Chapter 3.
Major Components and Architectural Design of Operational Business Intelligence System

3.1 Introduction

In this chapter, the major components of the proposed Operational BI system are identified from functional architecture presented in previous chapter.

Now-a-days Business intelligence (BI) becomes a major part of the decision making system for all business organizations. Traditional business intelligence systems are static, historic in nature and confined to strategic and tactical users only whereas modern business intelligence is dynamic, current, event driven and supports not only for strategic users but also tactical and even operational level users for their decision making. The modern business intelligence systems are popularly known as dynamic business intelligence or Operational BI. The purpose of Operational BI system not only provides operational intelligence to the operational users but also tactical and strategic users for their timely decision making. Thus, the components to be identified for the proposed system shall respond on real time or near real time.

The objective of this chapter is to present major components of Operational BI system. To model of three major components of the system those are business performance management, event monitoring and notification and incremental mining. Moreover, to present the enterprise architectural design framework of the proposed Operational BI system using Model View Controller Model 2 Architecture of J2EE.

*Part of this chapter has been


b) Presented orally and published in the proceeding of the 8th Int. Conf. on ICIPT 2013, April 1-3, 2013, Seoul, Republic of Korea.

c) Presented orally and published in the proceedings of the 2nd Int. Conf. on BIAKM 2013, April 17-19, 2013, IBS Hyderabad, India.

d) National Conf. on Recent Advances in Soft Computing and Knowledge Discovery (SCKD2012), Organized by S.V. University, January 19-21, 2012, Tirupati, India.

e) The 4th Int. Conf. on Electronics Computer Technology (ICECT 2012), April 6-8, 2012, Kanyakumari, India.
3.2 Related Work

Operational BI is an emerging concept in BI domain. There is no significant work has been reported in this area. Operational BI has gaining popularity in the recent past in all most all business domains because of increased business value in reduced action time.

David Hatch [8] was classified Operational BI into six different flavors which are 1) transactional BI with analysis and reporting, 2) real-time analytics with business rules applied to data as it is captured, 3) near real-time analytics business rules, 4) operational reporting, 5) business activity monitoring or business process monitoring, and 6) decision management based on business rules with integrated reporting and analytic applications.

Matteo Golfarelli et al. [23] were described purpose of Business Process Monitoring (BPM) in BI architecture that monitors time critical operational process which quantify the enterprise strategy.

Seufert et al. [19] were proposed architecture for real time analysis with the aim of reducing the action time and increasing the value of business intelligence that uses information and business integration infrastructure. The information infrastructure is responsible for managing data for business intelligence purpose and offers data analysis to decision makers. The business integration infrastructure is a sense and response system that communicates events via hugs with the internal and external business environment.

Event notification become a common module in most of the business applications which been studied and implemented widely [72], [73], [74], [75], [76].

Markus Aleksy et al. [73] described the design and implementation of a bridge that enables bi-directional messages transfer between applications based on CORBA Notification Service (CNS) and Java Message Service (JMS) architectures. This bridge provides a simple connection of both JMS and CNS systems that enables business to business (B2B) messaging. The bridge is divided into three software layers namely front end, backend and converter. The front end layer performs initialization of configuration files and starting of the connections between CNS and JMS. The backend layer is responsible for the connection between a concrete Channel, for message reception, selection of a suitable Converter, and for sending the converted message. Converter layer converts message from one service architecture to the other.
Gan Deng et al. [74] were developed Federated event service modeling language (FESML) using the Generic modeling environment (GME) which supports distributed real-time and embedded (DRE) systems. The artifacts of FESML include event consumers, event suppliers, event channels, CORBA Gateways, UDP Sender, UDP Receiver and Multicast ports. FESML can be used to model a federation of events channels using CORBA Gateways.

Srinivasan Ramani et al. [75] were studied suitability of the Common Object Request Broker Architecture (CORBA) event services for reliable message delivery mechanism. An application level scheme was implemented to build a reliable communication of CORBA’s event service that uses resynchronization of states. This application level scheme neither uses log nor retry mechanism of the event service nor rebinding of the restarted objects. The failure of the event service is detected with the help of a daemon that will ping the event service periodically. If the daemon notices that the event service has died, it restarts the event service with the help of resynchronization.

### 3.3 Time versus Business Value

Richard Hackathorn [22] was described the relationship between time and business value by considering the value-time curve as in figure 3.1.

![Figure 3.1. Action time versus business value](image)

At an initial time, a business event requiring a response occurs within some business process. At a later time, action is taken to respond to that event. The action time (or action distance) is the duration between the event and the action, while the business value is the net value of the business either lost (or gained) over this duration.
The action time is defined as the time interval between occurrence of business event and action taken. There are three latency components are involved in action time namely data latency, analysis latency and decision latency. The action time is the algebraic sum of data latency, analysis latency, decision latency and response latency.

