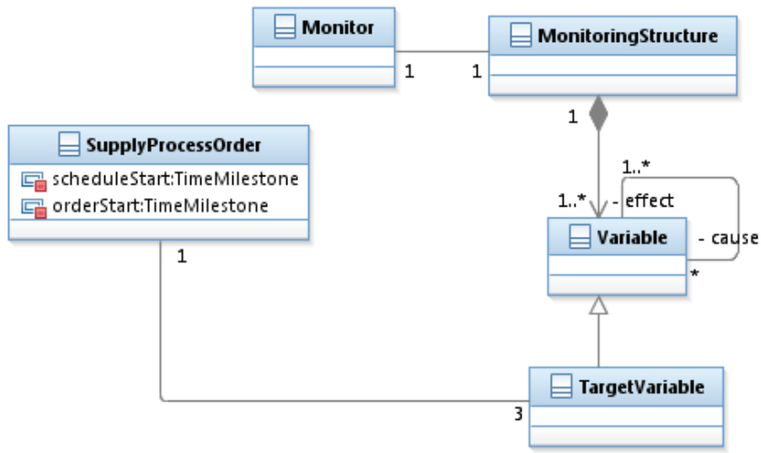


Fig. 11. UML class diagram of the Monitoring order specification changes

4. Case study: A commodity chemical supply chain

As illustrative example a case study presented in (Guarnaschelli et al., 2010) is used. In this supply chain Urea is produced in the factory located at Bahía Blanca, Argentina, warehoused in the factory warehouse, *FWBahiaBlanca* and distributed to three distribution centers *Urea-DCSanLorenzo* at San Lorenzo, Argentina; *Urea-DCUruguay*, at Montevideo, Uruguay; and *Urea-DCBrasil*, at Rio Grande, Brazil. The distribution centers are sourced by means of dedicated ships through fluvial and maritime routes. Table 1 presents the average trip times in hours.

	Urea-DCSanLorenzo	*Urea-DCUruguay*	*Urea-DCBrasil*
FWBahiaBlanca	96	144	168
Urea-DCUruguay	-	-	60

Table 1. Average trip times in hours

Distribution schedules for a horizon of 33 days are generated by a Distribution Resource Planning system. Product availability in *FWBahiaBlanca* is considered to be unlimited, this means that stock, demand and supply is managed for each distribution center attending constraints regarding to: Ships routes and availability, loading dock availability at factory warehouse, and inventory size and safety stocks constraints at each distribution center.
The schedule defines the replenishments to each distribution center, which imply coordination and timing for the resources implied by them. Using the reference model (Section 3.2.1) replenishments are modeled as transfers from *FWBahiaBlanca* to each distribution center, using the corresponding ship, loading dock and inventory resource at distribution center, like the supply process *transfer-Urea-BahiaBlanca-DCBrasil-26* depicted in Figure 12. This figure also shows how the implied resources are modeled.
Ships are coordinated with regards to its capacity and geographical position. A *stateBasedDimension* defines every geographical position along the scheduling horizon of every ship resource. The successive *stateBasedDimension* along the scheduling horizon defines the *ScheduledStatesProfile* for the corresponding ship resource.
Inventory resources modeled with a single *CapacityDimension*, only have a maximum capacity constraint (Table 2) captured by *availableCapacityItems* in the *scheduledCapacityProfile*.

