TOWARDS AN EVENT DRIVEN COMPUTING MODEL

Joby Joseph[1],C. Jothi Venkateswaran[2]

[1]Research Scholar, Bharath University, Chennai, (India)
[2]Head, Dept. of Computer Science, Presidency College, Chennai, (India)

ABSTRACT

There are various models of computing. Each mode of computing comes with its style and efficiency. Events are basic in any system of processing. The occurrence of an event is the basic in an event driven model. The components starting from the occurrence of an event to its capture, analysis to the level of its processing are seen in the paper. Every event has its logical end with a chain of events that can lead up to a scalable processing model. And it is this spanning out feature that allows the Event Driven Computing a promise for high level of process.

Keywords: Events, Event Driven Architecture, Event Processing, Event Computing

I INTRODUCTION

Events are in every aspect of life. Everything that happens around is an event that has its cause and effect on subsequent sections. One of the emerging modes of computing is the Event Driven Architecture, where computation takes place as response to events that occur in the system [1]. This paper is a detailed study on this model of computing. The paper begins with a detailed study of the fundamentals in the field of event processing, their characteristics, prerequisites and go on to analyse in detail an architectural layout that is useful in and Event Driven model.

II BASICS OF EVENT DRIVEN COMPUTING

An Event Driven model of computing presents a modern, dynamic and reconfigurable processing solution to the computing world. An Event Driven computing is one that can detect the various *events* that occur in the system and respond intelligently to it. In order to understand the proper intricacies of an Event Driven system, the term *event* needs to be defined. An event can be defined as a notable action that happens inside or outside the system. An event in the computing scenario can signify an occurrence or non-occurrence of an action [2]. Any event has three aspects associated with it that needs to be clarified in the realm of the computing model. The first part is that –an event has occurs as notable action in the system and the second aspect is that there is the definition for an event that classifies the action as an event and finally there is the details of the specific event. Hence it is important to understand these aspects of an *event:* viz., Event Notification, Event Definition and Event Specification while considering an EDA model [3]. These are seen in detail in the coming section of the chapter.

II CHARACTERISTICS OF EVENT DRIVEN COMPUTING

Any event driven model of processing exhibits the following computing properties:

- **Observation of Events:** This feature implies that the EDA model is actively monitoring the system and any occurrence of an event initiates the necessary alerts to the system [1, 3].

- **Information Dissemination:** This aspect of the EDA model is to deliver the right information to the right component at the right intensity to the proper respondents [1].
- **Dynamic Operational Behaviour:** Every event driven computing system needs dynamic and quick interaction to respond to the report of an event [1, 4].
- **Active Diagnostics:** Every event driven system should also be able to diagnose a problem if it occurs based on the events observed [1, 5].
- **Predictive Processing:** In an efficient EDA model, the style is not merely *report-respond* model of processing, there should also be methods to prevent erroneous events that can cause fault in the process [1, 5].

The conceptual diagram incorporating all these features is in figure 1.1.

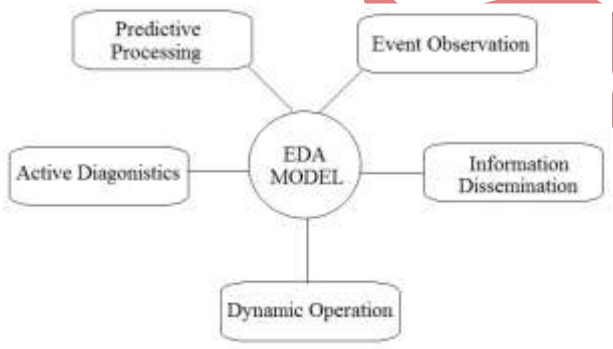

Figure 1.1: Event Driven Architecture Characteristics

III EVENT PROCESSING REQUIREMENTS

An Event Driven Architecture needs certain functional capabilities that allow the system to work with optimum throughput. Some of these requirements are:

- ***Event Input/Output***: Most event driven applications need some way to consume the events. These can come from other input devices or applications that are internal or external to the system. Similarly, applications also need a way to output an event as a response [1,6]
- ***Data Reduction:*** This involves the process of extracting information from the ingested input events. These include the steps of filtering, projection and aggregation [1, 6]
- ***Reasoning:*** This is the process of extracting the desired event from the set of raw events through the stages of transformation, aggregation and pattern detection [1, 6]
- ***Analysis***: The process of analysis involves the stages of detailed observation of extracted event in comparison to existing event details that are kept in logs [6].
- ***Prediction***: Based on the analysis of the event and the log data, predictive actions can be made regarding the system and possible events in the future can be anticipated [4, 6].

- ***Learning***: As events are observed, analysed and predictive plans are marked, an EDA system does lot of inherent learning and adaptation for the future [4, 6]
- ***Distribution***: The last task of the EDA system has the function of passing the events received to the appropriate module so that proper event processing can take place [6]

IV EVENT PROCESSING

Most event processing models have a computing process that responds to event objects. The event processing mostly includes creating, reading, deleting, transforming and responding to such objects. Such systems will mostly comprise of two major components: First, a Sensor or a source that senses any event and emits event objects. Secondly, a Responder, that receives and responds to such event objects. Both these actions occur at two layers of the model [7]. Any EDA model will consist of multiple levels of sensors and responders. Each acts in response to a request from the other. The mode of processing that occurs in most EDA is explained in the sections that follow [8]:

4.1 Event Processing Style

In most EDA, there are three general styles of event processing. They are Simple, Stream and Complex.

4.1.1 Simple Event Processing

In such a model an event happens and it initiates many downstream actions. In most real time flow of business systems, such models are used. Simple event processing is the base of an EDA system and it makes up the base component that initiates the event processing style in the computing realm [9], [1].

4.1.2 Stream Event Processing

In such a model, there are both ordinary and notable events happening. This model is commonly used to drive the real time flow of information in and around an enterprise, or a business or computing models, making decision making in time [7]. In the stream event mode of processing there is the combination ordinary events that make up a chain to have them in the stream format. Such events can be as the cause or effect of other preceding events that happen elsewhere in the system [9].

4.1.3 Complex Event Processing

Complex Event Processing deals with evaluating a combination of events and then taking the action. There is a high degree of event-correlation possible here. This model will need high degree of matching and processing techniques. This is commonly used in models that need high degree of adaptability, responses and inter-dependencies [6, 9].

In all these levels of interaction and information flow, there is a need of high degree of data interchange and interconnect. The architectural design of the models will have to be one that supports this at the processing and design levels of the model. The architecture needs systems that detect, transfer and process the events appropriately while maintaining efficiency and agility at each stage. These are performed at the various layers that occur in an EDA model [9]. These are explained in the sections that follow

4.2 Event Flow Layers

Every event in the model starts with an event being generated at some level and proceeds to any down-stream layers to meet its activity. The stages from the occurrence to the action done in response to the event can be classified in various layers. The layering of an event makes it easy to understand and to troubleshoot it at the within the right time and place. Figure 1.2 is a structural diagram for the layers. This involves the following logical layers [1, 7, 8].

4.2.1 Event Generator

This is the source that generates an event. The source may be an application, service or a request of any type. In most cases, not all events are generated in the required format for processing. In such cases, they have to be transformed to a format prior to depositing the event into the next stage. The generated event is as raw data, it is to be pruned, processed prior to being processed. An EDA model consists of such applications that perform these pre-processing tasks [7 , 8].

4.2.2 Event Channel

The event channel typically is the messaging backbone, which transports standard formatted events between the event generators, the processing systems and the downstream layers. An event that happens at one level of the system needs to be transferred to the proper level(s) for its subsequent processing or for initiating other successive event generation. The channel takes care of the flow of an event to such applications in the EDA model [1, 7].

4.2.3 Event Processing Layer

At this layer, once the events are received, they are evaluated against the set of rules for processing and appropriate span out actions are initiated. The actions may include invoking a service, initiating a process, publishing the event to another system, generating another new event or capturing the event for a later process. A lay out diagram of the three stages of the layers is shown below in the diagram [1, 7 , 8].

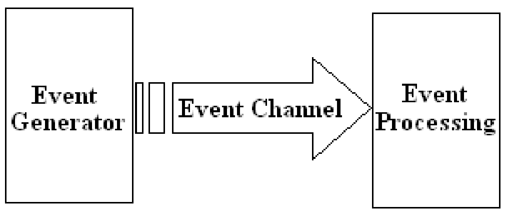

Figure 1.2: Event Flow Layers

V EVENT DRIVEN ARCHITECTURE

A layout diagram for the EDA model in the context of a Cloud paradigm is shown in Fig. 1.3. The basic nature of such a model is that the system is triggered by the occurrence or nonoccurrence of an *event* [1, 10].

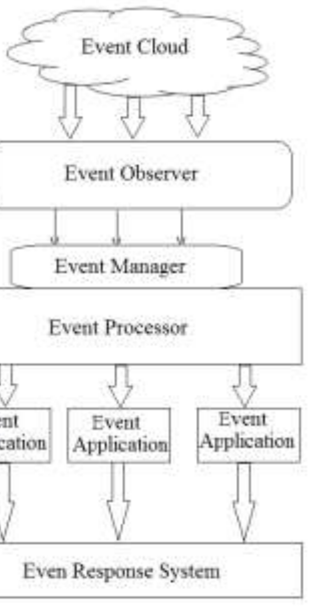

Figure 1.3: Proposed Event Driven Architecture for the Cloud Model

The system is as a single unit inherent in the Cloud set up and the whole set up is to be seen as a constituent part of the Cloud. The events that occur in the cloud are sensed by the Event Observer and it passes the events thus observed to the Event Manager. This is mostly a middleware that acts as a control and coordination units prior to passing the events to the next unit of Event Processor.

In the Event Processor, the input events are filter, correlated and all pruning is done to isolate and collect meaningful data from the raw input of events. The output of this unit is the notification to the proper Event Application within the EDA model. This initiates the proper *response* from the Event Application to the Event Response System. The Event Response system initiates proper actions and to the required unit within the cloud set up.

5.1 Uniqueness of Event Driven Architecture

Event Driven mode of computing is one that is gaining advance in the computing models where large scale distributed computing or service oriented computing takes place. In the sections that follow, an analysis is done to highlight the features of the EDA model that single it out from other emerging models of computing [8, 9, 11, 12]:

- ***Broadcasting Events:*** The occurrence of an event is broadcasted to all the units in the system that are actively participating in the architecture and any one system or multiple systems can listen to an event and take up the process so initiated. This allows a degree of parallelism as well as dynamism in response
- ***Asynchronomy:*** The constituent systems in an EDA publish events as they occur instead of storing them and delaying for a process cycle. This incorporates a lot of asynchronous process style in responding and attending to input events. This style of process can make the EDA mode of information process with optimum use of the resources for performance as they are done on the fly
- ***Ontology of Events:*** An EDA can have built in rules to classify and event or to group a collection of events into an ordered hierarchy. Hence, once an event is published, the receiving system can respond to

the event based on its hierarchy or handle a category of them in a like manner. This brings in a priority mode of processing into the EDA model.

- ***Complex Events:*** Many events that happen within the system have inter-relation to higher level of events and can cause a pattern of lower level events. Hence, in an EDA model, there can be an advantage of event aggregation or event causality. Both these features together allow requests and responses in the system more scalable and parallel.
- ***Event Replay:*** The EDA has another important and unique characteristic. If all the interaction within the system takes place as events, then the same can be recreated by replaying the events all over again. The advantage is that as events are recreated, some of these can be replayed with modifications. The replay feature is good to analyse and make future correction for the EDA model.