The action time of the business intelligence system is given by an equation

\[ t_{\text{action time}} = t_{\text{data latency}} + t_{\text{analysis latency}} + t_{\text{decision latency}} \]  

(3.1)

a) Data latency is the time from the occurrence of the business event until the data is stored and ready for analysis.

b) Analysis latency is the time from the point when data is available for analysis to the time when information is generated out. It includes the time to determine root causes of business situations.

c) Decision latency is the time it takes from the delivery of the information to selecting a strategy in order to change the business environment. This type of latency mostly depends on the time the decision makers need to decide on the most appropriate actions for a response to the business environment.

From action distance curve of figure 3.1 the business value and action time are related as follows:

\[ \text{Business value} \propto \frac{1}{\text{Action time}} \]  

(3.2)

The cost of the information systems and business value are related as follows:

\[ \text{Cost of information system} \propto \text{business value} \]  

(3.3)

The improved business value in the system is directly proportional to the cost of the system.

Operational BI system speed is critical. Thus, action time and business value (or information) parameters are considered for designing an Operational BI system. In business, time is critical and information is money. Here information stands business value delivered by an information system. According to equation (3.1) business value delivered by an information system is inversely proportional to the action time. Thus, a business information system which delivers higher business value should have reduced action time. In order to reduce action time of the system, the time elapse between event occur and action complete is to be reduced. Thus, the components to be identified in the proposed system should be as small as action time as possible and
these components are tightly integrated. Therefore, the communication between various components within the system should be as small as possible and reacts as fast as possible to deliver information to the users of the system.

According equation (3.3), the cost of business information system will increase as the business value delivered by the system increases. In order to identifying the components of Operational BI system, business value and reduced action time are two important parameters at the same time the cost of the system should not increase. Therefore the modules to be identified in the proposed system shall have reuse of previous mining results.

### 3.4 Major Components of Operational BI System

The major components of the proposed system are identified from functional architecture of the system as discussed in Section 2.5 of Chapter 2. In addition, while identifying these components reduced latency and improved business value parameters are taken into consideration. Ivar Jacobson et al. [84] described modelling is a proven and well accepted engineering technique. The major components of Operational BI system is envisaged in figure 3.2. Figure 3.2(a) shows the generic view of these components whereas figure 3.2(b) shows Unified Modeling Language (UML) representation of the components.

![Figure 3.2. Major Components of Operational BI](image)

*Figure 3.2. Major Components of Operational BI*

Business performance monitoring facilitates to configure operational, business and process parameters of an organization. These configured parameters are dynamically computed and provides qualitative measure. The users can monitor organizational performance from time to time and access what is happening in the business in current time.
Event monitoring and notification becomes a common functional module in most of the business applications which is a combination of software and hardware. This delivers right message to right person in right time on near real time/real time basis. Once an event is occurred in the system, monitors and then generates an alert if predefined configured parameter limits exceeds. The generated alert will contain a message that conveys brief description of an event. This system will deliver the messages in one or more forms of SMS, MMS, email and voice alert to the users.

Operational analytics contains two sub-components namely incremental mining and other analytics. Incremental mining extracts knowledge from incremental databases or incremental process databases which are formed from operational data sources and process database which provides decision making information in current time. This uses previous mining results inorder to provide updated knowledge to avoid re-learning of rules from the old database. Other analytics component includes text mining, forecasting, predictive analytics and optimization algorithms.

Operational reporting provides to view all operational information in the forms of reports to the user in current time as well as historical. This reporting tool will help the users for faster decision making as compared to traditional reporting business intelligence tools. The report engine includes customizable reports on self-service basis that can present high-level findings as well as enable a user to drill down to find specific details extracted from in the form of patterns, relations among the patterns, relation among the patterns with variables in the system. This module includes standard report templates that provide the user to create customizable reports. The reports module consist infrastructure for strategic reporting to serve the strategic and tactical management as well as operational reporting for low level decisions of business operations on day to day basis.

Operational reporting module includes knowledge management tools to present available knowledge in simple way to the users. Most commonly the following two popular approaches are employed for operational reports by querying transaction system directly or they can off-load the transaction data and query the data separately as described Eckerson [13]. This module includes data visualization tools to present data to the user in simple and understandable form.

Portal works as a collaboration tool between online analytical processing (OLTA) and online transaction processing systems (OLTP). This acts as information
dissemination tool. In addition, this acts as single entry points for resource access to the uses of the system.

3.4.1 Conceptual Architecture of Operational BI System

The conceptual architecture of the proposed system is shown in figure 3.3 from the identified major components.

![Figure 3.3. Conceptual architecture of Operational BI](image)

The inputs to the system are source systems which are nothing but various operational data sources. The output of the system consists of operational, tactical and strategic users. The major components constitute the core part of the system.

3.4.2 Improved Business Value

The identified components of Operational BI system are based on business value and reduced action time parameters. According to equation 3.1 the business value increases when action time reduces. In order to reduce action time of the system, the components are further decomposed in such a way that they work more efficiently and integrate to other components in the system tightly.