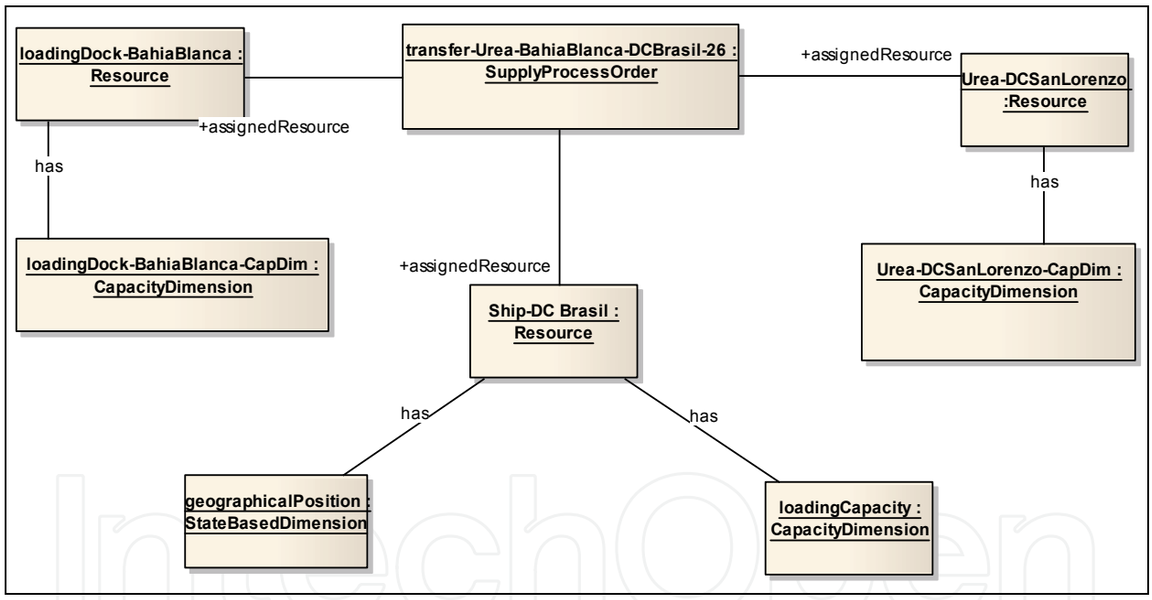


Fig. 12. A transfer-Urea-BahiaBlanca-DCBrasil-26 supply process order

The loading dock acts as a renewable resource, requirements it attends are of type *Renewable*.
Each distribution center has a list of Supply Process Orders (SPO) (transfer orders) to serve, which for this example are set together as a daily shipment for each of the 33 days. Each SPO imposes a *CapacityRequirement* on the corresponding inventory resource.

	Urea-DCSanLorenzo	*Urea-DCUruguay*	*Urea-DCBrasil*
Maximum capacity	30,000	20,000	20,000

Table 2. Maximum capacity of distribution centers in tons

In this schedule, the shipments from each distribution center have the following scheduling/ rescheduling policy: Each order has a time window for its dispatch, if it was originally scheduled for day d, in case of a repair procedure it cannot be dispatched outside the week that contains day d. Some orders allow quantity modifications but they cannot exceed 10% of the original value. Table 3 presents minimum and maximum shipment size in tons.

	Urea-DCSanLorenzo	*Urea-DCUruguay*	*Urea-DCBrasil*
minimum	953	393	705
maximum	2150	696	1147

Table 3. Minimum and maximum shipment size (*orderQuantity*) in tons

The replenishments to the distribution centers, that is transfer orders, are scheduled with a specific timing in the scheduling horizon, but, in case a of a repair procedure, their timing specification can be changed as long as resources capacities (inventories and ship capacity) and states (geographical position of ships) allow these changes. Transfer quantities are only limited by ships capacities (all of them have a storage capacity of 17000 tons). At Bahía Blanca ships are loaded at a constant rate of 1000 tons/hour. And in distribution centers ships are downloaded at a constant rate of 250 tons/hour in San Lorenzo and Uruguay, and at 425 tons/hour in Brasil.
In exceptional situations the supply from Bahia Blanca to Brasil can be done by a third party logistics provider that by contract provides a transportation capacity of 17000 tons with a delivery time of 7 days. This optional process is modeled as a *spareOrder* (also cancelable) that requires for its execution resources *loadingDock-BahiaBlanca* to load Urea and *Urea-DCBrasil* to download Urea.

4.1 Predicting and detecting disruptive events

The monitoring function performs four main activities during the execution of a schedule (Section 3.3). These are: monitoring order specification changes, monitoring current status of resource feasibility, monitoring order progress and monitoring changes in the expected future availability of a resource. The monitoring structure is generated using the reference model for monitoring orders and resources (Section 3.3.1).
In this supply process, the navigation conditions of the ship can be unfavorable due to the weather conditions (storms, winds, etc.). These unfavorable weather conditions are more frequent in the winter season and can produce a delay in the ship arrives to the port. The delay can be increased if in the arrival port or in the intermediate ports there are unfavorable weather conditions or the port is congested. This prevent to carry out unload operations. The *Ship-DCBrasil* can carry orders to ports on Uruguay and Brasil. I.e., it is not a dedicated ship to an order. Therefore, this ship that has to carry Urea an order requires from Bahía Blanca to Rio Grande, may need to go through an intermediate port to meet the requirements of orders of Montevideo. It can be see in Table 1 the ship is in transit 144 hours from Bahia Blanca to Montevideo and 60 hours from Montevideo to Rio Grande. The *MonitoringStructure* (Section 3.3.1.2) for monitoring the progress of this maritime transport order is graphically represented in Figure 13. The total navigation time of the ship will be of 204 hours. The milestones set contains:
depart_of_the_BahiaBlanca_port:StateMilestone, arrival_to_intermediate_position_1:StateMilestone, arrival_to_intermediate_position_2:StateMilestone, arrival_to_Montevideo_port:StateMilestone, arrival_to_intermediate_position_3:StateMilestone, arrival_to_RioGrande_port:StateMilestone.

