CHAPTER 4

BUSINESS LOGIC EVALUATION MODEL

4.1. INTRODUCTION

The Business Logic Evaluation Model enables the business analysts to make changes efficiently by themselves without the aid of the IT developers by ensuring high degree of automation and run time support and thus helps the analysts to meet their competitors. The model focuses on change management, run time management and proactive management with the help of the Finite State Machine acting as the backbone. Run time support is guaranteed to the analysts by evaluating the properties like computability, traceability, accessibility, configurability, dependency and interoperability by the property evaluator which provides better run time support to the analysts making them aware of the nature of the schema by informing them whether the code is manageable or not. Change management is efficiently done by evaluating the Change factors like order of execution, mapping function, similarity measure, schema validation, time complexity, policy enforcement, space complexity and code consistency before and after the change is made and notifying the variations to the analyst if necessary. Dynamic service integration which might be needed based on the demands is also handled as a part of change management using the connectors and cases where there arises the need for combining, substituting the services can be handled using the methods of dynamic service integration viz. union, composition, substitution and reduction. Proactive management uses a prediction engine to deal with the decidability, manageability, impact analysis, prediction analysis and growth rate analysis and predicts the risks involved in committing the change and the correctness of the change.

4.2 BUSINESS LOGIC EVALUATION MODEL

The Business Logic Evaluation Model enables efficient change management by ensuring high degree of automation and run time support and thus helps the analysts to meet their competitors without depending on the developers. The model focuses on change management, run time management and proactive management with the help of the Finite State Machine acting as the backbone. The different phases in this model are elucidated below.
Figure 4.1 The Phase Diagram of Business Logic Evaluation Model
4.2.1 Phase I of Business Logic Evaluation Model

With reference to the diagram depicted in figure 4.1, the phase I of the phase diagram of the experimentation framework consists of a Request Handler and a Source Controller. The Request Handler analyzes the incoming change request and prepares plan and schedule for managing the request. The Change Request is the incoming request for change which comprised of operation, resource and condition. The Request Analysis takes care of analysing the change request with respect to domain and context in association with planner and scheduler. It analyzes and determines the change information such as where and when the request has originated, who has originated the request, what type of change is requested, which business logic is involved and what operation needs to be performed.

The Planner and Scheduler plan about handling the request by diagnosing the nature of the request if it is a new one, if a similar request has been already handled or if the request is an already existing one. It also checks how often the changes occur and what the priority set for the change is. On account of a new change request, it proceeds to the next level with domain and context information. Emergency checks are also made to know whether the change is a critical one or not.

The Planner and Scheduler then process the new requests further by handing them over to the Source Controller. The job of a Source Controller is to extract and provide the source code. Change Plan is exclusively meant for similar requests. Business logic comprises of rules, functions, parameters and the dependency relationships and is extracted for any particular request. When similar requests are encountered, their reference points are identified and the corresponding business logic is extracted.

On account of a new change request, a fresh plan is created. The Business Logic pattern (BL pattern) helps to extract the required business logic from BL Source using pre-processed knowledge about logic extraction. Hence it saves enormous amount of time since an existing pattern can be referred immediately if available instead of generating fresh ones every time. The Context Analysis is with respect to time and environment. The change is raised at which context and whether the change is to be made at rule level or function level is identified by the context analyser. The domain on which the change request raised and its sub domain along with business process name, business process id and its owner details are identified by domain analyzer. For all change requests, domain and context analysis are made irrespective of their nature.
The BL source, i.e. the Business Logic source is a repository of the Business Logic which consists of Rule Set, Function Set, Parameter Set, Dependency Set and Policy Set, etc. The Business Logic Extraction involves extracting the appropriate logic from the BL source with corresponding source code access permission. All these together contribute towards the first phase of the experimentation framework.

4.2.2 Phase II of Business Logic Evaluation Model

The Phase II comprises of BL Analyzer and Schema Generator. The Business Logic extracted from the phase I is subjected to a Rule Bound Analysis. This analysis makes sure that the extracted logic is bound to the set of rules. It checks all segments of the extracted logic are bound with the rules available in Rule Set. After Rule Bound Analysis, the business logic set (BL Set) is composed in such a way that every set consists of rules, functions, parameters and dependency relations. During this BL Set composition, the business policies associated with the business rules are also be extracted and referred into the dependency set. All these steps are taken to facilitate the construction of the Finite State Machine (FSM) from the composed business logic.

The construction of FSM facilitates the entire business logic change evaluation at various stages since it acts as a conceptual model for describing the business logic set. All concepts of formal language theory is incorporated in business logic set in order to maintain the logical structure of BL Set. The classification of rules and their further deepening into functions reduces the time and space required by the FSM. So changes can be made at any required level, at a function, sub function or a primitive function etc.

The Interoperability verification checks the extent to which the interoperability can be achieved at the rule, function and primitive function level in order to generate Business Logic Schema with structural orientation. This is followed by generating the schema which is XML based and tags are used for describing every rule, function, relationships etc. The schema comprises of reference points which carry some amount of knowledge in terms of Meta data in the BL schema. These reference points refer to historical events which help to map the existing change incidents with respect to the current change event. On event of a change, the reference points are fine-tuned and therefore the number of reference points will be gradually reduced. Reference points focus on the required business logic entity and the knowledge gathered increases gradually. Since all changes are done at the schema level by the business
analyst, the evaluated change progress is notified as Meta data in the effect of any change event.

The Configure Schema evaluates all schemas which get stored in the BL source after property evaluation. It checks if all BL entities are available, if valid tags are available and if each entity has metadata associated with it.

4.2.3 Phase III of Business Logic Evaluation Model

The Phase III comprises of Property Evaluator and Change Analyzer. The Property evaluator includes evaluating the completeness, finiteness, accessibility and traceability, etc. Since there is no complete ownership, accessibility has to be evaluated as situations might arise where a prior permission from other analysts is required. It is considered that changes cannot be made to the content within the accessibility tag in the schema.

The Property Evaluator act as a pre request to incorporate changes over the logic and checks whether the logic is computable, traceable, accessible and configurable in prior to make the actual change. The key goal of this Property Evaluator is to provide better run time support during the course of change progress. The Security Assessment deals with the security issues checking if all business logic entities are safe with respect to schema level and source level, etc. It further involves verification of subject, resource and environment by scrutinizing the accessibility level required by each. Policy Verification involves verifying the different policies that the business logic entities follow and checking if all the entities abide by those policies.

Service Level Agreement Verification (SLA Verification) is very important in this context because there is no single ownership observed. This might cause any function in business logic to be dependent on a different function in the business logic of another owner. So an SLA Verification must be necessarily done. Apart from this, SLA verification also needs to be done between Service Provider and Service Consumer to make sure all service invocations are legal and as per the requirements.

Property Evaluator in fact investigates the defects in the logic and validates the quality of business logic by automating various functional and non-functional assessments to ensure that new changes have not impaired existing functionality. Functional assessments assess the business logic through properties such as computability, traceability and manageability and thereby ensure that efficiency and performance of the logic is not pulled
back. Here, Computability refers to the feasibility of the modification anticipated. It examines each and every rule, and checks if the functions and parameters in the logic are completely computable. Whenever the modification is done, it validates modified logic to be computable within the time limit.

Computability is the measure of finiteness or completeness. Traceability aids in tracing back to the point of problem and the previous state before it in case of errors if any. Accessibility deals with obtaining prior permission for accessing the content within the accessibility tag due to the absence of single ownership. In many cases, system logic is embedded in to the business logic which makes business analyst hard to focus on business logic. Configurability does not depend on the business logic but on the system logic. So this property helps to track and reconfigure the system logic without affecting the business logic. It provides the functional capability to verify that the resources required to handle the initial and modified logics are available in the system. These are the properties to be evaluated.

The Change Analyzer is the next part of phase III which highlights the Interoperability Verification with respect to the evaluated properties. The interoperability of the extracted business logic will be set to true if all properties associated with the logic are successfully evaluated and all properties values are set to true. Change Analyzer allows an authorized person to do, monitor and manage meaningful changes. It maintains a change log which records the change progress with completeness. The actual execution of the change leads to the creation and alteration of code. When this change is propagated, it might probably cause other code fragments to change as well.

Change Analyzer scrutinizes the validity of modified rules, functions and parameters. The results of the Property Evaluation are fetched after the change is made and then the Interoperability is verified. Interoperability is verified in order to make sure that the changes made do not bring in any degradation in the service quality and that there are no conflicting changes made. This is followed by calibrating the Cellular Automaton. From this, manageability is evaluated.

When every required run time support is available for performing a change, then the change is said to be manageable. Determining whether the changes are manageable or unmanageable beforehand is encouraged because it allows the analyst to infer if the change would be meaningful or not.
4.2.4 Phase IV of Business Logic Evaluation Model

The Phase IV involves Change Evaluator and Integration Adapter. Change evaluation makes use of a manageability algorithm which helps in evaluating whether a change will be meaningful or not. This is very advantageous since whether a change is manageable or unmanageable is said well before. When all the run time support is available, a change is said to be manageable.

Once the manageability is found to be true, the edit mode is enabled and all the change factors (dynamic in nature) are evaluated followed by determining the desirability (The Change Factor varies for each rule). Change Evaluation provides efficient behaviour analysis and provides the guidance verifying that the right path is followed. With respect to Impact Analysis, the history of incidence matters. This is because there is no use of analyzing the impact when the request is fresh, occurring for the first time.

This is followed by the Dependency Analyser where the dependence of the business logic entities with the entities in the business logic of other owners is analyzed. In the dependency analyzer, rule similarity and rule dependency are found. In general, parameters are associated within functions and functions within rules. Same parameters may be involved between two functions and functions between two rules. These similarities and dependencies are analyzed in this part. The entities may depend for the input (where the input for two entities are the same thereby making them dependent), output (where the output from two entities are the same thereby making them dependent), mapping (where two entities use the same mapping, data sets for instance, making them dependent), existence (where the presence of one entity is necessarily needed for the functioning of another entity) and call sequence (the order in which entities call each other like triggering).

These dependencies are analysed by the Dependency Analyser. Depending on these relationships, Union, Composition, Substitution or Reduction of the services is done appropriately. Then the Property Evaluator is called and the various properties like computability, traceability, accessibility and configurability are evaluated. The code is dealt at schema level and at source level. The order of execution is the sequence in which the business rules are executed. The Mapping function deals with mapping the appropriate entities. Policy enforcement involves verifying that the policies needed are properly enforced. Schema Validation is checking the schema to be of correct format and structure. It is checked that all the property information are available, the rule or logic is complete and the structural
orientation of the schema is well laid. The tags are verified and the presence of end tags for every tag is checked. At the source level, the code consistency, similarity measure and time and space complexity are taken care.

All these are checked and analysed in order to meet the goals which comprise of the incident matching, degree of automation, degree of authorization, success rate, rate of knowledge transfer and the amount of risk. Automation is said to be achieved because when a request arrives and a change is made, the properties are evaluated and the risk is estimated. The change is committed without any notification in case of the risk being nil. Even on cases where the risk is moderate, i.e. when the risk is below the risk threshold, changes are committed without any notification. Only when the risk is above the threshold, the analyst is notified. Therefore there is automation seen in making this decision. Degree of authorization refers to the degree to which a person is authorized to handle the change request. Actions can be taken in accordance with the authorization capability only. Incident matching relates to comparing a change request with the history of the events and the changes i.e. a match of incidence of change events is searched for.

4.2.5 Phase V of Business Logic Evaluation Model

The Phase V involves the Growth Rate Prediction Engine. The various mentioned goals are assessed. These are the goals of the framework and the analyst. When a part of the logic is extracted from an entire source, it tends to have issues related to computability, accessibility, configurability and traceability. So property evaluation is done and the correct logic enters the change evaluation part.

Here, the change factors like order of execution, mapping function, policy enforcement, schema validation, code consistency, similarity measure, and time and space complexity are evaluated and fetched. The Impact factor is fetched based on the incidence and then the Impact analysis is performed and the cellular automata patterns are generated. The Change Authority Board handles the code review consistently. However, Cellular Automata which is considered as static here can be made to recognize cell position changes through which dynamism can be brought into the picture and many patterns can be considered in future. Here, Cellular automata are considered to be static and every cell is considered to be in the stable position.
For deciding if a cell is stable or not, the change factor values are fetched and compared with the threshold value. The properties are evaluated after change again. The variation in the value before and after change is the Impact Value. This impact value is compared with the impact threshold and if found lesser than the threshold, the risk is assumed to be nil. CA calibration involves checking if the values are within the threshold. Decidability deals with the making decisions based on the analysis and deciding if the change can be committed or discarded.

4.2.6 Phase VI of Business Logic Evaluation Model

The Phase VI involves the Run Time System. This phase performs the BL Schema Verification which involves verifying if the schema is in the proper structural orientation and if all the property information is available. The WS Source Translation is needed because after a change is committed, the schema is altered and so is the source. Therefore recompilation and generation of class files must be done and redeployment is needed. The WS source Translation takes care of automatic source compilation and deployment. The Run Time Manager takes care of all these and provides run time support.