VI RESEARCH TRENDS IN EVENT PROCESSING

This section makes listing of the emerging research trends in the field of Event Driven processing. The field of Event processing itself is new and there are many areas of application and related research that can be anticipated. Some of the outstanding ones are listed below:

- ***Towards Decentralization:*** The Event Driven computing has found wide area of coverage. It is incorporated into many fields of application as Event Processing Technical Society (EPTS) has noted. This has brought about new programming and architectural challenges for research. The event driven model brings a shift from the monolithic architectures that were so commonly used to architectures that are diversified. The shift is from a centralized architecture to a distributed model keeping in line with the current mode of parallel computing. This diversification comes with variety of functions, quality of services and a variety of platforms over which such models can be tested and tried [1].
- ***Greater Standardization:*** As Event Processing becomes more popular and is implemented into diverse applications, there is the new research trend of incorporating a standardized mode of conceptualization and implementation. The various fields of standardization that the research focus is on are: Event structure and related metadata representation, event distribution standards, event processing language, event assembling model and event rules that can guide the system. A common norm and rule can bring about a greater level of interoperability in the processing arena [1, 6].
- ***Business Model:*** Current area of focus is much on business process management using the event model of data management and process. The EDA model is heavily used in business workflow systems [2].
- ***Embedded Models:*** Lots of current research is active into the area of embedded event models. This involves event functions being embedded into the packaged applications or into other middleware applications. In all these the focus of the current trend is to switch gear from having a reactive model of computing to a proactive model of processing [1, 6].
- ***Other Areas:*** There is lot of research on to realize EDA models of virtual platforms. These include hardware application platforms, embedded platforms etc. Hardware application platforms with multicore processors, embedded platforms like robotics or other gateways where EDA models are areas of related research [1, 4].

VII CONCLUSION

The uniqueness of the Event Driven architecture is a promise to many emerging models of computing. The EDA seems to promise lot of processing style in Service Oriented Architectures. The Cloud model of computing is one where there are applications that are service driven. In such model, the computing power in the model can be better utilized and thus served if the EDA model is incorporated into the Cloud. This opens a span of area for incorporating such a model of computing into the cloud model.

REFERENCES

[1]. Hugh Taylor, Angela Yochem, Les Phillips, Frank Martinez, “Event-Driven Architecture: How SOA Enables The Real-Time Enterprise”, Pearson, NewDelhi, 2011.

[2]. Peter Niblett, David Luckham, Opher Etzion, “Event Processing In Action”, Manning Publications, New Delhi, 2010.

[3]. Badri Sriramn, “Event Driven Architecture Augmenting Service Oriented Architectures”, Sun Mircrosystems, January, 2005.

[4]. Bartosz Kowalewski, Marian Bubak and Bartosz Balis, “ An Event-Based Approach to Reducing Coupling in Large-Scale Applications”, Krakow, Poland, 2009.

[5]. Jiahai Yang, Jianping Wu and Yue You, “A Web- based, Event Driven Management Architecture”, 2008.

[6]. Chung-Sheng Li, “Real-Time Event Driven Architecture for Activity Monitoring and Early Warning”, in Proc. Of Emerging Information Technology Conference, 2005.

[7]. Gregor Hohpe, "Programming Without a Call Stack- Event-driven Architectures", http://www.eaipatterns.com, 2006.

[8]. Brenda M. Michelson, "Event-Driven Architecture Overview", Patricia Seybold Group and Elemental Links ,Inc., 2006.

[9]. http://www.dagstuhl.de/10201, "The Event Processing Manifesto", in the proc. of 2010 Dagstuhl Seminar on Event Processing, Dagstuhl, 2010.

[10]. K. Mani Chandy, "Event-Driven Applications: Costs, Benefits and Design Approaches", in the Proc. of Gartner Application Integration and Web Services Summit 2006, Sao Paulo, Brazil, 2006.

[11]. Olga Levina and Vladimir Stantchev, "Realizing Event-Driven SOA", in the Proc. of 2009 Fourth International Conference on Internet and Web Applications and Services (ICIW '09), Los Alamitos,CA, USA, 2009.

[12]. Greg Eisenhauer, Hasan Abbasi, Matthew Wolf and Karsten Schwan, "Event-based Systems: Oppurtunities and Challenges at Exascale", in the Proc. of DEBS'09, Nashville, TN, 2009.