In Figure 14 a graphic representation of a Bayesian network of this *MonitoringStructure* is graphically represented. It is composed by discrete nodes and continuous nodes. The discrete nodes (the states are represented in braces) are the following: *season:DiscreteNode* {winter, non_winter}, *navigation_condition:Discrete Node* {favorable, neutral, unfavorable} and *weather_conditions_at_port:DiscreteNode* {favorable, unfavorable}, *delay_in_departure:Discrete Node* {[0-0.5),[0.5-2),[2-6),[6-24),[24-48),[48-inf)} and has a Gamma distribution. The continuous nodes are the following: *delay_in_transit:ContinuousNode*, *delay_in_position :ContinuousNode*, *delay_in_intermediate_port:ContinuousNode* and *delay_in_arrival:Continuous Node*. The node function is *estimated_delay_in_arrival:FunctionNode*.
The decision criteria used by the Comparator establishes that the ship is delayed when its probability is greater than a threshold. In this example, the threshold has been defined equal to 24 hours.

4.2 Disruptive event in the case study

To illustrate the capabilities of the Feasibility Manager Subsystem in the case study a scenario generated due to a disruptive event notified by the Monitoring Subsystem is considered. The disruptive event is an unexpected increase in demand at *Urea-DC-Uruguay* has occurred.
The supply chain of this example shares the Urea market with another provider and its supply chain. At Uruguay the distribution center of the competitor temporarily runs out of stock, this obligates its clients to supply from *Urea-DCUruguay*. As a result shipments scheduled from day 10 to 19 are duplicated and now their *orderQuantity* (without possibility of negotiating order quantities) are incremented as shown in Table 4:

Supply Process Order	*orderQuantity*	*new orderQuantity*
shipment-Urea-DCUruguay-10	668	1500
shipment-Urea-DCUruguay-11	589	1500
shipment-Urea-DCUruguay-12	481	1500
shipment-Urea-DCUruguay-13	628	1500
shipment-Urea-DCUruguay-14	693	1500
shipment-Urea-DCUruguay-15	656	1000
shipment-Urea-DCUruguay-16	502	1000
shipment-Urea-DCUruguay-17	683	1000
shipment-Urea-DCUruguay-18	649	1000
shipment-Urea-DCUruguay-19	509	1000

Table 4. Unexpected changes in *orderQuantity* of SPOs in tons

Simulating the effects of this event on *Urea-DCUruguay* inventory (*CapacityDimension : Urea-DCUruguay-capDim*), an important infeasibility clearly appears as seen in Figure 15, looking at the curve named "Exception".
The Feasibility Manager Subsystem returns the following results: A set of 10 SPOs are modified (within planned buffers) in order to restore feasibility (Table 4). In Figure 15 is visible how the solution of the mechanism is closely related to the original schedule. The second Urea transfer is put forward and the other modifications consisted in slightly reducing some orders quantities in order to restore feasibility preserving most of the original schedule.