4.3 WORKING PRINCIPLE OF BLEM

The Change request is originated. The Change request consists of the command to be executed i.e. the DML operations to be performed which might be an update or a substitution etc., the resource to which the change is to be done which is expressed in terms of the process name and the name of the rule or function if needed and the condition to be followed i.e. performing the mentioned steps on the satisfaction of certain condition mentioned. Once the request for change is received, the source code to which the change is to be done is extracted with the help of the resource mentioned in the change request. The source code can be in any language.

After the source code is extracted, the corresponding Finite State Machine is generated. The FSM provides better understandability and is language neutral. It is a conceptual model and is machine processable. FSM provides enhanced focus on the part of concern. The FSM reflects the schema. Any change made to the FSM is reflected in the schema. While FSM consists of nodes, the schema consists of Meta data which are very useful when fetched during backtracking. In order to fetch the Meta data, the information about the node is needed which is provided by the FSM.
From the overall business logics, the logic in which the change has to be made is extracted as a Business Logic set (BL set) containing business rules, business functions, business parameters and dependency relations (i.e., \( L \rightarrow (R, F, P, D) \)). Then extracted logics are forwarded to the property analyzer. Business logic is a group of the logic entities which includes rules, functions, parameters and dependency. Every operation in a web service is rule based. After the FSM is constructed, the rules, functions, parameters and dependency are identified and are placed in the business logic set. Every rule is converted into schema and this reduces the time taken in the generation of schema at run time. Rules, their syntax, their description and the schema are identified. Functions are expressed in terms of primitive computable functions. Patterns are generated during impact analysis.

This is followed by the dependency analysis. The entities are checked for any input dependency where two or more entities depend on the same output, output dependency where entities are expected to produce the same output, call sequence dependency, policy dependency where the entities are expected to abide by the same policy, mapping dependency where entities depend on the same data set for instance and existence dependency where the existence of an entity is meaningless in the absence of another particular entity. This analysis is made always.

After the dependency analysis, the business logic schema before change is identified. Property Evaluation is done all the time during the schema extraction, during the change process and even after the change is committed or discarded. The property evaluator checks for the four key run time compatibilities by constructing STT (State Transition Table) and FA (Finite Automaton) for the target business logic. BL Source has State Transition Table (STT) of the FSM for the logic associated with the change request and Change Measure Table (CMT). The output of property evaluator is tracked by the 4 tuple 4T\textsubscript{OP} tracker and given as input to the cellular automaton for manageability calibration.

The manageability of the business logic is determined by the property evaluator which eventually generates property pattern and acts as prerequisite for subsequent change evaluation phase and impact analysis phases. The properties like computability, accessibility, traceability and configurability are evaluated and the property evaluation tag is updated in the business logic schema. After the change is made, the changed business logic schema is extracted and the change evaluation is done.
Figure 4.2 Workflow Diagram of BLEM
While the analyst is performing the changes on the service logic at schema level, the changes made are transformed into the source level and the necessary code and resource segments are updated after the change request is committed. Excessive care is required in perpetrating the changes made, especially in case of importunate change requests. For this, Change factors like order of execution, time complexity, similarity measure and the schema validation are evaluated. Not only the functional factors but also the non-functional factors are evaluated of which the response time, service interruption, service availability time and reliability are of the main concern here. These factors are evaluated before and after change. The change evaluation values are updated as reference points in the business logic schema before the change is committed. Based on these values and the impact factors, impact analysis is made.

The Impact analysis graph is the diagram is plotted with the change factors against the percentage of their magnitude. Cellular rule patterns are generated in order to perform the impact analysis. Based on all these, decidability i.e. deciding whether the change has to be committed or discarded is done.

**4.4 SUMMARY**

This chapter therefore focuses on the Business Logic Evaluation Model. This model is the core of the proposed Change Management Framework and supports Run Time Management, Change Factor Management and Proactive Management. Whilst the Business Process Model is efficient but applies for the complex, concurrent processes alone and the Business Rule Model is applicable for simple processes but handles the modifications at the rule level alone, there was a burning necessity for filling the gap between these two models which is satisfied by the proposed Business Logic Evaluation Model. This chapter has focussed on the workflow of the proposed model and the depiction of its working. The proposed model involves various phases including the Request Handler, Source Analyzer, BL Analyzer, Schema Generator, Property Evaluator, Change Analyzer, Change Evaluator, Integration Adapter, Growth Rate Prediction engine and the Run Time System. The extraction of the Business logic based upon the Change request and the further processing to discover the manageability, decidability and the behaviour analysis are described.