Sense-and-respond supply chain using model-driven techniques

S. Kapoor
B. Binney
S. Buckley
H. Chang
T. Chao
M. Ettl
E. N. Luddy
R. K. Ravi
J. Yang

The results of an effort to build a sense-and-respond solution for a supply chain by using a model-driven development framework are described in this paper. One of the components of the framework is the IBM Research-developed model-driven business-transformation (MDBT) toolkit, a set of formal models, methods, and tools. The inventory optimization analytics used to improve supply chain performance are also described. This approach is illustrated through a case study involving the IBM System x™ supply chain.

INTRODUCTION

To remain competitive in a rapidly changing marketplace, an organization must be able to transform itself rapidly, reliably, and within a given budget. The complexity and scope of the business transformation process make its cost high and its outcome uncertain. Over the past six years, we have been exploring and piloting the sense-and-respond (SaR) concept to support adaptive business operations.[1] Recent advances in software development, and in particular the use of model driven development (MDD**)[2] technology, have made the development of SaR systems easier to achieve, in two ways. First, sensing mechanisms can be developed and modified faster and more reliably. Second, business transformation as a response to sensed changes can also be implemented and modified more quickly and easily by using MDD. We demonstrate both of these points in a single case study.

In 2003 the IBM Integrated Supply Chain (ISC) organization and the IBM Research Division undertook a joint project whose goal was to transform a supply chain that involves a high-speed customer fulfillment function and short product cycles. This effort focused on synchronizing demand with supply and reporting shortfalls or surpluses.[1] A second effort, launched in 2005 by the IBM Research Division and the IBM CIO (chief information officer) organization, aimed at building a reusable SaR solution based on model-driven technology.[3,4] We describe in this paper the results of these efforts, in which the model-driven development framework is applied to the transformation of the IBM System x* supply chain.[5] We also describe the inventory optimization analytics developed to improve supply chain performance.[6]

The rest of the paper is organized as follows. In the next section we describe the MDD methodology

and its incorporation into the MDBT toolkit. Then we discuss the software-as-a-service concept and its implementation into the SaR solution infrastructure. We describe the business environment, the architecture of the MDBT development environment, and its three major components: data integration, monitoring, and management, and the user interface in the form of dashboards. We close by summarizing the business and information technology (IT) benefits achieved from this effort.

MODEL-DRIVEN BUSINESS TRANSFORMATION TOOLKIT

Model-driven business transformation (MDBT) is a model-driven technology developed by the IBM Research Division for enabling rapid, reliable and cost-effective transformation of business processes. MDBT realizes business strategies and transforms the full life cycle of business processes by choreographing workflow and human activities achieved through a four-layered modeling architecture involving business and IT users.[4] In addition, MDBT leads to solutions in which the performance of the business is made visible to management by combining monitoring and graphical display functions in the user interface. MDBT thus helps optimize business performance, an activity known as business performance management (BPM).[7] An essential part of BPM is the monitoring and management of key performance indicators (KPIs), a set of indicators which help management determine whether the business in on track to meet its objectives.

The model-driven development framework we use in this work is based on the MDBT toolkit, a set of formal models, methods, and tools.[4] The framework employs and extends the IBM business observation metamodel and introduces a data warehouse metamodel.[3] The MDBT toolkit uses the business observation metamodel to formally define expressions and data to compute operational KPIs, relationships between the KPIs, problem situations that are triggered based on KPI violations, and appropriate corrective action. It uses the data warehouse metamodel to generate a performance data warehouse schema; the schema is used to store the KPIs to be displayed on a dashboard. In addition, the performance data warehouse also stores historical data, thereby enabling analytical services like analyzing trends and data mining.

Monitoring and measuring the business process is enabled through the MDBT continuous improvement cycle with model-driven development methods and tools.[4] The following steps are performed in sequence:

1. *Gathering business requirements*—Includes specification of the KPIs, problem situations, alerts, and actions that are normally executed to resolve the problems. Also included is the visualization of the KPIs and alerts on the dashboard.
2. *Modeling*—Involves using the two BPM models: the observation model, which includes BPM elements for monitoring the performance of business processes, and the data warehouse model, which enables the data warehouse schema and the display of its elements on a Web portal.
3. *Transforming and code generation*—Involves applying the MDBT toolkit to transform the models and generate deployable code for the BPM-service runtime components and data warehouse.
4. *Deployment*—Involves deploying the generated BPM services, data warehouse, and dashboard on the target platform.
5. *Runtime monitoring*—Involves monitoring of business processes. There are two distinctive patterns of monitoring. In *proactive monitoring*, we periodically inspect the BPM dashboard for warnings and alerts and, when necessary, we take appropriate corrective action. In *management by exception*, we wait passively for the system to issue a warning or an alert. The notification channel for such an alert could be configured based on the user's delivery preferences, the most common being by e-mail and through instant messaging. It is also common to have the BPM service trigger a business process for corrective action based on the detection of an alert.
6. *Action*—Includes notification by e-mail or instant messaging, or an action performed by a business process triggered in order to resolve critical business situations.

Monitoring often leads to new requirements that need to be implemented. This information is fed back to Step 1, thereby starting the continuous improvement cycle again.

Aside from adding a new data warehouse metamodel, the MDBT toolkit extends the observation

metamodel in a variety of ways, three of which are especially critical to satisfying the hub-management solution described in a later section. These three extensions enable a variety of input sources and data types to be processed by the observation model expressions, and they make the expressions more powerful and suitable for complex metric computation. This is important because one of the key motivations to use the model-driven methodology embodied in the MDBT toolkit is to capitalize on automated code generation and encapsulate the computation logic within the model, thus avoiding the need for handcrafted code. Subsequent change requests, such as an update to an expression used in computing a metric or an additional event attribute, can be carried out by updating the model and generating the new code.

The three aforementioned MDBT extensions are as follows:

1. *Enable the use of an external database as a data source*—Generating real-time ad hoc business events is the mechanism previously used to collect data from the myriad of systems monitored by the application. The need arose to allow direct access to external databases, in order to avoid the task of generating a business event to be issued to the BPM system. This feature is primarily used to read in the values monitored with respect to business thresholds, which are used for evaluating monitoring rules as described in the hub-management solution of our case study.
2. *Enable the processing of complex data types such as structured data and arrays*—It is common for supply chain processes to have KPIs incorporating measures such as forecasted demand and supply. For example, the KPI DOS, which stands for days of supply, is calculated based on this forecast, which is typically represented as an array. Whereas Java** programs are usually developed in order to manipulate arrays and calculate the value of these KPIs, these computations can now be incorporated within the observation model followed by the code generation, which represents considerable savings in cost and time.
3. *Enable the synchronized invocation of external services*—Analytics and optimization play important roles in most BPM solutions. They are usually custom built or, in some cases, configured specifically for the business process and enabled as Web services. This extension of the MDBT toolkit supports the invocation of external services through a Web service call.

SaR infrastructure

Software as a Service (SaaS) is a new model of software delivery in which a strategic partner (also called a service provider) provides maintenance, technical operations, and support for the software on a subscription basis. While SaaS provides many benefits in terms of cost and time, there are many hidden facets to the business model, which are easily overlooked at first glance. Limited configuration, inability to provide quick custom changes, and difficult integration with existing applications, especially with legacy systems, are just some of the complaints from many of the experienced SaaS customers.[8] SaaS applications are also usually limited to a specific industry or repeatable business processes, or both, and therefore not applicable to many of the custom development efforts being pursued within organizations. SaaS is therefore reaching a critical point in its evolution, and SaaS vendors are looking to combine software, business process services, and technology integration and management to remain competitive and overcome some of the challenges and limitations of the current business model.