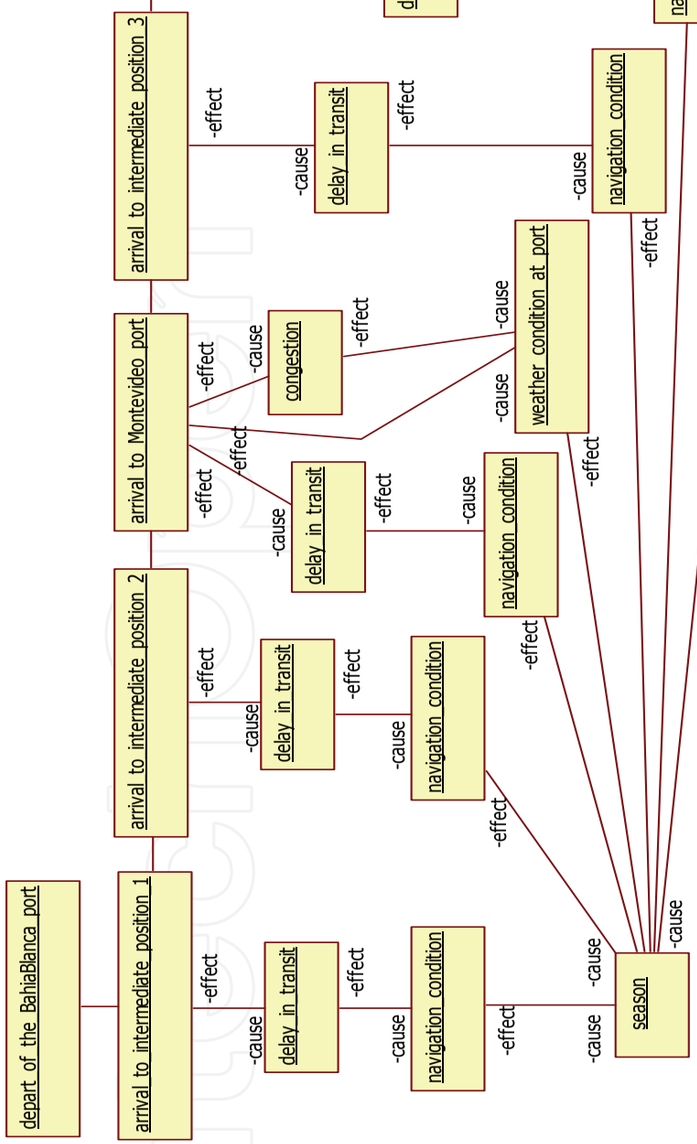


Fig. 13. *MonitoringStructure* of a marine transport order

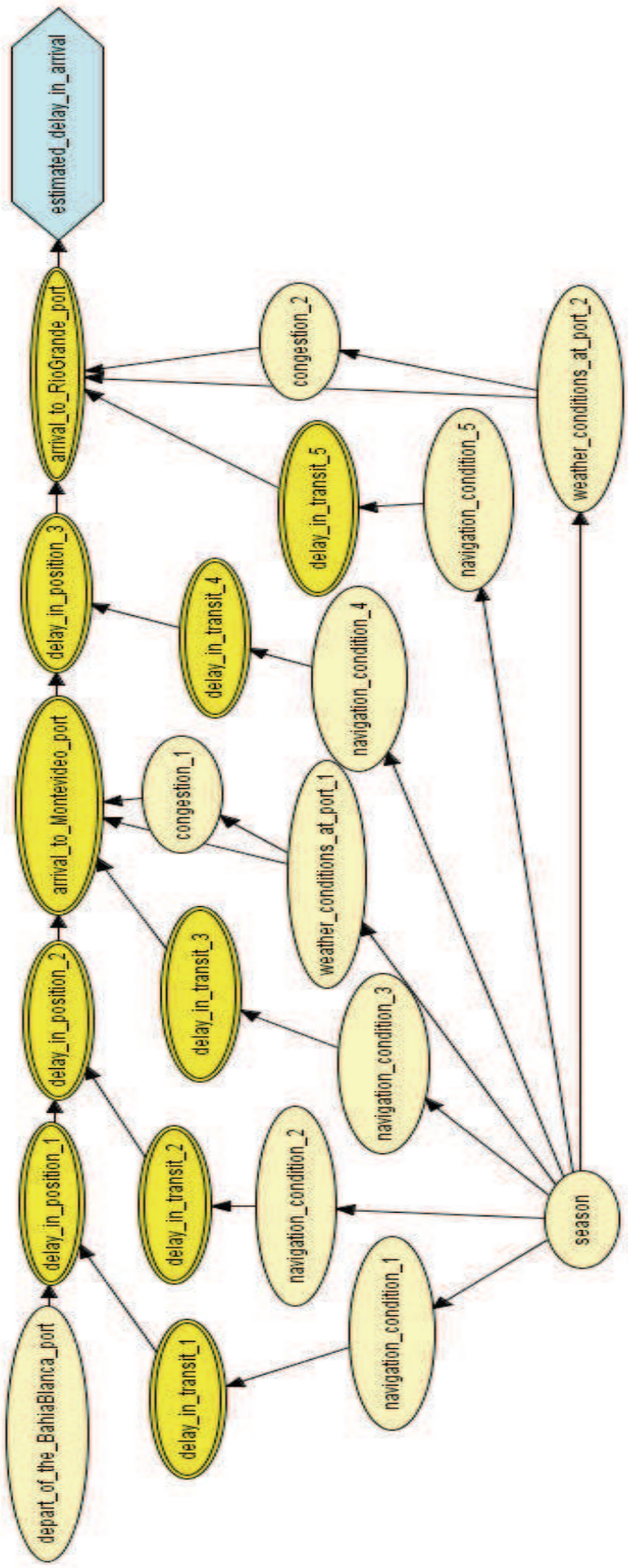


Fig. 14. Monitoring structure based on Bayesian Network

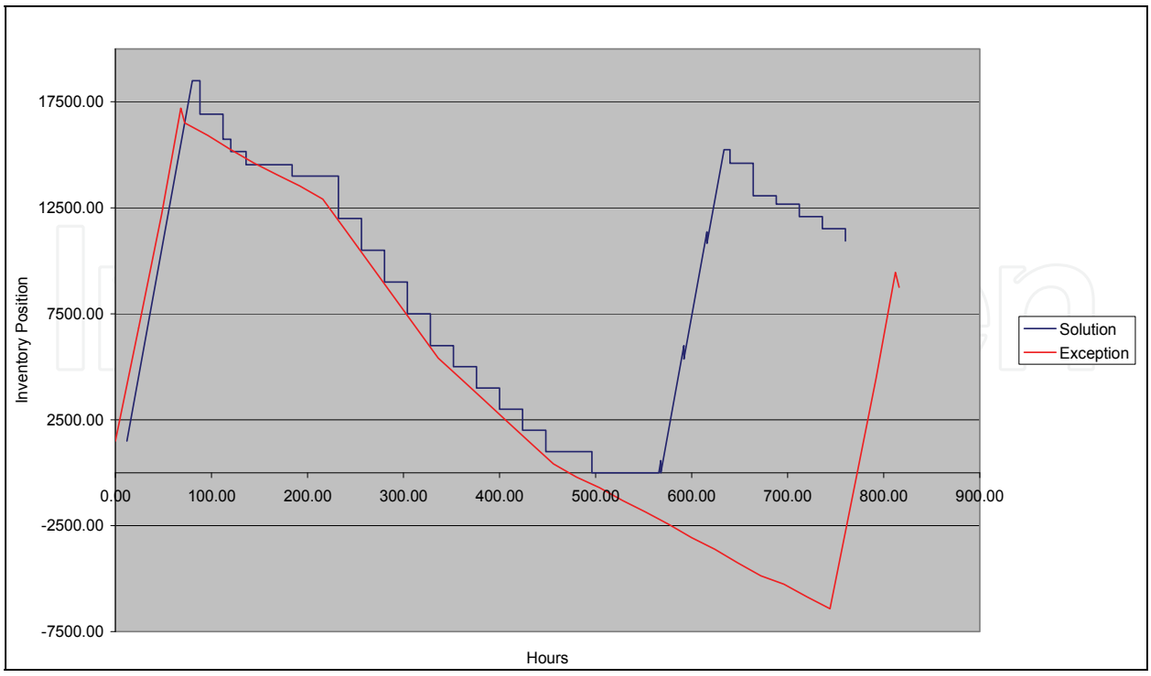


Fig. 15. *Urea-DCUruguay* projected available inventory (exception and solution)

5. SOA-based implementation of the SCEM system

Service oriented computing and architecture is the technology chosen to enact the SCEM business process in (Fernández et al., 2010 and Guarnachelli et al., 2010). It provides a way to create software artifacts that supports requirements on heterogeneity and autonomy that arise on current supply chain management practices. It provides a way to capture business requirements and processes and software artifacts delivered using Service-Oriented Architecture (SOA) (Papazoglou et al., 2007) are tied to them. The collaborative nature of this proposal for SCEM is benefited by adopting SOA technologies as it promotes few coarse grained interactions between service providers and consumers. Additionally a SOA architecture definition is platform independent allowing its implementation on business partners with diverse co-existent technologies.

Using the methodology for SOA development (Arsanjani et al., 2008), to support the functionalities of the SCEM business process three participants have been proposed: Controller, Monitor and Feasibility Manager. Following, the collaborative SCEM business process has been defined (Fernández et al., 2010) (Guarnachelli et al., 2010). The business process defines the necessary collaboration among participants (messages) and the tasks each them has to perform to provide the SCEM functionalities to any SC implementing it. These participants define the components of the SCEM system architecture presented in Section 3.

To identify all the capabilities and services required to enact the SCEM business process, a standard modeling technique (SOMA) (Arsanjani et al., 2008) was followed. The service model has been designed starting with the capabilities each participant in the business process has to provide to enact the collaboration. After that, the services exposing these capabilities were defined.

In this proposal a document-centric approach to support the access to service operations has been adopted, therefore for each operation defined in a service there is an associated

document containing all the information required to provide the service. These documents are specified using the *messageType* artifact from the SOAML specification. (OMG, 2009).
In order to define the messages and its information content in a consistent and complete way, we use the reference model described in Section 3.3.1. This model provides self-contained descriptions of any on-going execution schedule of supply process orders with all the information required to assess its feasibility.