Business Performance Management provides an efficient monitoring of the operational business process and helps to provide in managing the key performances of an organization. Business is agile in nature. Hence, the system is flexible to adopt changes over time. Thus, the proposed business rule engine provides not only configuration of rules but also flexible to change from time to time. Thus, the proposed business performance management component provides monitoring of business performance in current time as opposed to traditional BI system which in turn improves business value.
Event monitoring and notification is one of the important components of Operational BI system that sends alerts to the users on real time basis about business performance parameters configured in the system for their immediate decision. In addition, event analytics engine provides extracted knowledge from the events database to the user for timely information. Thus, the proposed event monitoring and notification component provides timely information to the users but also helps for timely decision making which in turn improves value to the business.

Operational analytics is one of the most important components in the over system which has two sub-components namely incremental mining and other analytics. Incremental mining provides extracted information in current time that uses previous mining results to provide updated mining results. Thus, incremental mining results will help to operational users for their immediate decision making whereas update mining results help for timely decision making to tactical and operational users. Other analytics include text mining, forecasting, predictive analytics and optimization which in turn provides additional analytical information to the user. Thus, the proposed operational analytics component provides timely information to the users and also improves business value.

Operational reporting component provides dynamic reports which can be linked to historical reports to forecast the trend analysis. This module not only provides the timely update of information but also trend analysis to the users through which the decision making become much faster which in turn improves business value.

The portal integrates OLAP as well as OLTP systems which act as single source of information dissemination.

The above identified major components of Operational BI system reduces the action time and increases business value greatly to the organization. Thus, the effective use of Operational BI system not only improves the efficiency of an organization but also higher business value.

3.5 Business Performance Management

Every organization is associated in extending product or services to the external world that associate a business process or processes. Here the term business process means sequence of activities to complete a service. The business process may include simple or complex set of activities. In the competitive business environment performance of an organization is to be measured from time to time. In order to
measure performance of an organization there is a great need of tools to help to understand, manage and improve what organizations do in current time. Thus, the proposed business performance management sub-system will help to monitor and measure organizational performance in current time. The component diagram of business performance management is shown in figure 3.4.

![Component Diagram](image)

**Figure 3.4. Components of business performance management sub-system**

Performance measures quantitatively which tells something important about product and services of an organization. This includes the measurement of business, operational and process parameters of organization. Business performance management provides key operational metrics to BI and BI captures these metrics on real time basis from process database and event databases. Further, a relationship between these metrics can be extended from strategic operational level. Prasanna Keny et al. [26] was described Operational BI applications are closely integrate with operational data and operational processes.

*Process database* facilitates configuration of business rule definitions of an organization which include configuration of business, operational and process parameters. These parameters can be configured as and when changes from time to time. In addition, this holds extracted process data from operational sources which will act a input to the performance management.

*Process analytics* extracts information from business process data which is obtained from the operational sources and process databases.

*Business rules engine* facilitates configuration of business rule definitions and change of these configurable values from time to time. More commonly, Operational BI applications generally cause changes to operational procedures or processes. Modern business is dynamic in nature which demands continuous change in business rule definitions. Over a period of time in any business, the existing business rules become obsolete and new business rules may come or possibility need to modify the existing business rules. So, Operational BI systems are agile to adopt these changes
in order to highly optimize the evolved information from operational system. Hence, there is a great need of business rule engine for Operational BI system to make the changes of business rules dynamically without affecting system. In addition, this sub-component provides SLAs/ KPIs management that include configuration of business, process and operational parameters of an organization.

*Dynamic parameter computation* which computes the configured parameters defined in business rules engine from the process data. The performance of configured parameters is compute dynamically of on a continuous basis and these values are feed to business monitor for display. Similarly, the output of computation engine is feed to event monitoring and notification sub-system for delivery of messages to the users.

*Process monitor* is to measure the performance of configured parameters qualitatively in the form of dashboard to the users of the system. This provides the qualitative measure of configured SLA/ KPIs in the business rule engine against the computed values.

The functional architecture of business performance management is shown in figure 3.5. The inputs are process database which is extracted from operational data sources that consist of process data as well as log files information.

![Figure 3.5. Functional architecture of business process monitoring](image)

The parameter computation engine computes aggregate value from the process database which is validated with KPIs/SLAs parameters, configured parameters and business rules defined in in the system. The computed results are stored in files which are resided in main memory. These files are updated dynamically from time to time. The computed results are sent to monitor module which in turn feeds to data visualization module. The data visualization module provides display of monitor data in the form of various reports and dashboards to the users.
3.5.1 Check Algorithm

The Check algorithm is one of the most important functional blocks of the proposed business performance management sub-system. The inputs to the Check algorithm are dynamically computed business performance parameters from the process database and the configured parameters by the user for measurement. The Check algorithm uses associative search for comparison between threshold limits of configured and pre-computed parameter values which are defined in configuration files, KPIs/SLAs and business rule modules. The Check algorithm generates output if there change in these parameter values. The output of check algorithm is a message which contains the parameter name and their quantity which is feed to event monitoring and notification and as well as monitor modules simultaneously. The Check algorithm pseudo code of is envisaged in algorithm 3.1.