IBM has defined and popularized the notion of "on demand" and is focused on enabling itself and its customers to be on demand organizations. Information availability coupled with cost-effective technology deployment play a key role in the transformation to being on demand. Information as a service is the IBM vision of how information should be open (through industry standards) and easily accessible (in minutes, not months) to people, applications and business processes.[9] This allows for improved agility, where information is not tied to proprietary systems, formats, and technologies. The IBM SaR initiative provides "information as a service" to its customers. It aims to provide a cost-effective solution to the two key on demand drivers, thereby enabling business transformation through the use of IT, as follows:

1. *Information availability*—Critical business events sometimes go unnoticed and fail to trigger the desired response. SaR provides the ability to monitor business processes and enterprise data sources, to retrieve and analyze KPIs, to generate

alerts, and to visualize the state of the business by means of an enterprise dashboard. MDD allows for rich customizations to be built into the solution. Automated code generation from the models targeted toward a service-oriented architecture platform allows for quick custom changes and easy integration with legacy systems.[9]
2. *Technology deployment*—SaR provides a multitenant shared-server infrastructure for deploying and hosting the models. *Multitenant* implies that multiple models, each belonging to a different customer, execute on a single instance of the system. *Shared server* implies that the models share computing resources on an on demand, pay-per-use basis.[10]

The SaR initiative is part of the overall IBM business intelligence solution. It currently supports two IBM internal business processes, with many more to be introduced in 2007 and beyond. Careful thought has been taken in building the roles and responsibilities providing support to the models and shared server environment. Three groups of users have been defined:

1. *Business engineers* drive the business transformation by modeling the business requirements using the models natively provided by SaR. These models are handed to the application support specialists for further analysis, thereby working toward reducing the business-IT gap.[11]
2. *Solution integrators* are responsible for validating, integrating, and testing the models. They add IT-level configurations to the business models before generating the code and packaging for deployment on the shared infrastructure.
3. *Infrastructure specialists* are responsible for the maintenance and support of the server infrastructure. Applying product upgrades and regular server maintenance are some of the tasks performed by these users.

The solution owners share the cost of the single infrastructure and are charged on a pay-per-use basis. The solution-integrator and infrastructure-support specialists provide support across multiple models; this further reduces the cost for each solution. For each solution, the responsible team independently develops the appropriate models, databases, and dashboards. When a model undergoes a change at the hands of the responsible team (the business engineer and the solution integrator), the infrastructure supports these changes without interrupting the other users and or requiring a refresh of the SaR runtime. This is a critical feature for a shared environment in which multiple users share a single production infrastructure.

IBM SYSTEM X SUPPLY CHAIN: A CASE STUDY

The business transformation of the supply chain aims at achieving a competitive advantage through supply chain performance measured in operational cost and customer-order serviceability. The specific capabilities required to enable transformations are usually known; the critical determinant of success is the efficiency and speed with which the appropriate business processes can be enabled. In other words, it is not the "what" but the "how" that determines success. In this section we describe the SaR that was developed for System x to manage the supplier relationship processes better. The system uses the model-driven approach and is deployed on a multitenant, shared-server infrastructure.

Business environment

Managing a complex multitier supply chain has been hindered by the lack of visibility across multiple tiers. Disparate systems and slow, manual processes hamper the exchange of information. In the supply management processes across the progressively outsourced System x supply chain, supply assurance in tiers 1 and 2 depends on key business measures such as inventory levels, demand forecasts, supply "commits," and consumption rates. In this extended value chain, commodity supply through replenishment hubs (centralized supply warehouses) and commodity inventory management at hardware systems manufacturing sites (tier 1) play a key role (see ***Figure 1***).

While the outsourcing of operations offers advantages and cost savings, it also forces organizations to innovate and find new ways to manage complex processes across multiple supply-chain tiers. The business transformation team identified and sought to achieve optimal inventory management. At the same time, the team endeavored to provide the rapid identification of alerts in response to events in the supply chain indicating the need for corrective actions exercised jointly across functional constituencies within Manufacturing, Fulfillment, and Procurement, along with supply partners. In doing so, incipient supply shortages and surpluses would be more quickly identified to improve downstream

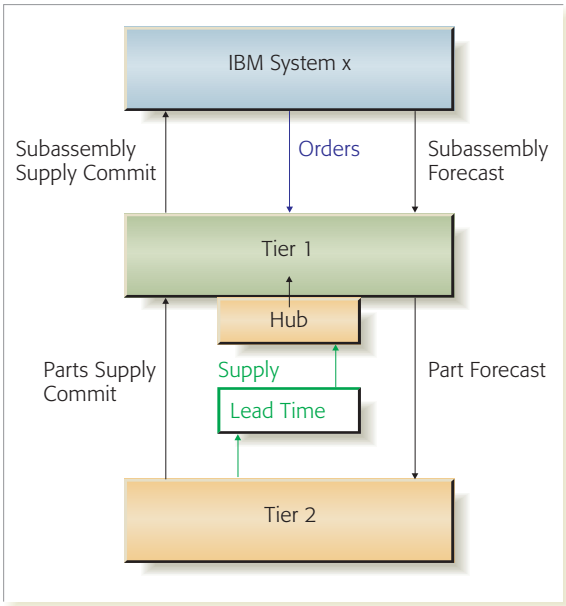

Figure 1
IBM System x multitier supply chain

supply availability and order serviceability, and to improve asset utilization. The goal was to enable dynamic hub inventory visibility and management, tied directly to the strategic on demand supply chain attributes of SaR and underpinned by rigorous supply management analytics. More important is precisely how the solution itself is designed and delivered. An IBM Academy study recently concluded that application development and deployment of even small changes to enterprise systems within the IBM Corporation were taking up to a year and resulting in inordinate cost.

In the following section, we describe the high-impact nature of MDD applied to supply chain problems. The guiding principles for this effort were

- to focus on imbalances within the supplier hubs to maintain high fill rates and to minimize inventory liabilities,
- to develop a Web portal in which users receive alerts of current demand-supply imbalances and recommendations for possible solutions, and
- to track and manage solution effectiveness through demand and supply imbalance resolution to make appropriate midcourse corrections and expedite decision-making to prevent future imbalances.

Architecture and flow

The hub-management solution is an inventory management and optimization application that streamlines the supplier management process of an organization. It interfaces electronically with data feeds from various suppliers, recommends optimum volumes of inventory, aggregates, correlates and monitors data according to business rules and thresholds, and generates alerts that are made visible on a Web-based dashboard. It helps the IBM System x supply chain to "sense and respond" to demand and supply changes, quickly flagging demand-supply imbalances and potential shortage and surplus situations. Powerful exception-management and decision-support capabilities allow all tiers of this supply chain to track actual execution against forecasts and demand against inventory positions. It also incorporates an inventory optimizer that calculates optimum inventory bands, meeting business-acceptable fill rates while minimizing inventory liabilities.

The business transformation team realized that the existing in-house hub-management solution was no longer adequate. Like most legacy systems, it was an application built from the ground up by using traditional development techniques, which resulted in large cost and time penalties for every change request. Proprietary architecture, "file and fit" transformation techniques, and complex waterfall development methodology was becoming increasingly expensive to maintain and had limited scalability. It resulted in high defect rates, client dissatisfaction, and a disconnect between the business and IT teams. In addition, the legacy solution lacked easy access and visibility for the suppliers, a serious roadblock to extended supply chain management.

To overcome many of the limitations noticed with the legacy system, the team turned toward using model-driven technologies and focused their efforts on reengineering the existing legacy solution. *Figure 2* shows the architecture of the MDBT development environment, highlighting the roles and responsibilities of the various team members during solution development.

The solution is composed of three main components: data integration, monitoring and management, and the dashboard. IBM Rational* Software Architect (RSA) is used as the modeling environment.[12] Once

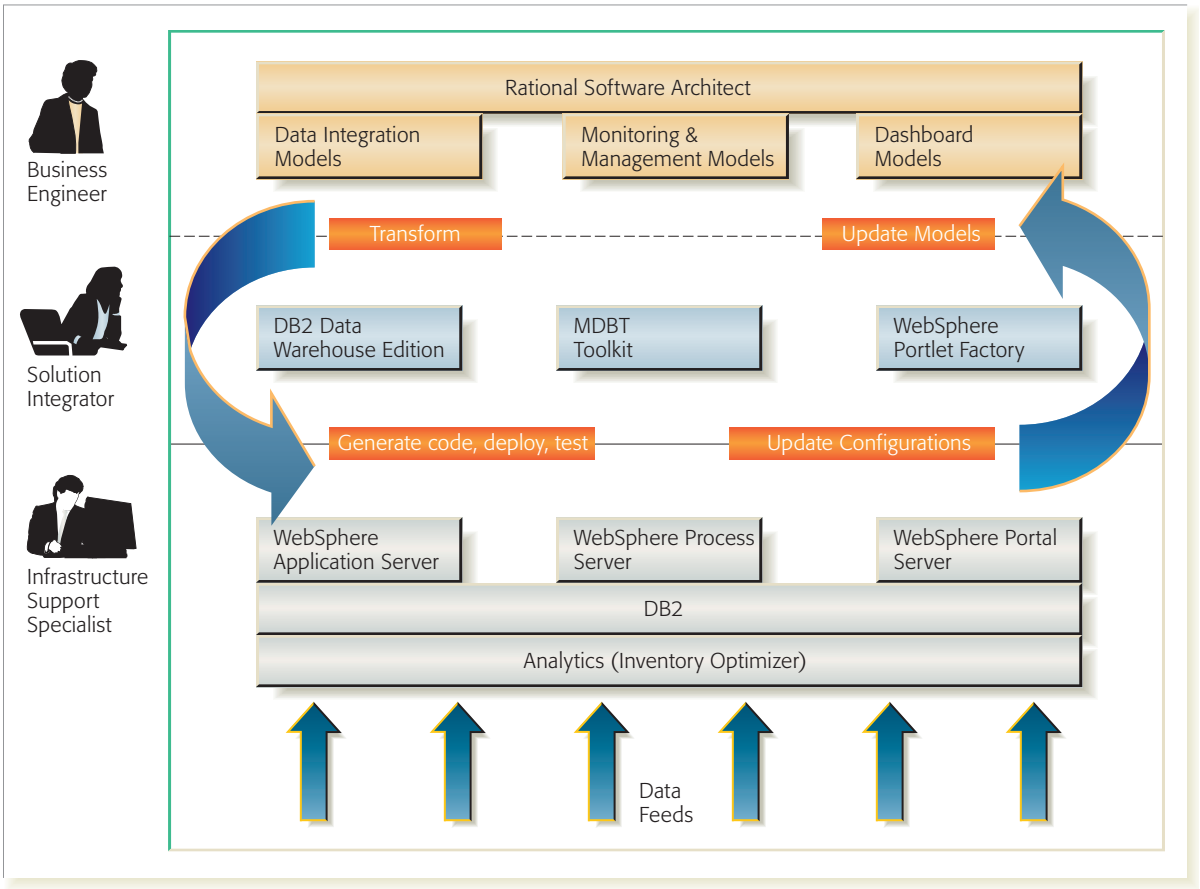

Figure 2
Architecture of the MDBT development environment

the models have been constructed, the business engineer shares them with the solution integrator, who further augments the models with IT-specific configurations, generates code, and packages it for deployment performed by the infrastructure specialist. Once deployed, the solution is tested by the business engineer and solution integrator to make sure the results are as expected. If any changes need to be made, the models are updated by the solution integrator or the business engineer, followed by reapplication of the transformation and code generation. This innovative method is an initial step toward reducing the business-IT gap because it clearly defines the deliverables for every team member and formalizes the handoffs between the teams. It also reduces the overall cost for solution development because it involves fewer resources and minimizes the code development (the code being auto-generated). Additionally, the modeling framework also improves application quality by restricting the touch points for code modifications by developers, thereby preventing them from breaking architecturally sound code. The remainder of this section dives deeper into each of the three components, highlighting the individual models constructed and the challenges encountered along the way.

Data integration

The discipline of data integration comprises best practices, patterns, techniques, and tools for achieving consistent access and delivery of data throughout the spectrum of the enterprise.[13] In many organizations there is a general dissatisfaction with the lack of data consistency and a constant desire for the proverbial "single version of the truth." Broadly speaking, data integration spans a variety of scenarios including the following:

- Data warehousing; extract, transform, and load (ETL); and business intelligence (BI)

Table 1 Hub management KPIs

KPI	Description
Inventory	The inventory volumes currently in the hub owned by the tier-2 supplier but managed by tier-1
Stock	The inventory volumes owned and managed by the tier-1 supplier not currently allocated to any manufacturing order
WIP	The inventory volumes owned and managed by the tier-1 supplier that have been allocated to a manufacturing order
Receipts	The volume of parts that were received by IBM from the hub
Lead time	The replenishment lead time for the hub
Parts Forecast	The parts forecast that is generated by listing in detail the machine type forecast shared by IBM with the tier-1 supplier
Plan-commit for parts	The volume planned and committed by the tier-2 supplier against the forecast

- Data migration, synchronization, and conversion
- Master data management (MDM)
- Structured and unstructured data
- Federated data management

The SaR initiative aims to cover the broader data integration scenarios, but for the purposes of the hub-management solution, we implemented only the first (warehousing, ETL, and BI) scenario. The integration is composed of four broad steps:

1. *Extract*—Data integration within the hub-management solution is triggered by data feeds received from the tier-1 supplier. These feeds, which correspond to individual part numbers, arrive at various times in the week, each with different attributes. They can be grouped into two categories:
   A. *Weekly data feeds*—after the weekly MRP run by the tier-1 supplier has been completed and the supply-commit statements and inventory volumes in the hub are available, the tier-1 supplier publishes the data to IBM. This typically occurs early Friday morning with a refresh on Saturday morning to update any errors or late feeds from the tier-2 suppliers. The content of the data feeds covers the measures shown in *Table 1*.
   B. *Daily data feeds*—typically occur every day and span the hub inventory, stock and WIP measures indicating the latest supply positions available in the week. The intent is to enable a near-real-time snapshot of the supply chain, making quick responses to shortages and surplus situations possible.
2. *Cleanse*—The data received as part of the daily and weekly feeds is first cleansed to remove any anomalies. One of the common data discrepancies noticed was inconsistent supplier names. A single supplier was referenced by several names, thereby causing disruptions in the upstream analysis performed by using the dashboard. Location and commodity names were also inconsistent and needed to be cleansed before any upstream analysis. Various lookup tables were created that referenced the master set of supplier, commodity, and location names, which was used by the data integration model as the basis for cleansing the data. Once the data was cleansed, it was loaded in a staging area to be analyzed further.
3. *Transform*—The data retrieved from the staging area is normalized, transformed and stored in the performance warehouse. As part of the normalization, new locations, suppliers, and part numbers are identified, and master tables in the performance warehouse are updated accordingly. A supplier hub, identified by a unique combination of location, supplier, and part number, is detected, assigned a unique identifier, and loaded in the master hub table.
4. *Generate events*—Once the hubs have been identified, the measures received from the supplier feeds are assigned to the hub and published on the enterprise service bus to be consumed by the subsequent monitoring and management

component. This allows for loose coupling between the data-integration and monitoring and management component connected by a JMS (Java Messaging Service) publish-subscribe interface.

We are currently using IBM DB2* Data Warehouse Edition (DWE)[14] as the platform for the data integration models—it supports MDD for data warehousing, ETL, and BI solutions.

Monitoring and management

Continuous and real-time monitoring helps to identify and eliminate problems before they develop into larger issues. It helps organizations to monitor operations and manage efficiency targets. Monitoring in the hub-management solution is performed on the inventory volumes at the supplier hubs. Two weeks of supply, or DOS10 (10 days of supply), has usually been considered as the industry-standard policy for managing most hubs. This is calculated as an average of the six forward-looking weeks of forecast. This can be expressed as follows:

$$\text{DOS10} = \frac{\sum_{i=1}^{i=6} Forecast_i}{3}$$

We define two distinct policies to enable monitoring of the hubs: fixed DOS-based thresholds and dynamic optimum thresholds.

For the fixed DOS-based thresholds policy we define the following:

1. *DOSRedMin*—This is the minimum acceptable level of inventory below which a shortage alert would be generated by the system. The DOSRedMin threshold is expressed as follows:

   $$\text{DOSRedMin} = \frac{DOSRedMinTolerance}{10} \times DOS$$

   The default DOSRedMin tolerance is set to 5.
2. *DOSYellowMax*—This is the high yellow mark, above which a yellow surplus alert would be generated by the system. The DOSYellowMax threshold is expressed as follows:

   $$\text{DOSYellowMax} = \frac{DOSYellowMaxTolerance}{10} \times DOS$$

   The default DOSYellowMax tolerance is set to 16.
3. *DOSRedMax*—This is the high red mark, above which a red surplus alert would be generated by the system. The DOSRedMax threshold is expressed as follows:

   $$\text{DOSRedMax} = \frac{DOSRedMaxTolerance}{10} \times DOS$$

   The default DOSRedMax tolerance is set to 21.

The dynamic optimum thresholds are generated by the inventory optimizer described in the section "Analytics." The intent was to dynamically generate a min-max band, indicating the optimum inventory levels for every hub, based on the analysis of historical customer demand and the hub-replenishment lead time.

The DOS thresholds are controlled and updated by the user by means of the enterprise dashboard. In contrast, the dynamic optimum thresholds are generated automatically by the inventory optimizer based on analysis of historical receipts, open demand, and lead time to replenish the hubs. The intent was eventually to replace the traditional DOS monitoring policy with the more accurate statistical control bounds provided by the inventory optimizer. The inventory in the hub is monitored against the DOS thresholds as well as the dynamic optimum thresholds generated by the inventory optimizer, creating three possible alerts (see ***Figure 3***):

1. *Shortage alert*—When the inventory falls below the DOSRedMin threshold, a shortage alert is generated, indicating shortage of volumes for the particular part.
2. *Yellow surplus alert*—When the inventory falls between the DOSYellowMax and the DOSRedMax thresholds, a yellow surplus alert is generated, indicating a warning of surplus parts.
3. *Red Surplus Alert*—When the inventory crosses the DOSRedMax threshold, a red surplus alert is generated, indicating a critical situation involving surplus of the particular part.

Similar shortage and surplus alerts (only red surplus) are generated by evaluating the inventory against the dynamic thresholds generated by the inventory optimizer.

Gathering of requirements

As part of the design phase, in preparation for modeling we capture the business requirements by using a pictorial BPM model, which serves as a

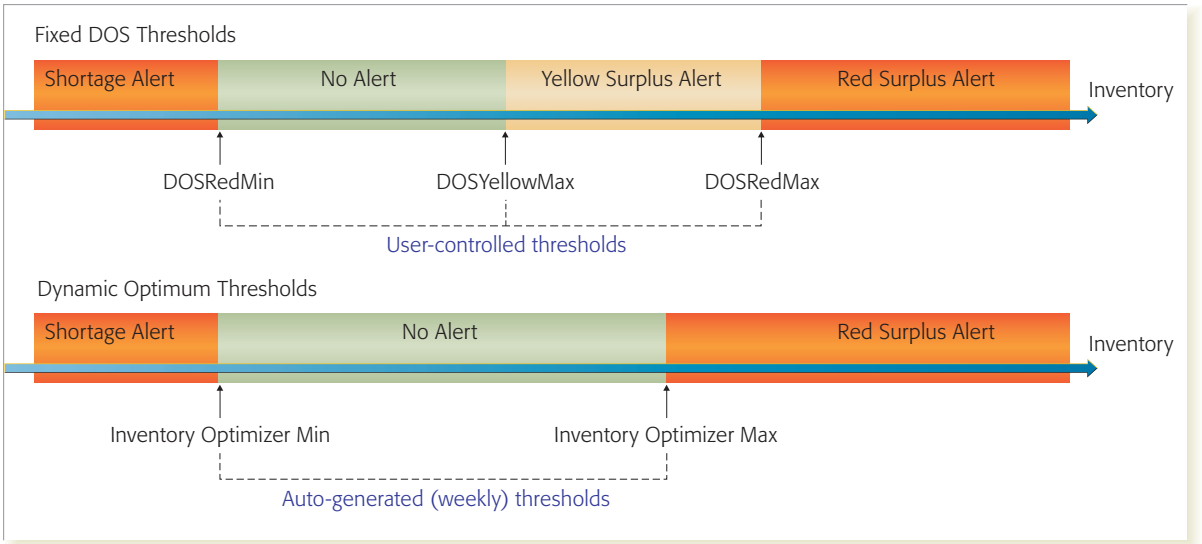

Figure 3
Monitoring policies for the hub management solution

blueprint of the actual observation model and a communication vehicle to confirm requirements with business users. The purpose of this model is to specify "what" is to be monitored, whereas the observation model specifies the "how." The pictorial BPM model encapsulates and translates the business requirements into BPM elements, which can then be specified with an observation model. Such BPM elements include monitoring scope, input sources, metrics and KPIs, situations to be detected, and the actions to be triggered. ***Figure 4*** shows an example of a pictorial BPM model for the hub management solution.

Monitoring scope represents the context within which the monitoring and management of the inventory is performed. In the solution, the monitoring scope is defined as a supplier hub identified by a unique `hubId`, along with the week in which the data were received.

Input data sources (at the bottom of Figure 4) represent sources containing data to be monitored and measured. There are two data sources in the solution: (1) business events such as inventory events and forecast events; these events encapsulate the data measures received from the tier-1 supplier and are published from the data integration component described earlier; and (2) external data store, where additional input data can be retrieved from a database for computing metrics and KPIs.

Metrics and KPIs (in green and yellow) represent data either obtained from input sources or calculated based on other metrics. The metrics in yellow are the threshold values retrieved from the external data store and used for computing the DOS-based thresholds, which are the basis for detecting surplus or shortage situations. The thresholds are flexible and can be changed from the dashboard.

Maps (in purple) represent operational expressions that trigger the evaluation of operations associated with a metric.

Situations (in orange) represent exceptional conditions that require attention.

Actions are represented by a metric with a distinct value, which is set when a situation is detected. Although there are no direct business actions modeled or triggered by the BPM system in the hub-management solution, that value is used to display a corresponding alert on the BPM dashboard for the stakeholders to review and take further manual corrective actions.

Modeling the business requirements

In this step, we model the business requirements by using the observation model based on the pictorial BPM model. In addition, we also create the data warehouse model by using the elements defined in the observation model.

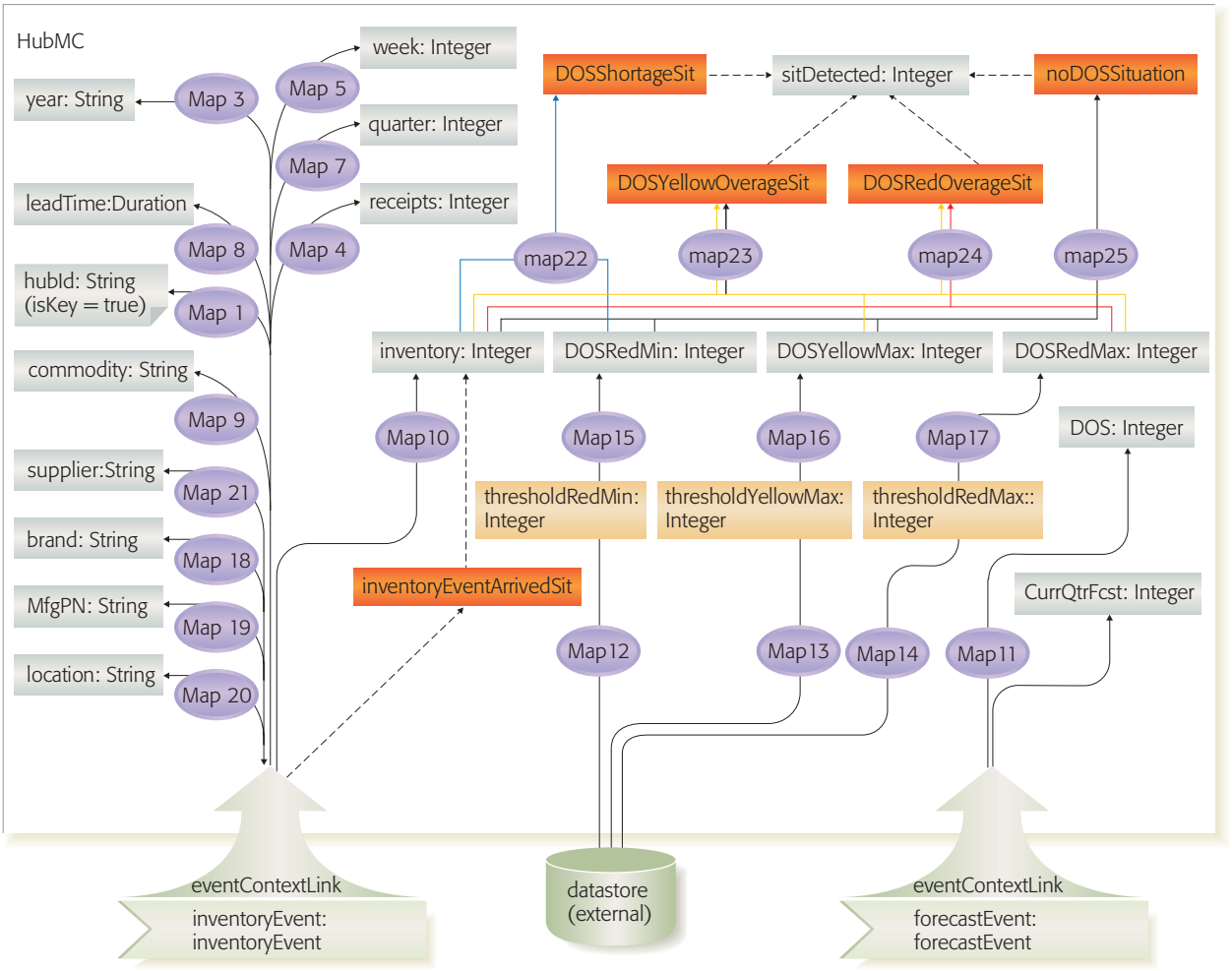

Figure 4
Pictorial representation of the BPM observation model

Observation model. The observation metamodel extends the Unified Modeling Language** (UML**)[15] to formally define KPIs and other BPM elements. The MDBT framework employs and extends the IBM business observation metamodel and is used as the basis for creating an observation model. The toolkit has created profile extensions to the UML modeling aspect of RSA so that observation model stereotypes can be defined by using these profiles. See Reference 3 for more information about the observation model and how to create and use it. In this paper we use a simplified version of the hub-management observation model in order to highlight key BPM elements in it, as shown in ***Figure 5***. The complete model contains additional metrics, maps, and situations as illustrated in Figure 4.

Monitoring Scope is defined by a class called *hubMgtMC* that represents the supplier hub management context and is an instantiation of the ⟨⟨ManagementContext⟩⟩ stereotype.

Input Sources are of two types. The first type of input source is from external business events, and it is defined by two classes called *inventoryEvent* and *forecastEvent*, both of which are stereotyped as ⟨⟨BusinessEvent⟩⟩. Each event contains attributes representing business measures supplied by the tier-1 supplier (e.g., inventory and forecast) to be monitored by the system. One attribute, `hubId`, is considered a key attribute in both business events because it uniquely identifies the event and the management context instance the event is intended for; it is used for correlating events and context instance during runtime execution of the model. The

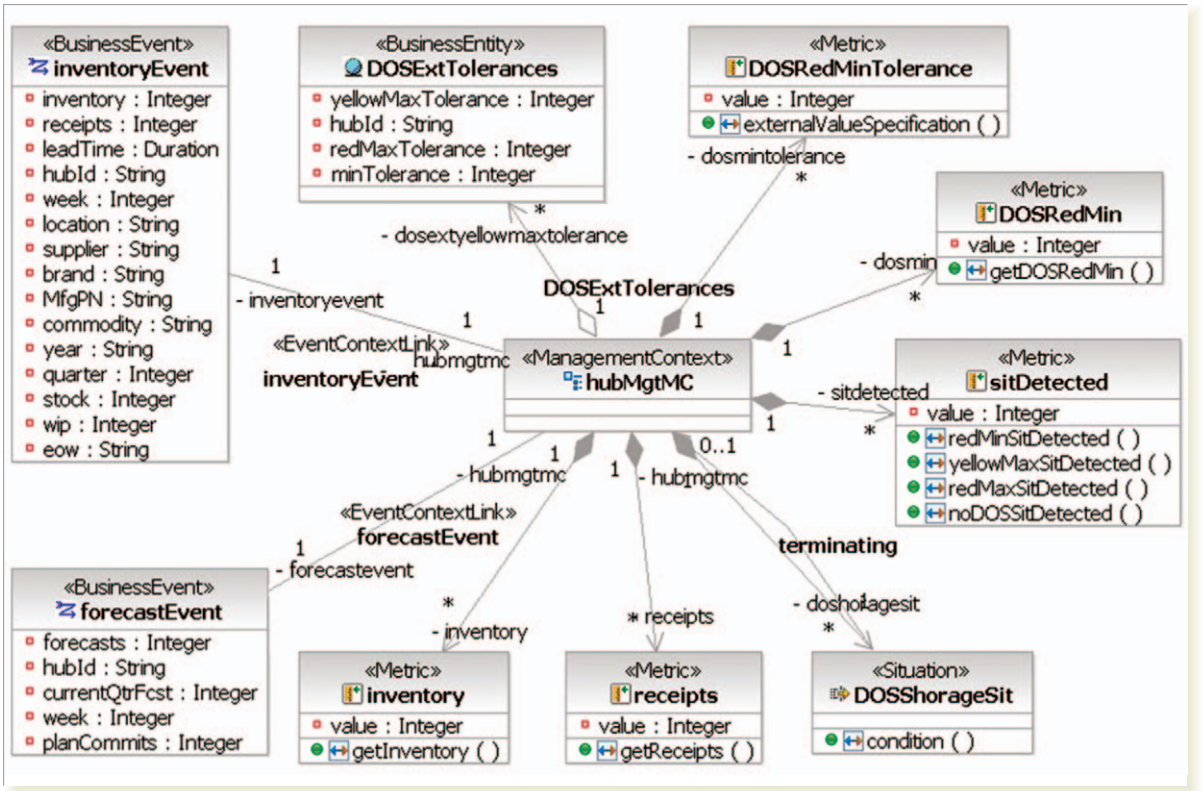

Figure 5
Hub management UML observation model

second type of input source is an external database for storing the DOS thresholds. It is defined by a class called *DOSExtTolerances* and stereotyped as ⟨⟨BusinessEntity⟩⟩. *DOSExtTolerances* contains three DOS threshold attributes along with the hubId. An association is defined between the *DOSExtTolerances* class and the *hubMgtMCManagementContext* class, and a constraint called *lookup* is created and stereotyped as ⟨⟨BusinessRule⟩⟩. The rule specifies how the database lookup is performed.

Metrics and maps are used to implement DOS thresholds. To retrieve the value for each of the DOS thresholds from the external database, a corresponding metric must be defined to hold the value. For example, a class called *DOSRedMinTolerance* stereotyped as ⟨⟨Metric⟩⟩ is created for the DOS Min threshold. Additional metrics in the management context instance can be obtained either directly from attributes of a *BusinessEvent* instance or computed from other metrics. For example, metrics such as *inventory* and *receipts* are obtained from an *inventoryEvent* instance, whereas the metric *DOSRedMin* is computed by using *DOSRedMinTolerance* and *DOS*.

Situation is defined by a class stereotyped as ⟨⟨Situation⟩⟩, an example of which is *DOSShortageSit*. An operation is defined as the gating condition of the situation and stereotyped as ⟨⟨Map⟩⟩. For example, the expression for evaluating the operation of *DOSShortageSit* is defined as *hubMgtMC.inventory* < *hubMgtMC.DOSRedMin*. When the condition is evaluated to true, implying that the inventory is below the DOSRedMin for a particular hub, a metric called *sitDetected* is set to an integer value representing the situation detected. The value 2, for example, indicates a red shortage alert on the dashboard.

Data warehouse model. With the KPIs identified, an important next step is to work with the business users to define the visualization of the KPIs on the

enterprise dashboard. These business requirements are captured by the data warehouse model. However, before we can build such a model, we need to understand the concept of dimensions and facts; interested readers are referred to Reference 4 for additional details. In this paper, we provide a condensed description, using a sample data warehouse model for the hub-management solution.

Figure 6 shows the sample data warehouse model, comprising of seven dimensions and four DOS-related facts; these are the monitored metrics that business users wish to view on the dashboard. A dimension consists of one or more related metrics grouped in a hierarchical fashion; the grouping is used as the basis for rearranging the data so that it can be viewed from different perspectives. For example, in the *calendarDim* dimension of the model, three attributes (*year, quarter*, and *week*) are related by a parent-child hierarchy, with *year* being the root of the hierarchy and *week* being the leaf. Data can thus be visualized at three different levels, that is, by year, by quarter, and then by week. Dimensions containing only one metric, such as *brandDim* and *locationDim*, are primarily used to group or filter data, such as display hubs of a certain brand or at a specific location. Dimensions can also be grouped together to aid in analysis of the data. For example, combining the brand, location, and calendar dimension would provide insights into the KPIs (e.g., Inventory or Supply) for a brand at a particular location within a given time period. A *fact* is a quantifiable and measurable metric, often qualified by dimension attributes to provide the appropriate business context, such as, DOS is a fact qualified by the dimensions *brand*, *location*, and *week*.

For each dimension to be defined, a class is created, such as *brandDim*, and stereotyped as ⟨⟨Dimension⟩⟩. Each attribute in the dimension is stereotyped as ⟨⟨DimensionLevel⟩⟩ with an attribute called *level*. The value of the *level* starts at 0 for the leaf node in the hierarchy, such as 0 for *week*, 1 for *quarter*, and 2 for *year* for the *calendarDim*. One of the attributes is the primary key, stereotyped as ⟨⟨PrimaryKey⟩⟩; for example, *week* is the primary key for the *calendarDim* and *brand* is the primary key for *brandDim*.

To link a fact with a dimension, a directed link is created from the fact to the dimension class and stereotyped as ⟨⟨MeasureDimensionLink⟩⟩. Facts can be linked to multiple dimensions, thereby allowing for rich data visualization by using OLAP[16] techniques.

Once the modeling is complete, a transformation is applied to the observation model to generate code for the MDBT services targeted to a particular runtime platform, such as IBM WebSphere* Process Server. In addition, the transformation is also applied to the data warehouse model to generate the performance warehouse schema currently targeted for deployment on DB2.

Analytics

The inventory model developed as part of the hub-management solution incorporates the trade-offs between serviceability, liability exposure, and inventory in an outsourced supply chain with vendor-managed inventory (VMI). Applying nonlinear optimization techniques, the model helps assess buffer stocking levels and the operating and financial impacts of liability exposure; it also helps identify the best VMI strategy for particular supplier contracts. Inventory buildups across the supply network impose a great financial risk as demand begins to taper off before a recognizable downward trend emerges or when a market downturn coincides with the end of life of a product.

To mitigate such drawbacks, we developed advanced analytics that, in conjunction with better supply chain visibility, optimize the pre-positioning of supplier parts inventory in the IBM extended supply chain.[6] The analytics complement existing planning applications by leveraging transactional data from enterprise business applications. Through this technology, business managers are able to make adjustments to optimize inventory based on monitored performance and to reduce response time by using decision analysis support.

The model assumes that the supplier is responsible for managing and replenishing component inventory in a VMI stocking location (i.e., supplier replenishment hub) based on an IBM demand forecast. The objective is to find the optimal operational policy for managing the VMI buffer stocks, taking into consideration order delinquencies and the cost of potential liabilities and buyback of surplus inventory. More specifically, we consider a finite time horizon of T weeks where the demand

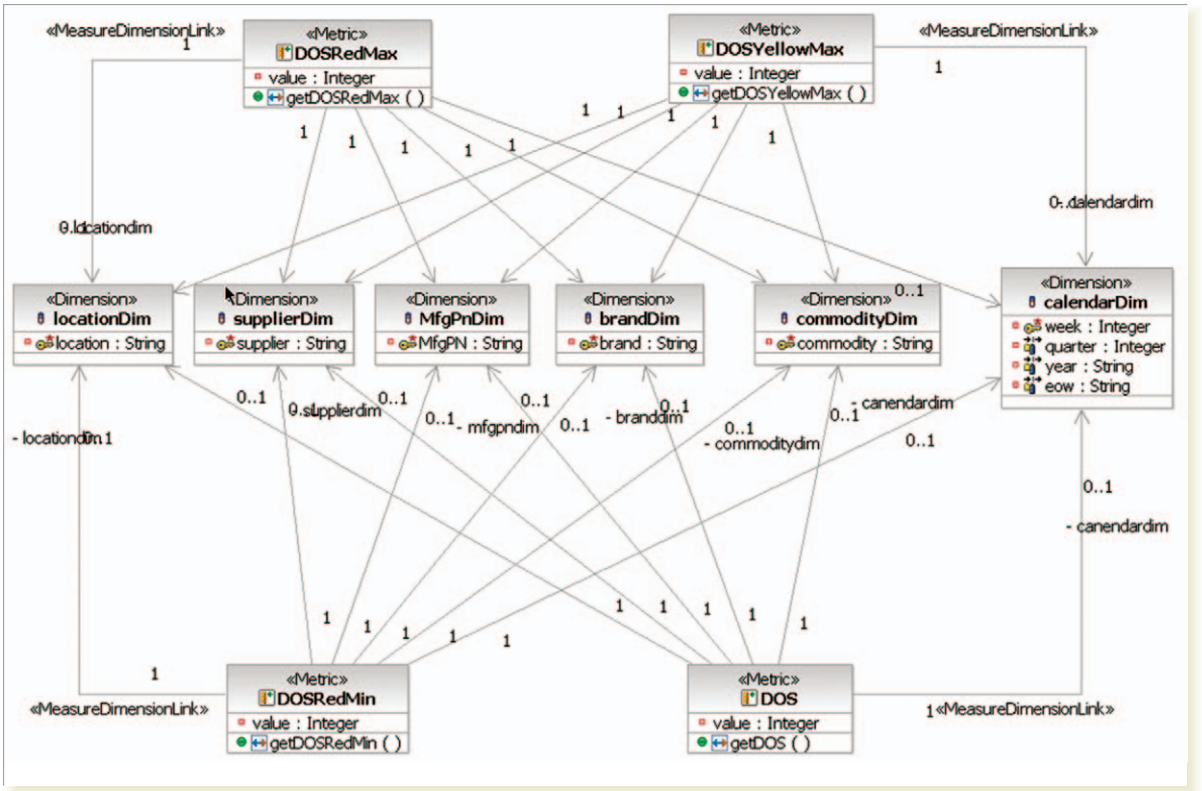

Figure 6
Hub management UML warehouse model

forecast for week t is characterized by an independent normal random variable with mean μ_t and standard deviation σ_t. The demand forecasts are nonstationary, that is, μ_t and σ_t can vary over time. Let I_t and B_t be the inventory and backlog at time $t = 1,2,\ldots, T$, respectively, with initial inventory I_0 and initial backorder B_0. The costs incurred at the end of each week t are based on the inventory level I_t and the backlog level B_t. The decision variables are the reorder points at each time t, denoted by R_t, under a lead time of L weeks for replenishment orders.

Although the details of a liability settlement differ from contract to contract, most supplier contracts track demand forecast updates coupled with a liability window to determine what amount of inventory that is already built up at a supplier will become a liability for IBM. The major difference between our model and classical single-item inventory models is the additional liability and pending-order costs incurred at the end of the planning horizon. Our overall objective is to minimize the total costs incurred over the planning horizon, which includes the per-period holding and delinquency costs and the end-of-horizon liability and pending-order costs. Hence, the model needs to determine the optimal reorder points R_t for each supplier replenishment hub according to the following criterion:

$$R_t = \min \sum_{t=1}^{T} hE[I_t] + dE[B_t] + qE[V_T]$$

where h and d are the unit holding and delinquency cost, q is the unit liability cost, and V_T is the inventory liability at the end of the planning horizon. We developed an efficient iterative algorithm to compute the optimal solution of the above inventory planning problem. See Reference 6 for additional details.

Our work with contract manufacturers and IBM component suppliers has led us to define a type of VMI policy which we call a (*min, max*) policy. Under a (*min, max*) policy, the OEM (other equipment manufacturer) sets a minimum inventory

level, *min*, and a maximum inventory level, *max*, that represent the lowest and highest inventory levels which IBM wishes the inventory level in the hub to reach. The values of *min* and *max* are tied to customer service levels and inventory liability risk and are derived from the optimal reorder points R_t that solve the optimization problem just described. In certain VMI agreements, a supplier might commit to pay a penalty for every unit of hub stock inventory that is less than *min* or more than *max*. In other cases such penalties might not be contractually enforced, but the (*min, max*) policy nevertheless provides incentives for the supplier to manage the inventory hub within the agreed-upon inventory levels.

The optimizer is invoked as a Web service from the MDBT observation model, which uses the (*min, max*) policy to compute dynamic thresholds and generate shortage and surplus alerts. The optimizer is also optionally invoked from the dashboard should users wish to override the default input parameters.

Dashboard

Dashboards geared toward BI clearly convey business objectives throughout the organization, report instant snapshots of the designated KPIs, and visualize real-time graphs and reports personalized for various roles in the organization. Drill-down capability allows users to move from high-level aggregate views down to operational details, allowing proactive management for better results and maximization of productivity.

With the hub-management solution, IBM System x management has an integrated supplier-inventory view that includes real-time inventory volumes along with stock, WIP, receipts, lead-time, forecasts, and supply-commit statements. Once the data is received from the suppliers and analyzed by the data integration component, MDBT and its associated inventory optimizer, the portal is accessible to users for viewing the KPIs and associated alerts. High-level aggregate views allow executives to visualize overall hub health and quickly narrow down, for example, the top 50 troubled hubs. Operational users (primarily from Procurement and Fulfillment) have the ability to personalize the hubs according to their job functions. Multiple views allow the users to filter and narrow down hubs based on location, supplier, part numbers, and commodities. Once the troubled hubs have been identified, a user can "drill down" and view the current and historical performance of the hub. An imbalance (supply shortage or surplus) in the hub signals the need for intelligent actions (known as conditioning actions) in order to improve the business performance. Such actions may touch on three aspects of the supply chain: demand, supply, and product offering:

1. *Supply conditioning*—A collection of supply-based actions, such as securing additional supply or rebalancing supply between geographic areas to resolve supply imbalances. ***Figure 7*** shows a dial display indicating a yellow surplus alert for the hub—its value was calculated based on the thresholds described earlier. The portal also allows the users to visualize the inventory positions of all the hubs for the same part across multiple geographic areas, thereby enabling the decision to rebalance across these areas or procure supply externally.
2. *Demand conditioning*—A collection of actions that affect demand in a desirable way. Examples of such actions are marketing promotions of a surplus product or reducing the price of an alternate product in order to move demand from a constrained product.
3. *Offering conditioning*—A collection of offering-based actions such as creating a new product model that uses either surplus components or substitutions from demand-constrained components.

These actions are enabled from the portal by users. The dashboard currently contains data spanning four worldwide locations, 200 suppliers, and 4500 part numbers, resulting in over 10,000 hubs currently being monitored.

We are currently using IBM WebSphere Portlet Factory[17] for the dashboard models, which helps create, customize, maintain, and deploy portlets. It provides a true model-driven environment, complemented with off-the-shelf builders and a factory designer that provides a reusable framework which can facilitate portlet development.

Business and IT benefits

The SaR infrastructure is currently being set up and configured in a service delivery center in Ehningen, Germany. The hub-management solution is the first service to be deployed and is expected to be in

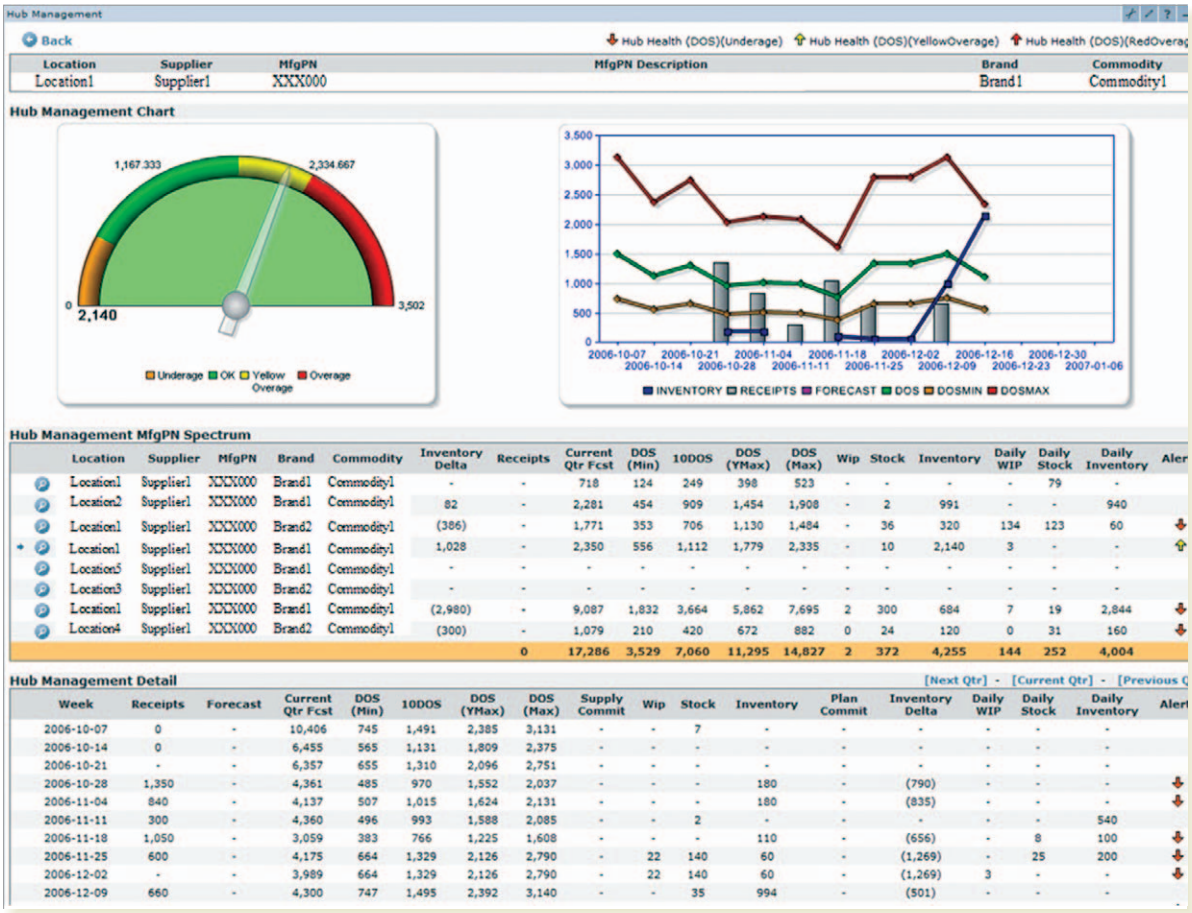

Figure 7
Hub management dashboard

production by the end of 2007. Initial user acceptance tests have begun and have resulted in very positive feedback. Some of the business benefits expected from the solution are

- to ensure supply availability across multiple tiers of the supply chain, enabling a demand-driven supply network,
- to reduce inventory levels through increased visibility and near real-time monitoring of the supplier hubs and reduce obsolescence cost by better utilization of available inventory in the supply chain,
- to improve operational efficiency by automating information exchange among partners and by supporting decision-making with integrated end-to-end supply and demand visibility,
- to ensure supplier compliance with contract and service level agreements through improved monitoring across all supply tiers, and
- to improve order serviceability and asset utilization through visibility into sources of supply.

In addition to the business benefit, we also expect significant IT benefits starting during system development and continuing through the application life cycle, such as the following:

- Lower total cost of ownership due to the managed software-as-a-service delivery model. We currently have five solutions in the pipeline whose deployment should result in solution owners sharing infrastructure costs. Savings are expected to increase as usage of the system increases.
- Auto-generation of code directly from the models should lead to significant cost savings. *Table 2* shows our experience to date for developing the model-driven hub management solution and serves as a comparison to the original legacy

Table 2 Benefits achieved from the SaR model-driven solution

	Duration	Expense	Size	Resources
Existing Legacy System	12–18 months	1M	>50,000 lines of code	20
SaR Model-Driven Solution	6 months	200K	300 lines of code (remaining was auto-generated)	4
Total Reduction/Savings	50%	80%	99.4%	80%

solution. Using a variety of model-driven technologies, we experienced an 80 percent reduction in development related expenses and 50 percent faster development time, and achieved the end-to-end solution with approximately 300 lines of manual code (the remaining code was auto-generated from the models).

- Reduced intervals between change requests, thereby providing flexibility to add custom changes for every customer.
- Simpler integration with external systems and services, such as connecting to custom data sources or integrating third-party analytical services.

CONCLUSION

MDD methods represent a leading-edge solution likely to replace traditional handcrafted and error-prone methods of business transformation. In this paper the transformation of the supply chain for the IBM System x product was described. The use of model-driven techniques benefited the hub-management project in several ways. The business models were expressed in a completely platform-independent fashion, thereby allowing for the highest level of reuse across multiple platforms. The models therefore became the core assets of the organization and allowed for a cost-effective, rapid, and reliable approach to business transformation. The integration of optimization analytics further improved business performance, as demonstrated by the integration of the inventory optimizer within the hub-management solution. Finally the shared-server environment provided by SaRs allowed for a multi-tenant, cost-effective infrastructure for hosting several business performance models.

The ultimate value of incorporating SaRs in the enterprise can be summed up as "business responsiveness"—the ability to quickly and effectively adapt in order to cope with various risks and opportunities. This approach was demonstrated by the transformation of the IBM System x supply chain.

ACKNOWLEDGMENTS

We thank Sarah Santo, Steve Molella, Anthony Humphrey, and Darrell Harrod and the Integrated Supply Chain organization for their guidance and business leadership in defining the hub-management process and data requirements. We thank Liangzhao Zeng, Michael Dikun, Pawan Chowdhary, and Brian Merzbach in the United States, and Jimmy Tan and Sam Wang at the IBM Software Development Laboratory in Taiwan, for their contribution to the development of the MDBT toolkit. We thank our colleagues at IBM Software Development Labs in India, and especially Prashant Yadav, Sam David, Somnath Mukhopadhyay, and Namrata Gupta, for testing and integrating the models discussed here.

CITED REFERENCES

1. S. Kapoor, K. Bhattacharya, S. Buckley, P. Chowdhary, M. Ettl, K. Katircioglu, E. Mauch, and L. Phillips, "A Technical Framework for Sense-and-Respond Business Management," *IBM Systems Journal* **44**, No. 1, 5–24 (2005).
2. *OMG Model Driven Architecture*, Object Management Group, http://www.omg.org/mda/.
3. P. Chowdhary, K. Bhaskaran, N. S. Caswell, H. Chang, T. Chao, S.-K. Chen, M. Dikun, H. Lei, J.-J. Jeng, S. Kapoor, C. A. Lang, G. Mihaila, I. Stanoi, and L. Zeng, "Model Driven Development for Business Performance Management," *IBM Systems Journal* **45**, No. 3, 587–605 (2006).
4. S. Kumaran, P. Bishop, T. Chao, P. Dhoolia, P. Jain, R. Jaluka, H. Ludwig, A. Moyer, and A. Nigam, "Using a Model-Driven Transformational Approach and Service-

Oriented Architecture for Service Delivery Management," *IBM Systems Journal* **46**, No. 3, 513–529 (2007).

5. *IBM System x*, IBM Corporation, http://www-03.ibm.com/systems/x/.
6. M. Ettl, Y. Lu, and M. Squillante, "Liability and Serviceability Trade-Offs in Vendor-Managed Inventory Systems," *Proceedings of the 2006 M&SOM Conference*, Georgia Institute of Technology, Atlanta, Georgia (June 2006)
7. *Business Performance Management*, IBM Corporation, http://www-306.ibm.com/software/info/topic/perform/partnerpage.html.
8. M. Santosus, "What You Don't Know About SaaS," *SearchCIO.com* (2006), http://searchcio.techtarget.com.
9. *Information On Demand: Leveraging the Business Value of Information*, IBM Corporation, http://www-306.ibm.com/software/data/information-on-demand/.
10. M. A. Rappa, "The Utility Business Model and the Future of Computing Services," *IBM Systems Journal*, **43**, No. 1, 32–42 (2004).
11. S. Kumaran, "Model-Driven Enterprise," *Proceedings of the Global Enterprise Application Integration (EAI) Summit 2004*, Banf, Canada (2004), pp. 166–180.
12. *IBM Rational Software Architect*, IBM Corporation, http://www-306.ibm.com/software/awdtools/architect/swarchitect/index.html.
13. "Magic Quadrant for Data Integration Tools," *Gartner RAS Core Research Note* G00141484, Gartner, Inc. (November 22, 2006), ftp://ftp.software.ibm.com/software/data/integration/analystreport/gartner2006.pdf.
14. *DB2 Data Warehouse Edition*, IBM Corporation, http://www-306.ibm.com/software/data/db2/dwe/.
15. *UML2 Profiles*, The Eclipse Foundation, http://www.eclipse.org/modeling/mdt/uml2/docs/articles/Introduction_to_UML2_Profiles/article.html.
16. L. Gong and D. Venditti, "DB2 Data Warehouse OLAP Services," developerWorks, IBM Corporation (2006), http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0606gong/.
17. *WebSphere Portlet Factory*, IBM Corporation, http://www-306.ibm.com/software/genservers/portletfactory/.

Accepted for publication May 21, 2007.
Published online September 25, 2007.

Shubir Kapoor
IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (shubirk@us.ibm.com). Mr. Kapoor is a senior engineer in the Analytic Models and Architecture department, Mathematical Sciences, which researches and develops techniques and systems for managing enterprise supply chains. He received his master's degree in computer science in 1997 from Pune University, India and joined IBM Research in 2000. Mr. Kapoor's nine years experience in the IT industry includes the architecture, design, and implementation of business systems, and using analytic techniques for the solution of business problems.

Blair Binney
IBM Integrated Supply Chain, Hopewell Junction, NY 12533 (bbinney@us.ibm.com). Mr. Binney is responsible for business transformation in the demand/supply planning and execution processes across IBM hardware brands. He manages and leads broad-based initiatives for supply-chain process transformation within IBM HW core Demand/Supply Planning and Supplier Integration processes. He has advocated and led the adoption of model-driven development technology in the IBM Integrated Supply Chain organization.

Steve Buckley
IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (sbuckley@us.ibm.com). Dr. Buckley, a research staff member since 1987 and a manager since 1995, is currently manager of the Analytic Models and Architecture department, Mathematical Sciences. His current activities are focused on sense-and-respond systems, an area in which his team has implemented prototypes for several IBM lines of business. Dr. Buckley received a Ph.D. degree in computer science from Massachusetts Institute of Technology (MIT) in 1987. He also received an M.S. degree in computer science from Pennsylvania State University in 1978, and a B.S. degree in applied mathematics and computer science from Florida State University in 1977.

Henry Chang
IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (hychang@us.ibm.com). Dr. Chang, a senior technical staff member and manager in the Business Informatics department, leads the research effort in business performance monitoring and management with product impact on the IBM Websphere® business integration suites and internal supply chain applications. His recent research interests include business monitoring, continuous process improvement, and event-based business collaboration.

Tian Chao
IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (tian@us.ibm.com). Ms. Chao is a senior software engineer in the Business Informatics department. She has a B.A. degree from National Taiwan University and an M.S. degree in computer science from Virginia Polytechnic Institute. Her research interests include the use of service-oriented computing technologies in business performance monitoring and management, security of business processes, and Web services.

Markus Ettl
IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (msettl@us.ibm.com). Dr. Ettl is a research staff member at the Watson Research Center. He received his doctoral degree in computer science in 1995 from Friedrich-Alexander University in Erlangen, Germany. Since joining IBM in 1995, he has focused on advanced research in supply chain management, an area in which he holds several patents. Dr. Ettl's current research interests lie in decision support for production systems and logistics networks and sense-and-respond business management for adaptive organizations. He received the INFORMS Franz Edelman Award in 1999.

E. Noel Luddy
IBM Enterprise Business Information CoE, Kissimmee, FL 34746 (eluddy@us.ibm.com). Mr. Luddy is an architect in Enterprise Business Information CoE. He leads the Sense and Respond Solution Initiative, which provides business performance management capability for all IBM divisions.

Rajesh Kumar Ravi
IBM India Software Labs, Koramangala Ring Road, Bangalore, 560071, India (rajeshkumar@in.ibm.com). Mr. Ravi, a staff software engineer, is currently on assignment to the Analytic

Models and Architecture department at the Thomas J. Watson Research Center in Yorktown Heights, N.Y., where he develops optimum structures to manage enterprise supply chains. He received a bachelor's degree in computer science in 2000 from a leading university in India and joined IBM in 2001. Mr. Ravi has significant work experience in the architecture, design, and implementation of software systems using rapid development tools and advanced techniques.

Jeaha Yang
IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598 (jeaha@us.ibm.com). Mr. Yang is an advisory software engineer in the Business Informatics department. He has a B.S. degree from Polytechnic University and an M.S. degree from Syracuse University. His recent work has involved business performance monitoring and management and integration of business processes. ■