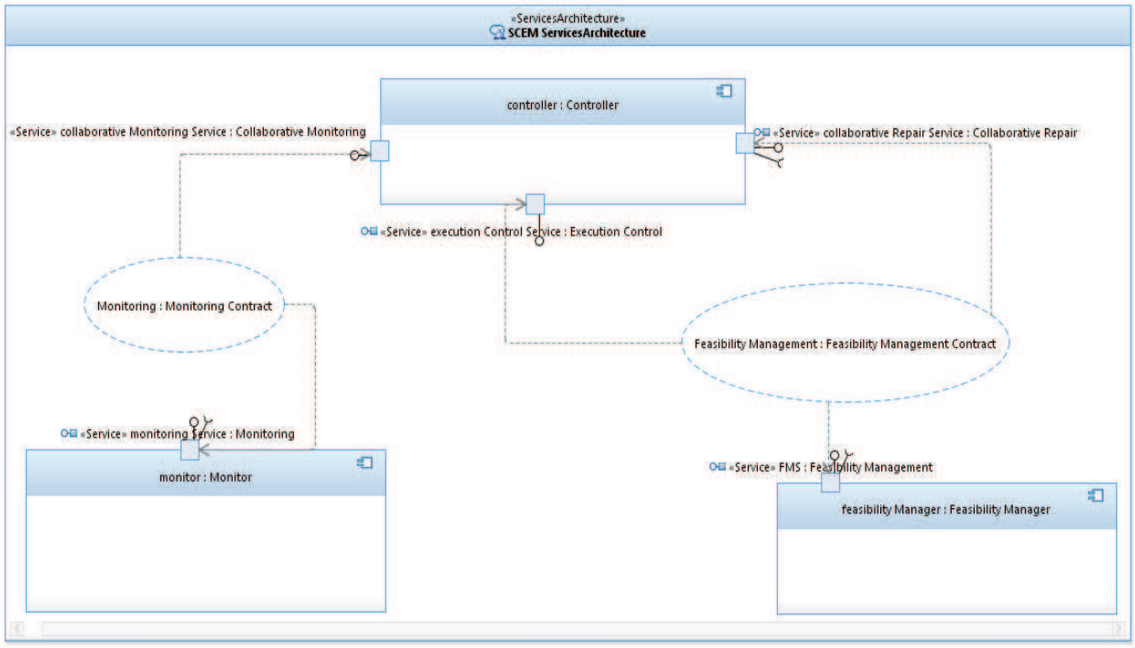


Fig. 16. SCEM service architecture

Fig. 17. The monitoring and feasibility management contracts

The architecture resulting from organizing the participants and services through specific service contracts including the collaboration choreographies is represented in Figure 16.

In order to specify how compatible requests and services are connected by service channels, service contracts Feasibility Management and Monitoring are defined (Figure 17).
These contracts contain all the metadata regarding the description about the SCEM SOA based solution services that is: the purpose and function of their operations, defined in service identification; the messages that need to be exchanged in order to engage the operations, defined in the service interface specification and the data models used to define the structure of the messages, defined in the service data model.
The choreographies required by the collaboration among the participants in these contracts were described using UML sequence diagrams specifying the asynchrony of operation calls, and the logic and sequence in which operations are used.

6. Conclusion

In this chapter a proposal to systematically address the problem of disruptive event management in SC is described. The proposal includes the definition of a SCEM system architecture conceived to provide system support for companies willing to engage in collaboration agreements for controlling the execution of their supply processes. The architecture allows supporting a collaborative execution of the SCEM business process by independent supply chain partners. Because of the complexity of the models required for feasibility check and schedule repair, and the models for monitoring orders and recourse, in the proposed SCEM system architecture, theses functionalities are performed by two centralized subsystems, the Feasibility Management and the Monitoring Subsystems, which are implemented as web services. The aim is to prevent members have to make these complex processes and decision analysis. That is: To evaluate the feasibility of a schedule and to collaboratively repair a disrupted schedule a Constrain Satisfaction Problem (CSP) has to be solved. This requires the use of appropriate CSPsolvers, for example, the IBM ILOG OPL Development Studio (IBM, 2010) has been used to solve the CSP for the scenario of the case study in Section 4.2. To predict if significant changes regards to the expected in the environmental variable may produce significant changes to the specification of an order or a resource, a cause-effect network has to be used. For example, in the case study (section 4.1) a representation of the Bayesian Network has been used, which has been processed using the inference engine of Hugin Expert A/S (Hugin, 2010).
A distributed Control Subsystem implemented by each member, is responsible for providing the functionality to control a schedule requesting monitoring function to the Monitoring Subsystem and feasibility verification and repairing to the Feasibility Management Subsystem when a disruptive even is detected, and engaging in collaborative repair.
The consistency of interoperation between the subsystems is granted in the semantic level with reference models that provide the basis for the definition of the business documents being exchanged among the subsystems. Reference models accomplish the description of the problem information in a very high level of abstraction and therefore being applicable to a wide range of SC processes, from procurement, manufacturing, distribution, and retailing domains. The reference models have the characteristic of providing self-contained descriptions of the information required for the decision making activities involved in the SCEM business process. This feature enables the possibility of automating the generation of decision models expressed in standard representations for decision making tools as mathematical programming solvers or inference engines.
The Feasibility Management Subsystem provides generic feasibility checking and repair mechanisms for local adjustments of the coordinated execution schedule within the space of

buffers already provided in the planned operations. This level of intervention is suitable for being delegated into automated procedures avoiding the need for triggering complex re-planning iterations.