**Algorithm 3.1. Check algorithm**

| Inputs: Configured parameters, business rules, SLAs/KPIs |
| Output: Alert message |
| 1. READ data from config file |
| 2. READ operational computed data |
| 3. FOR EACH DO |
| 4. Compares each config parameter value against operational computed data |
| 5. IF (computed parameter > config parameters) |
| 6. BEGIN |
| 7. Generate Alers () |
| 8. Feed data to Monitor() |
| 9. END |
| 10. END IF |
| 11. END FOR |

3.6 Event Monitoring and Notification

The component diagram of event monitoring and notification sub-system is shown in figure 3.6 which is one of the most important components of Operational BI.

![Figure 3.6. Components of Event monitoring and notification sub-system](image-url)
Message queue consists of set of messages. Inter object communication uses a standard message format and all such generated messages are stored in a queue.

Events monitor continuously monitors the configured parameters and business rules on real time basis from the incoming messages.

Event analytics module consists of set of mining algorithms which extracts information from log files, process database and event databases.

Message formation fabricates message dynamically by selecting suitable message template which is based on event type generated by an event monitor. Event recording records the events generated by event monitor in event database. Similarly, all transmitted messages are also stored in message database.

Message delivery this delivers event information to the users of the system. The message may be one or more or combination of SMS, MMS and E-mail.

3.6.1 System Architecture of Event Monitoring and Notification

The system architecture of event monitoring and notification sub-system is envisaged in figure 3.7 which consists of three major modules such as event engine, system services and support resources. The inputs to the system are various messages. The users of the system can access dashboard, various formats of graphs, and of alerts messages - SMS, MMS, and Emails.

![Figure 3.7. System architecture of Event monitoring and notification sub-system](image)
**System services** help to control and manage the functionality of overall system. Scheduler will run as per the predefined time intervals to perform the specified activities. Admin services deals management of system service.

**Support services** include set of resources required for system configuration files, business rules, message templates and message database. Scheduler will run as per the predefined time intervals to perform the specified activities. The set of resources required for system configuration and management are collectively known as support services that includes configuration files, message templates and scheduler services.

**Event engine** receives input messages from various functional blocks of Operational BI system and are put into message queue. The messages are pre-defined XML templates. Each message contains header and body. The typical message header contains attributes such as template id, source id, message id and the body contains contents of event information.

The functionality of message processing module is to process the message serially which are received from the message queue. During message processing parsing of message is the main step. In addition header part of the message is removed and then forms body of the message into a string. The processed message is simultaneously feed to the event analytics as well as event monitoring blocks. The output of event monitoring is provided to message formulation module.

Figure 3.8 shows the functional architecture of message formulation. The input is event notification and the output is message delivery. The working is envisaged in provided below.

**Figure 3.8. Functional architecture of message formation**

**Dynamic message formulation** fabricates message dynamically by selecting suitable message template.

**XSLT processor** converts the dynamically formed messages into a standard presentable form. XSLT stands for Extensible Style sheet Language, is a style sheet
language for XML documents that XSLT converts the incoming messages into target user presentable form. In order to present multiple views of alerts to the target user separate XSLT style sheets are to be required. The output of XSLT will be given to the message dispatcher.

*Message dispatcher* can have a queue management internally. All the messages that are generated will be put in queue. Dispatcher will dispatch messages from queue based on priority of the alert. These alert messages will push these alerts or messages to a web-based persistent communications channel and finally deliver to the right decision maker to notify the issue. The various types of alert messages are Email, SMS, MMS, audio, video clip or even log file information.

*Event dashboard* module of figure 3.7 provides monitoring of the events to the users which include line graphs, bar graphs, pie chart and dashboards.

*Message delivery* unit will generate alerts to the subscription users in the form of SMS, MMS and Emails.

*User of event monitoring* the users of the event monitoring and notification system are mainly classified into three major categories such as admin, end user and executives. Admin user manages the administration of the application by managing system services; end users will receive alert notifications based on changes in the threshold levels of configured parameters and executive users monitors the operational parameters on daily basis with the help of reports and dashboard.

### 3.6.2 Message Templates

In this section, the structure and XML formats of input and output messages of notification sub-system are presented. The structure of the message provides various elements and their data structure whereas the message format is a pre-defined XML template. Each message will have DTD that defines the schema of the message. The message template has two sections namely header and body. The header section contains elements which describe the properties of the message whereas body of the message section contains the actual information.

The structure of input and output messages are shown in figure 3.9 and 3.10 respectively. The typical elements of the input message header are message identification (msg_id), module identification from which it was send (module_id), template identification (templ_id), message generation time (timestamp_id) and message type (msg_type). The default type of msg_type is text.
The body of input message contains an attribute known as message size (msg_size) and followed by actual message (msg).

![Figure 3.9. Structure of input message](image)

![Figure 3.10. Structure of output message](image)

The structure of output message is identical to input message. However, the header section of output message contains few more elements that describes message properties namely priority of the message e.g. low, medium and high (msg_priority), type of the message e.g. SMS, Email, audio clip, synchronization flag 0 indicates no sync and 1 indicates sync (msg_sync), and model: pull model / push model/ hybrid model (model).

Input and output message formats are shown in figure 3.11 and 3.12. The root element of input message is in_msg whereas output message is out_msg. The body of the message contains the actual data.

```xml
<in_message>
  <head>
    <msg_id> </msg_id>
    <module_id> </module_id>
    <templ_id> </templ_id>
    <timestamp_id> </timestamp_id>
  </head>
  <body>
    <msg_size> </msg_size>
    <msg> </msg>
  </body>
</in_message>
```

```xml
<out_message>
  <head>
    <msg_id> </msg_id>
    <templ_id> </templ_id>
    <timestamp_id> </timestamp_id>
    <msg_priority> </msg_priority>
    <msg_type> </msg_type>
    <msg_sync> </msg_sync>
    <model> </model>
  </head>
  <body>
    <msg_size> </msg_size>
    <msg> </msg>
  </body>
</out_message>
```

![Figure 3.11. input message format](image)

![Figure 3.12. input message format](image)
3.6.3 Advantages of Event Monitoring and Notification System

Alert monitoring and notification engine has become a common part of many applications on day to day life. Few applications of the proposed event monitoring system are provided below:

- Banking and finance - debit/credit card transaction details, check clearances, funds transfer alerts, check clearness, bill payment and fraud detection.
- Real-time information of stock market.
- News alerts
- Supply chain management: inventory level alerts, reorder levels of stock.
- In travel: ticket booking confirmation, ticket cancellation, travel schedule notification, time table notification and delays in operation.
- Environmental: Earth quakes, disasters, weather, rainfall, temperature, humidity and snowfall.
- Traffic: road clearances, accidents and traffic violation.
- Avionics: flight timings.
- Telecommunications
- Enterprise: SLAs/ KPAs monitoring, sales notifications, servicing alert

3.7 Operational Analytics

The component diagram of operational analytics is shown in figure 3.13 which is one of most important component in the proposed system. This contains several analytical engines that suits to both operational as well as traditional BI system.

![Figure 3.13. Sub-components of Operational Analytics of Operational BI](image)
An Operational BI system can be considered as hybrid BI system because it covers operational and analytics functionalities together. This provides complete view business intelligence to the users of an organization for their operational, tactical and strategic decision making. The major functionality of analytics layer is to extract knowledge from the operational data and operational business process into incremental database. Incremental mining algorithms work as per the predefined schedule and extract knowledge from incremental database. This includes various mining algorithms such as association, classification, clustering, classification, and prediction.

Other analytics include text mining, forecasting, predictive analytics and optimization algorithms. Text mining algorithms will discovers and extract meaningful patterns and their relationships. Forecasting algorithms or methods will analyze and forecast process which takes over a period of time. Predictive analytics will provide predictive information to the users. Optimization algorithms will provide best results from the identified scenario. The final output of the Operational analytics modules is extracted patterns or results which will be made available to the users from operational reporting module.

3.7.1 Incremental Mining System (IMS)

Data mining system is a collection of tools and technology that helps to analyze datasets, find patterns and relationships in the data. The tools will find patterns in the data with the help of algorithms whereas technology helps to build efficient tools. Figure 3.14 shows typical Incremental Mining System (IMS). The major functional blocks of IMS system are: operational data sources, incremental database, data warehouse, knowledge database, metadata, incremental mining and mining algorithms, reporting and query tools and user access layer.

![Image: Typical incremental mining system](image-url)
Incremental data mining or incremental mining system (IMS) will help to analyze more importantly operational data that provides the corresponding changes in current and as well as historical data in terms of insertion, deletion and update operations. The users of an IMS will get timely update of knowledge for their decision making. The IMS will also support for ongoing maintenance of data warehouse or business intelligence applications. The objective of incremental mining were defined by Chris et al. [143] which avoids re-learning of rules from the old data and utilize knowledge that has already been discovered.

**Operational Data Sources (ODB):** This includes various data sources and applications in the business organization. These data sources will contain day to day business transactions that generally include ERP, CRM, SCM and legacy.

**Incremental Database (IDB):** This is the data obtained from the operational systems that is transaction databases in a given window of time interval. The data collection time is as small as days, hours or even minutes that depends on the type of business and business value to be measured accordingly scheduler extracts data from source systems and extracts knowledge from incremental databases. It is assumed that incremental database is available as one of the inputs to the proposed algorithm. The cost of maintaining the rules can be reduced considerably provided the type of incremental database is known that can be visualized as three types according to Sarada et al. [128] for incremental mining of association rules. (a) The incremental database can be considered as a sample of the original database. In this case, there are no significant differences between the patterns in the original and the updated databases. (b) The incremental database has more or less similar patterns to that of the original database. Essentially the original patterns may still exist but there may be a few new patterns. (c) The incremental database may not necessarily be a good sample of the original database. The patterns found in the incremental and the original database may be entirely different.

**Data warehouse database (WDB)** is a central repository of data which is created by integrating data from multiple disparate sources. This stores current as well as historical data. It is single source of truth of an organization. This database is used for reporting and data analysis purposes for creating trend reports for senior management reporting that includes annual and quarterly comparisons which is used for strategic and tactical level decision making.
**Updated Database (UDB)** is a combination of original database and incremental database.

**Knowledge Database (KDB)** consists of extracted knowledge from incremental databases as well as historical databases. Mining algorithms have to run first time on original database to extract knowledge. This extracted knowledge contains the frequent patterns which are known as previous mining results or knowledge database of original database. The proposed algorithm has to run on the incremental database to extract knowledge which is known as incremental knowledge or incremental mining results. These incremental mining results are updated with previous mining results which form as an updated knowledge database. Thus knowledge database contains multiple versions of mining results (or extracted patterns) of incremental and updated databases. So, the users of an IMS can access not only incremental mining results but also updated mining results which provide good visibility of frequent patterns in both short and longer time periods. This in turn gives more business value to the users of the system for their effective decision making.

**Incremental Mining and Mining Algorithms** contain various mining algorithms as well as incremental mining algorithms. The proposed algorithm will reside in this functional block which extracts knowledge from incremental database that uses previously mining results. The extracted knowledge is updated with already available knowledge or previous mining results. So the users of the system can access both incremental knowledge as well as updated knowledge for all the levels of decision making.

**Query and Reporting Tools** module provides ad-hoc querying and reporting functionalities. A good IMS system should include efficient query and reporting modules that include functionalities: ad-hoc querying and reporting dimensional and non-dimensional data, dash boards, and self-configurable reporting. The end user can access frequent patterns from the updated knowledge database with the help of reporting and query tool. These tools include functionalities for extract, sort, summarize, and present selected data to the user in suitable visual form. In addition to this these tools will provide an additional feature including alerts for presenting the right information from incremental and as well as updated knowledge to the right people at the right time for right decision making.

**Metadata** module is one of the most important functional blocks in IMS which is similar to data warehouse and business intelligence systems. This describes data of
data and is often used to control the handling of data and describes: Rules, Transformations, Aggregations, and Mappings.

*Incremental data mining* system will help to analyze operational data and the corresponding changes in the historical data (in terms of insertion, deletion and update operations) and support ongoing maintenance of data warehouse or business intelligence applications. Nittaya Kerdprasaop et al. [41] defined the objective of incremental mining as to avoid re-learning of rules from the old data and utilize knowledge that has already been discovered. This greatly reduces analysis latency of action time which in turn brings value to the business. The typical process of incremental mining system is shown in figure 3.15.

![Figure 3.15. Typical process flow of incremental mining](image)

The proposed IMS provides incremental mining results as well as previous mining results. Previous mining results will be updated with recently extracted mining results and this forms an updated mining results.

### 3.8 Architectural Design Framework of Operational BI

In this section, the design of Operational BI system is presented using Model View Controller (MVC) Model 2 architectural framework of J2EE. MVC is a software architecture pattern which supports multi-layer architectural, event based and real time applications, seamless integration between multiple layers, large number of user access, highly scalable and support enterprise applications. The MVC architecture has its roots in Smalltalk, where it was originally applied to map the traditional input, processing, and output tasks to the graphical user interaction model as mentioned by Glenn E. Krasner et al. [85].

#### 3.8.1 Multi-tier Architecture

Modern enterprise applications are designed using several components connected to one another; each component provides a specific functionality. Components that perform similar functionality are grouped into layers. These layers are further organized as a stack. The proposed architecture for Operational BI is considered as stack of tiers. A tier is a logical representation of concerns in the system. Each tier is assigned its unique responsibility in the system. Each tier is logically separated from another and is loosely coupled with the adjacent tier.
The multi-tiered architecture of the proposed Operational BI system is shown in figure 3.16 which consists of five layers known as resource, integration, business, presentation and client tier.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client Tier</td>
<td>Application client, Device and GUI</td>
</tr>
<tr>
<td>Presentation Tier</td>
<td>Java Server Pages, Servlets and UI elements</td>
</tr>
<tr>
<td>Business Tier</td>
<td>Workflow, Business rule engine and BusinessObjects</td>
</tr>
<tr>
<td>Integration Tier</td>
<td>JMS, JDBC, SOA, Connectors and Legacy</td>
</tr>
<tr>
<td>Resource Tier</td>
<td>Databases, external systems and legacy applications</td>
</tr>
</tbody>
</table>

**Figure 3.16. Tiered architecture of Operational BI**

*Client tier* represents all devices or system clients accessing the system resources. A client can be a Web browser, a Java application, or a device or a mobile and Graphical User Interface (GUI).

*Presentation tier* encapsulates all presentation logic required to service the clients that access the system. The presentation tier intercepts the client’s requests, controls access to business services, construct the responses and finally deliver the response to the clients. The presentation layer contains Servlets and JSPs that produce User Interface (UI) elements.

*Business tier* provides the business services required by the client. This tier contains the business data and business logic. Nadir Gulzar [86] was described that most business processing for the application is centralized in business tier. Mostly workflow, Java bean components or Enterprise beans are used as business objects. Bodoff *et al.* [83] were described Java bean components are Java classes that can be easily reused and composed together into applications. JSP technology directly supports using JavaBeans components with JSP language elements. You can easily create and initialize beans and get and set the values of their properties.

*Integration tier* includes the components such as JDBC, JMS, SOA, J2EE connectors, or middleware or any legacy application. Resource tier contains the business data and external data or application sources, legacy systems and data warehouse system. The design of data warehouse system is out of scope of this chapter and as this thesis and is considered as an external system to the proposed architecture. Integration tier is responsible for communicating with external
resources and systems. The business tier is coupled with the integration tier whenever the business objects require data or services that reside in the resource tier.

**Resource tier** contains database servers, legacy systems, other transaction application and ftp servers and directory servers.

According to IEEE 1471-2000 standard [95] layered frameworks and models for enterprise architecture have proved useful because layering has the advantage of defining contained, non-overlapping partitions of the environment. The components in a higher layer use the services of components in a lower layer. A component in a given layer will generally use the functionality of other components in its own layer or the layers below it. The major functional blocks of enterprise architecture are mapped to individual layers known as data sources, data layer, business layer, service layer, presentation layer and user layer. Layered architecture focuses on the grouping of related functionality within an application into distinct layers that are stacked vertically on top of each other. Functionality within each layer is related by a common role or responsibility. These layers are loosely coupled to each other. The layered architecture provides reuse of the functionality; improve performance, scalability and maintainability of the system which also provides security to the proposed system.

### 3.8.2 Model View Controller (MVC) Model 2 Architecture

According to IEEE Standard 1471-2000 [95], architecture is defined as “the fundamental organization of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution”. MVC is one the software architectural design patterns which support enterprise applications. Figure 3.17. shows the MVC Model 2 architecture.

![Figure 3.17. MVC Mode – 2 Architecture](image)

Figure 3.17. MVC Mode – 2 Architecture
The client will access the application through browser. All user requests are handled via controller and responses through JSP. Internally, model (Java bean) will connect to the enterprise servers/databases inorder to access the required data.

Model encapsulates the core data and functionality. The model represents enterprise data and the business rules that govern access to and updates of this data.

View encapsulates the presentation of the data. The view renders the contents of a model. It accesses enterprise data through the model and specifies how that data should be presented. It is the view's responsibility to maintain consistency in its presentation when the model changes. This can be achieved either by using a push model or pull model. In push model the view registers itself with the model for change notifications whereas in pull model the view is responsible for calling the model when it needs to retrieve the most current data.

Controller accepts inputs from the user and makes request from the model for the data to provide a new view. The controller translates interactions with the view into actions to be performed by the model. In a stand-alone GUI client, user interactions could be button clicks or menu selections, whereas in a Web application, they appear as GET and POST HTTP requests. The actions performed by the model include activating business processes or changing the state of the model. Based on the user interactions and the outcome of the model actions, the controller responds by selecting an appropriate view.

The request and response flow of the system is envisaged as follows:

1) User requests are directed to the controller servlet.

2) The controller servlet accesses required data and builds the model, possibly delegating the processing to helper classes.

3) The controller servlet (or the appropriate sub-ordinate task) selects and passes control to the appropriate JSP responsible for presenting the view.

4) The view page is presented to the requesting user.

5) The user interacts with the controller servlet (via the view) to enter/modify data, traverse through results.
3.8.3 Sequence Diagram

The sequence diagram of MVC with request dispatcher (or manager) is shown in figure 3.18 that shows interaction between various objects such as user, request manager, controller, JSP and bean. Ivar Jacobson et al. [84] were described sequence diagram is a kind of interaction diagram which shows the dynamic view of a system.

Figure 3.18. Sequence diagram for request manager

3.8.4 Generic Objects

The generic objects and their flow of the proposed web based Operational BI system is shown in figure 3.19.

Figure 3.19. Generic objects of Operational BI system

Java Server Pages Model -2 is Sun’s attempt to wrap JSP within the MVC paradigm by Krasner et al. [85]. The Java Server Page Model 2 architecture has front
controller, data access and application logic, Service-To-Worker and Dispatcher View, Intercepting Filter, Value List Handler and Data Access Objects (DAOs).

*Front Controller* is a Servlet that acts as the centralized entry point into a Web application, managing request processing, performing authentication and authorization services, and ultimately selecting the appropriate view.

*Data access and application logic* contain entirely within the controller servlet and its helper classes. The controller servlet (or the helper class) should select the appropriate JSP page and transfer control to that page object based on the request parameters, state and session information. One of the major advances that come with JSP Model 2 is Sun’s specification of the Java Standard Tag Library (JSTL). It specifies the standard set of tags for iteration, conditional processing, database access and many other formatting functions. The guidelines associated with JSP Model 2, Sun also provided a set of blueprints for building application using the MVC paradigm and these blueprints renamed the J2EE Core Patterns.

*Service-To-Worker and Dispatcher View* strategies for MVC application where the front controller module defers processing to a dispatcher that is selected based on the request context. In the Dispatcher View pattern, the dispatcher performs static processing to select the ultimate presentation view. In the Service-To-Worker pattern, the dispatcher’s processing is more dynamic, translating logical task names into concrete task module references, and allowing tasks to perform complex processing that determines the ultimate presentation view.

*Intercepting Filter* allows for pluggable filters to be inserted in the “request pipeline” to perform pre and post processing of incoming requests and outgoing responses. These filters can perform common services required for all or most application tasks, including authentication and logging.

*Value List Handler* is a mechanism for caching results from database queries, presenting discrete subsets of those results, and providing iterative traversal through the sequence of subsets.

*Data Access Object* (DAO) is the centralized mechanism for abstracting and encapsulating access to complex data sources, including relational databases. The DAO acts as an adapter, allowing the external interface to remain constant even when the structure of the underlying data sources changes.
3.8.5 System Architecture

The system architecture of the proposed Operational BI is shown in figure 3.20 which consists of database resources, application server, web server with portal framework and users.

**Figure 3.20. System architecture of Operational BI**

*Data resource layer* contains all the legacy application data, data sources, data warehouse and Metadata.

*Application server* is a J2EE compliance application server.

*OBI engines* with admin services, user management and other resources code is deployed on Application server. Operational BI engines includes ETL, Real Time ETL, business rules, OLAP, data compression, reporting, dashboards, alert notification and monitoring, analytics and SQL streaming. In addition to this admin services includes managing various business services available to the user, user management and message templates.

*Web server* contains various UI components, collaborative framework including email, SMS, MMS and as well as portal. All the users of the system will access the resources through portal. Portal acts as single entry point to access the applications resources available.

*Users* can access the resources from portal which include operational, tactical and strategic of an organization.
3.8.6 Deployment Architecture

Figure 3.21 shows deployment diagram of the proposed Operational BI system. Ivar Jacobson et al. [84] described deployment diagram shows the configuration of run time processing nodes and the components that live on them which model the static view of a system.

![Deployment Architecture Diagram]

**Figure 3.21. Deployment architecture of Operational BI**

*Client tier* is loaded with web browser through which Operational BI application can be accessible. *Web tier* consists of UI components and presentation logic components. These UI components facilitate the users to access the application in predefined format for better presentation. *Application tier* consists of various components consists of business logic, alert engine, data access, service layer, workflow and other support components. *The data tier* consists of RDBS, OLAP and OBI engines. *OBI engines* include business rules, data compression, and reports components. *External system* consists of different sources and legacy applications.

3.8.7 Advantages of the Proposed Architectural Design Framework

The proposed design of Operational BI system uses MVC 2 model that removes page centric property of MVC 1 and separates presentation logic and application logic. The controller receives all requests for the application and is responsible for taking appropriate action for each request. It is well proven and industry accepted architecture. The proposed design of the system has the following advantages over monolithic architecture:

- Clear separation between presentation logic and business logic: - Each object in MVC has distinct responsibilities. All objects and classes are independent of each other. So change in one class does not require alternation in other classes.
- Multiple views using the same model: - The separation of model and view allows multiple views to use the same model. This is not only facilitates easier implementation of enterprise model but also easier to test, and maintaining of the enterprise application.

- Efficient modularity: - This architecture highly supports modular development of the application either by the use of different controllers for each module or single controller with different action classes.

- Easier support for new types of clients:-This model is easier to support for new types of clients. We need to a view and controller for each type of client and wire them into the existing enterprise model.

- Support for web applications: This model is often seen in web applications.

- High scalability: - Controllers and views can grow as the model grows; and older versions of the views and controllers can still be used as long as a common interface is maintained.

- This model supports easier maintenance of the code and future improvements of the application.

### 3.9 Conclusion

In this chapter the major components of Operational BI are presented from functional architecture, business value and reduced action time parameters. The conceptual architecture of the proposed system is envisaged from the identified components. The business performance management sub-system is presented and explained the monitoring of operational parameters on real time basis with the help of the proposed Check algorithm. The working of event monitoring and notification sub-systems is presented which generates alerts on real time basis. The advantages of alert notification system are presented. In addition, the working of incremental mining system is presented which is a major part of operational analytics.

The architectural design of Operational BI system is presented using MVC Model 2 architecture of J2EE. The multi-tiered architecture of the proposed system is presented with associated components in each tier. The sequence diagram and generic object flow of the proposed Operational BI system are presented. In addition, the system and deployment architectures are presented. Finally, the advantages of the proposed architectural design framework are envisaged.