The Monitoring Subsystem also exploit the generality of the reference models by framing the monitoring task into four well-identified activities that will capture the disruptive events that are rooted either on the orders dynamic or the resource availability.

A service-oriented solution applying standard SOMA techniques has been briefly described. The application of this technique allows the generation of the SOA specifications in full compliance with the SCEM business process and its requirements.

There are some issues that need to be addressed further for the proposal described in this chapter can be effectively deployed in real world scenarios. First, the generality of the reference models impose the burden of creating very rich and dense documents that collect information normally disperse in different business applications and databases. This is not a simple task in nowadays enterprise software. However as the service oriented approach gains momentum in the industry, and the concepts of cross-organizational information buses are becoming more and more popular, the gap for the requirements in this proposal will narrow in the short future.

7. References

Adhitya, A.; Srinivasan, R. & Karimi, I.A.(2007). A model-based rescheduling framework for managing abnormal supply chain events. *In Computers and Chemical Engineering 31* 496 – 518.

Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T., Ganapathy, S., Holley, K. (2008). SOMA: A method for developing service-oriented solutions. *IBM Systems Journal* 47 (3), pp.377-396.

BedraxWeiss, T., McGann, C., and Ramakrishnan, S. (2003). Formalizing Resources for Planning. I*n International Conference on Automated Planning and Scheduling (13th)*. Italy, Trento.

Bui, T.; Hempsch, C.; Sebastian, H. & Bosse, T.(2009). A multi-agent simulation framework for automated negotiation in order promising. *In Proceedings of the Fifteenth Americas Conference on Information Systems*, San Francisco, California August 6th – 9th.

Cauvin, A.; Ferrarini, A. & Tranvouez, E. (2009). Disruption management in distributed enterprises: A multi-agent modelling and simulation of cooperative recovery behaviours. *In International Journal Production Economics 122* - 429 – 439.

Derrouiche, R. , Neubert, G. , Bouras, A. (2008). Supply chain management: A framework to characterize the collaborative strategies. *International Journal of Computer Integrated Manufacturing* 21 (4), pp 426-439.

Fernández, E., Salomone E., Chiotti, O. (2010). Model based on Bayesian Networks for Monitoring Events in the Supply Chain. *IFIP Advances in Information and Communication Technology* 338, pp.358-365.

Guarnaschelli, A., Chiotti, O. , Salomone, E. (2010). Service Oriented Approach for Autonomous Exception Management in Supply Chains. *I3E 2010*, pp. 292-303.

Hoffmann,O.; Deschner, D.; Reinheimer, S. & Bodendorf, F. (1999). Agent-Supported Information Retrieval in the Logistic Chain. *In proceeding of the 32nd Hawaii International Conference on System Sciences*, Maui.

Hugin Expert A/S, Hugin Researcher (2010). In: www.hugin.com.

IBM ILOG OPL Development Studio API.

Kärkkäinen, M., Främling, K. & Ala-RiMKU (2003). Integrating material and information flows using a distributed peer-to-peer information system. *In Collaborative System Production Management* (Jaddev H.S., Wortmann, J.C., Pets H.J. Eds), pp 305-319, Kuwer Academic Publishers, Boston.

Kleindorfer, P.R., and Saad G.H. (2005)., Managing Disruption Risks in Supply Chains, Production and Operations Management 14 (1), pp.53–68.

Knickle, K. & Kemmeler J. (2002). *Supply Chain Event Management in the Field Success with Visibility*, AMR Research, Boston.

Lee, H. L., Padmanabhan, V., and Whang, S. (1997). The Bullwhip Effect in Supply Chains, Sloan Management Review 38 (3), pp.93-102.

Liu, Q.& Min, H. (2008). A Collaborative Production Planning Model for Multi-Agent based Supply Chain. *In 2008 International Conference on Computer Science and Software Engineering*.

Masing, N. (2003). *Supply Chain Event Management as Strategic Perspective – Market Study: SCEM Software Performance in the European Market*. Master Thesis. Universitié du Québec en Outaouasis.

Mellor, S., Scott, K., Uhl, A., Weise, D. (2004). *MDA Distilled, Principles of Model Driven Architecture*. Addison-Wesley Professional, Addison-Wesley.

Object Management Group, Service Oriented Architecture Modeling Language (SoaML), version 1.0 - Beta 2, http://www.omg.org/spec/SoaML/, 2009.

Papazoglou, M. P. and Van Den Heuvel, W. (2007). Service oriented architectures: Approaches, technologies and research issues, *VLDB Journal* 16 (3), pp. 389-415.

Radjou, N., L.M. Orlov, and T. Nakashima (2002). Adapting To Supply Network Change, in The TechStrategy Report, F. Research, Editor.

Simchi-Levi, D., Kamanisky P. & Simchi-Levi E. (1999). *Designing and Managing the Supply Chain*, Mc Graw Hill.

Soosay, C., Hyland, P. and Ferrer, M. (2008). Supply chain collaboration: Capabilities for continuous innovation. *Supply Chain Management* 13 (2), pp. 160-169.

Speyerer, J.K. & Zeller, Andrew, J.(2004) Managing Supply Networks: Symptom Recognition and Diagnostic Analysis with Web Services. *Proceedings of the 37th Hawaii International Conference on System Sciencies*.

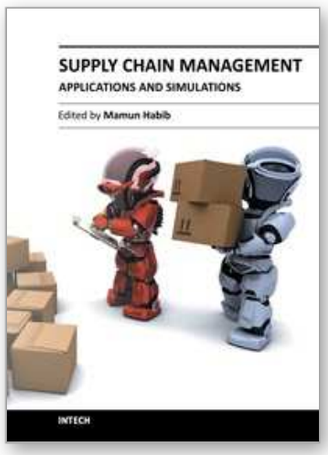
Szirbik, N.B., Wortmann, J.C., Hammer, D.K., Goosenaerts, J.B. M. & Aerts, A.T.M. (2000). Mediating Negotations in a Virtual Enterprise via Mobile Agents. *In Proceedings of the Academia/Industry Working Conference on Research Challenges* (IEEE Computer Society Eds.), pp. 237-242, Buffalo, NY.

Teuteberg F. & Schreber, D. (2005) Mobile Computing and Auto-ID Technologies in Supply Chain Event Management – An Agent-Based Approach. *In Proceedings of the Thirteenth European Conference on Information Systems* (Bartmann D, Rajola F, Kallinikos J, Avison D, Winter R, Ein-Dor P, Becker J, Bodendorf F, Weinhardt C eds.), Regensburg, Germany. (ISBN 3-937195-09-2)

Van Landeghem, H. & Vanmaele, H. (2002). Robust planning: a new paradigm for demand chain planning. *In Journal of Operations Management* 20, pp 769 – 783.

Wang, L.& Lin, S. (2009). A multi-agent based agile manufacturing planning and control system. *In Computers & Industrial Engineering* 57 620 – 640.

Zimmermann, R. (2006). *Agent-based Supply Network Event Management*. Whitestein Series in Software Agent Techonologies, Eds: Marius Walliser, Stefan Brantschen, Monique Calisti y Thomas Hempfling.



Supply Chain Management - Applications and Simulations

Edited by Prof. Dr. Md. Mamun Habib

ISBN 978-953-307-250-0
Hard cover, 252 pages
Publisher InTech
Published online 12, September, 2011
Published in print edition September, 2011

Supply Chain Management (SCM) has been widely researched in numerous application domains during the last decade. Despite the popularity of SCM research and applications, considerable confusion remains as to its meaning. There are several attempts made by researchers and practitioners to appropriately define SCM. Amidst fierce competition in all industries, SCM has gradually been embraced as a proven managerial approach to achieving sustainable profits and growth. This book "Supply Chain Management - Applications and Simulations" is comprised of twelve chapters and has been divided into four sections. Section I contains the introductory chapter that represents theory and evolution of Supply Chain Management. This chapter highlights chronological prospective of SCM in terms of time frame in different areas of manufacturing and service industries. Section II comprised five chapters those are related to strategic and tactical issues in SCM. Section III encompasses four chapters that are relevant to project and technology issues in Supply Chain. Section IV consists of two chapters which are pertinent to risk managements in supply chain.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Bearzotti Lorena, Fernandez Erica, Guarnaschelli Armando, Salomone Enrique and Chiotti Omar (2011). Supply Chain Event Management System, Supply Chain Management - Applications and Simulations, Prof. Dr. Md. Mamun Habib (Ed.), ISBN: 978-953-307-250-0, InTech, Available from: http://www.intechopen.com/books/supply-chain-management-applications-and-simulations/supply-chain-event-management-system

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen