Design Of Context-Aware Interactive Voice Response System

Dr. Thirumaran M., Subham Soni, Gayathry @ Brendha G.

Abstract
Interactive Voice Response (IVR) is an automated telephony system that answers most of the customers’ inquiries. An IVR system can handle 20% to 60% of the calls regarding common queries. Most of the IVR systems used today are content-based IVR systems. Hence they do not provide quality service to the customers because the information is directly fetched from the database and presented to the users and also the IVR systems use APIs that are in-built in the system and hence dependent on the system. In this paper we are proposing a Context-Aware Interactive Voice Response System that generates a dialogue based on the context, meaning based on the user’s requirements the IVR menu is dynamically generated. Unlike the traditional IVR systems, it is a service-based IVR, meaning all the operations in the IVR system are implemented as web services. Also, the IVR system is implemented as a Finite State Machine (FSM). The request of a user by the IVR system is processed in stages (the number of stages depend on the number of sub-menus generated) and during each stage the FSM saves the state information of the user with respect to the query, that is to what extent the query is answered to the user and the state transitions that take place during the call flow. Because of the use of Finite State Machine, the IVR system is more reliable and robust.

Introduction to Interactive Voice Response System:
Interactive Voice Response System or IVRS is a technology that allows interaction between a computer system and the user through the use of voice (or) Dual Tone Multi Frequency (DTMF) keypad tones, to provide services for common queries to the customers. The ultimate aim of IVRS is to reduce the customer service costs by providing self service to customers and guiding them to the department, person or information they need.

Service-based Interactive Voice Response (IVR) System:
The traditional IVR systems are content-based IVR systems. In these systems, the information is fetched directly from the database using the database queries and stored procedures. The major disadvantage of these systems is that these systems are static and hence it is tedious to build a module dynamically for a request.

Service-based Interactive Voice Response System uses web services to fetch information from the database. According to W3C, “Web Services are a software system designed to support interoperable machine-to-machine interaction over a network” and hence they don’t depend on the system level APIs. When the system receives an IVR request, it determines the I/O parameters and composes a web service and invokes it. The web service in turn fetches the information from the database and gives the results to the IVR system. Unlike the content-based IVR systems which involve only extraction of information from the database, service-based IVR using web services which involve both computation and extraction; for a particular request the IVR can extract information from the database and perform other computations (e.g. conversions) with ease for that request.

Introduction to Finite State Machine (FSM):
A Finite-State Machine (FSM) is a mathematical model of computation used to design both computer programs and sequential logic circuits. It is formulated as an abstract machine that can be in one of a finite
number of states. The machine can be only in one state at a particular instant of time. This state which the FSM is in at a given instant of time is called the current state. When an event is triggered, the FSM can change from one state to another and this process is called as transition. A particular FSM is defined by the list of possible states it can be in and the triggering event/condition for each transition.

**Related Works:**

The process of creating a new web service from web services which are available in the service repository is termed as web service composition. Many times there are redundant web services available. Meaning, for the same operation, multiple services might be available. Safi, A.; Jawawi, D.N.A.; Wakil, K describe a new method for redundant-free web service composition. To select the ideal services, the method uses a linked list data structure called Composition List (CL). Also to find web services which are redundant and reject them from the process of web service composition a hash table called Available Output Parameters (AOP) is used (Safi, A., et al., 2013).

A large number of web services exist without semantic descriptions. Henceforth, a lot of web services that might be relevant or useful to the user’s request might not be taken into consideration during service discovery. Aabhas V. Paliwal, Basit Shafiq, Jaideep Vaidya, Hui Xiong and Nabil Adam approach to semantic based web service discovery involving semantic-based service categorization and semantic enhancement of the service request. Based on an ontology framework a solution is proposed for achieving functional level service categorization. Also, to classify the web services accurately according based on their functionality clustering is used for accurately classifying the web services based on service functionality. With relevant services the semantic enhancement of the service request achieves a better matching. An efficient matching of the enhanced service request with the retrieved service descriptions is achieved utilizing Latent Semantic Indexing (LSI) (Paliwal, A.V., et al., 2012).

In Service Oriented Architecture (SOA), Automatic Web service Composition (AWSC) (Kuehne, B.T., et al., 2013) is a major challenge because of the lack of availability of a platform to implement and test algorithms for AWSC. Tardiole Kuehne, B.; Carlucci Santana, R.H.; Limnemann, V.; Santana, M.J. present the Automatic Web service composition Architecture (AWSCA), an architecture where it is possible to implement and execute the AWSC algorithms for a fair comparison.

In software development applications can be developed by assembling reusable blocks called web services. To create new business applications industries have adopted web service composition. During a web service composition, many unexpected errors might occur and sometimes it is difficult to detect the source of the error as the errors might spread (or) propagate (or) accumulate. To enhance testing of interactions among the web services, Ching-Seh Wu; Chi-Hsin Huang propose a technique of Model-Based Testing (MBT). This technique is a combination of Extended Finite State Machine (EFSM) and UML sequence diagram and generates a test model called EFSM-SeTM (Wu, C.-S., C.-H. Huang, 2013).

Dynamism, Traffic load, Scalability and Reliability are the important parameters using which the performance of an IVR system is evaluated. Chang-Xing Qi, Jie Liu, and Hang Li propose a Statistical Regression Algorithm to generate a dynamic menu for the IVR system. The Statistical Regression algorithm is implemented using IVR application gateway. The IVR server, TTS (Text To Speech) component and the application gateway together build the voice files for the data requested and played on response. The advantages of this algorithm are the performance of the system is improved, the system load is reduced and the requirements of the customers are met (Qi, C.X., et al., 2010).

Providing cost effective services to the customers is one of the major challenges in contact centers. Personalization is one of the ways to achieve it. Personalization, here refers to user-specific menu. Mudili Soujanya and Sarun Kumar propose a Personalized IVR system in Contact Center. Here each customer have their own privileges to customize the menu according to their requirements. In this way the customers have finer control over the IVR system and henceforth it results in a more satisfied self-service (Soujanya, M., S. Kumar., 2010).

Present IVR systems do not provide enough customer satisfaction because they lack dynamism to provide the required response to the customers, who wish to perform business critical emergency changes reliably and sophistically by themselves without depending on the agents. Thirumaran,M, Dh vachelvan.P and Aishwarya.D by using Dynamic IVR systems propose an approach to discover a set of composed services. With respect to user targeted QoS and the set of business policies of the organization, this approach makes the service discovery process more efficient, accurate and reliable. For analyzing the dependency which exists between the composed set of services, the approach uses Fuzzy Analytical Hierarchical Process (FAHP). FAHP (Thirumaran, M., et al., 2012) is also used for optimizing the request-reply dialog process between the service provider and consumer of the IVR system.

Most of the companies use Telephony Call Center (CC) to deliver services through telephone lines to the customers and thus it has been the traditional way to main customer relationship. Call centers have developed Interactive Voice Response (IVR) systems for this purpose. But the major disadvantage of IVR is that it is not
dynamic in nature and the menus do not change with the user’s need and context. Ramazan Karademir and Emre Heves have introduced a Dynamic Interactive Voice Response Platform. The major advantage of this system is that it enables all kinds of IVR scenarios to be designed, changed, reported, inspected and managed by nonprogrammers. With this dynamic nature, the scenarios in the IVR system can be implemented easily, without modifying the underlying computer programming (Karademir, R., E. Heves., 2013).

A significant part of CTI center is the IVR. But the current IVR systems are designed for specific requirements with low extensibility and are not reusable. A new IVR system must be designed for another purpose. Shanchuan Xu, Wanlin Gao, Zhen Li, Shuliang Zhang, Jianing Zhao propose a hierarchical system of IVR which is reusable. The system reduces the dependency between the voice cards and it uses an XML file as configuration file to add maintainability to the system (Xu, S., et al., 2010).

Dynamic web service selection is the process of determining a subset of services to be invoked at runtime to compose web services. In a failure-prone environment, the composite and constituent web services are constrained by the sequence of invocation of their operations. San-Yih Hwang, Ee-Peng Lim, Chien-Hsiang Lee, and Cheng Hung Chen propose a model to permit invocation sequences of web services using a finite state machine. Each state of execution is assigned an aggregated reliability to measure the probability that the given state will lead to successful execution in the context where each Web service may fail with some probability. The computation of aggregated reliabilities is equivalent to eigenvector computation and the power method is adopted to derive the aggregated reliabilities efficiently (Hwang, S.-Y., et al., 2008).

Proposed Method:
The proposed Context-Aware Interactive Voice Response System is a service-based IVR system. Also, the system uses domain-specific knowledge to provide response to the users. The domain-specific knowledge is designed with the help of ontology. Ontology is a knowledge representation framework. The requests in an IVR system might be deterministic or non-deterministic.

2.1 Deterministic Requests:
Deterministic requests are those for which the web service exists already (or) the call-flow is direct (involving sub-menus and finally a service call).

2.2 Non-deterministic Requests:
Non-deterministic requests are those for which the web services are not defined (or) there is a lack of knowledge about the request and the system does not know how to proceed. In such cases, the system refers the domain specific knowledge and continues the dialogue conservation.

The architecture of the proposed IVR system is shown in figure 1. Every request will have unique id known as event-id. The event-id consists of <state, message-id>. The state in the event-id is the state the request is in the FSM during the execution. The message-id contains the address of the service to be executed from the service repository.

2.3 IVR subsystem and Request Handler:
When a request arrives at the IVR system, the IVR sub-system takes the request and passes it to the request handler. The system tries to determine the context of the user based on the user’s earlier interactions with the IVR system. If the request matches the pre-determined context which was achieved by referring the previous sessions of the user, then the menu for the request is generated by the IVR menu generator and it is given as a response to the user.

2.4 Semantic Analyzer:
If the pre-determined context does not match with the request, the request is handed over to the semantic analyzer. The semantic analyzer with the help of context, the event details and the semantic information generates a dynamic menu for the user. The semantic analyzer with the help of the semantic information, determines the deterministic services and uses the Flow Decider and Event Delegator to create the call flow.

2.5 Reasoner:
A Semantic Reasoner is a software that is used to infer logical consequences from a set of asserted facts (or) axioms. The Reasoner is one of the major components that help in the generation of the dynamic menu and also it is one of the integral component of the Semantic Analyzer. The Reasoner takes the results from Flow Decider and Event Delegator to generate the menu and passes it to the IVR Menu Generator. If any exception occurred, then the Reasoner will call the Exception Handler to handle the exception.
2.6 Service Detector:

The service detector detects the deterministic services from the service repository. It does this, with the help of the context and event details. The service repository is the storage space where all the services are stored. Based on the services stored in the service repository, the IVR Ontology Set is built. Also, as the service repository is updated, correspondingly the IVR Ontology Set is updated.

2.7 Flow Decider:

Since the IVR system is modelled as a Finite State Machine, the Flow Decider is used to keep track of the state of execution in the IVR system. Whenever a part of the request is processed successfully, the state is updated in the IVR system, with the help of the flow decider. In case of any failure, the system will continue the execution from the last saved state. Because of the introduction of Flow Decider, the system is more fault-tolerant.

2.8 Event Delegator:

The Event Delegator is responsible for the execution of the services based on the request, which were detected by the service detector. The Event Delegator contains the address of the services of those options in the menu that were dynamically generated by the semantic analyzer. Because of the use of event delegator, the system becomes customizable and easily maintainable. Any major change in the service code, will not have serious affects to the execution flow of the IVR.

2.9 IVR Ontology Set:

The IVR Ontology Set contains an ontology of the web services that are available in the service repository. The services are classified based on their usage. Example: In the banking domain, the services can be classified as loan services, transaction services, information services, etc. The IVR ontology set plays an important role in the generation of the dynamic menu. When a non-deterministic request is given, the IVR ontology set is used to determine what are all the deterministic services available based on the request and then the menu is generated.

2.10 Exception Handler:

Some times in the generation of the dynamic menu for a non-deterministic request, the services in the service repository might not be available for the request. Also, in some cases, the IVR system might hang (or) malfunction. All these are exception cases. The exception handler is used to handle such exceptions and provide appropriate response to the user.

Experiment Methodology:

The IVR system is modelled as a finite state machine. Whenever a request arrives at the IVR system, it is processed in stages and the corresponding states are updated accordingly. At any time, if a failure occurs in the system, the request is processed from the state where it was saved by the IVR system. The FSM implementation of the IVR system is shown in figure 2.
Fig. 2: IVR system modelled as a Finite State Machine

Here,
1. dr = Deterministic Request
2. ndr = Non-Deterministic Request
3. f = Direct Service Execution
4. q0 = Initial menu with options
5. q1 = User selecting options from the available menu
6. q2 = Required option not available in the menu
7. q3 = Service Call
8. q4 = Semantic Reference
9. q5 = Dynamic generation of menu
10. q6 = Service Call

The transition table for the FSM in figure 2 is shown in table 1.

<table>
<thead>
<tr>
<th>Next State</th>
<th>q0</th>
<th>q1</th>
<th>q2</th>
<th>q3</th>
<th>q4</th>
<th>q5</th>
<th>q6</th>
</tr>
</thead>
<tbody>
<tr>
<td>q0</td>
<td></td>
<td>dr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>q1</td>
<td></td>
<td>dr</td>
<td></td>
<td>f</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>q2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ε</td>
<td></td>
<td></td>
</tr>
<tr>
<td>q3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>q4</td>
<td></td>
<td></td>
<td></td>
<td>ε</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>q5</td>
<td></td>
<td>dr</td>
<td></td>
<td></td>
<td></td>
<td>f</td>
<td></td>
</tr>
<tr>
<td>q6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Initially when the user interacts with the IVR system, a default menu (or) menu with pre-defined context is displayed to the user and the FSM is in state q0. For the user inquiry, if the user selects an option from the menu itself, the FSM is in state q1. It is possible that further menus might be generated and hence the system stays in the state q1. If the option is a direct service call, then the FSM goes to state q3 and returns the reply to the user thus terminating the execution.

Sometimes it might also be possible that what the user requires is not available in the menu. In such cases, the system provides a facility in the menu itself to generate other possibilities. If the user selects such an option, the FSM goes to state q2. The system determines the possible services for that request by referring the ontology and hence is in state q4. From the services gathered, a dynamic menu is generated with the help of the Reasoner and Flow Decider and the FSM is in state q5. From q5, the user can select an option from the list of options provided and make the FSM go to state q1 (or) request for a new dynamic menu again and make the FSM go to state q2. If a direct service existed for the user in the options, then the service is executed and the result is given as a response to the user and state of the FSM is modified to q6.
3.1 Implementation Details:

As shown in figure 2, the proposed IVR system is modelled as a finite state machine. In specific it is an event-driven finite state machine where a transition from one state to another takes place based on an event. The FSM can be implemented using a switch case. As mentioned before, every request has a unique event-id. Based on the event-id, the transition takes place in the FSM. A sample switch case is shown in figure 3.

The switch case works based on the eventid. Based on the eventid, the service is executed and the menu is generated (or) the result is given to the user. The services to be executed are based on the service list extracted from the ontology. The service execution is based on the concept of function pointers. They contain the address of the service to be executed. Any change in the service implementation will not majorly affect the execution of the IVR system, because the service is not directly executed.

![Switch case code](image)

**Fig. 3:** Switch case for the Finite State Machine

**Metrics and Results:**

The evaluation metrics on which our proposed work is to be evaluated is shown in Table 2.

<table>
<thead>
<tr>
<th>METRIC</th>
<th>FORMULA</th>
<th>EXPLANATION</th>
<th>SYMBOLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level of semantic reference (Sf_l)</td>
<td>$S_{l} \times 10^L$</td>
<td>Level of ontology extracted to gain the required information</td>
<td>$L$ → Required level of ontology, $T_{s}$ → Total levels in ontology</td>
</tr>
<tr>
<td>Dynamic menu generation time (T_{dm})</td>
<td>$S_{l} 	imes \text{n}$</td>
<td>Time taken to create a dynamic menu for a non-deterministic request.</td>
<td>$\text{n}$ → Number of options in the menu, $S_{l}$ → Level of semantic reference</td>
</tr>
</tbody>
</table>

With the help of the metrics in the above mentioned table, the IVR is evaluated and its results are calibrated in Table 3.

<table>
<thead>
<tr>
<th>No</th>
<th>TYPE</th>
<th>SERVICE EXECUTION TIME</th>
<th>SERVICE RESPONSE TIME</th>
<th>LEVEL-SEMANTIC REFERENCE</th>
<th>OF</th>
<th>DYNAMIC GENERATION TIME</th>
<th>MENU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Direct</td>
<td>5 ms</td>
<td>2.5 ms</td>
<td>2.0 ms</td>
<td>3.0 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Indirect</td>
<td>7 ms</td>
<td>4.0 ms</td>
<td>3.0 ms</td>
<td>5.0 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Indirect</td>
<td>6 ms</td>
<td>3.0 ms</td>
<td>3.0 ms</td>
<td>4.5 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Direct</td>
<td>4 ms</td>
<td>2.5 ms</td>
<td>2.5 ms</td>
<td>3.2 ms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Indirect</td>
<td>6.5 ms</td>
<td>3.2 ms</td>
<td>2.5 ms</td>
<td>4.9 ms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In the above table the service execution and response time are evaluated with number of tools like Application manager, WAPT etc. From the results, we can see that this IVR system mostly provides solution to the users request by understanding his context with the help of the ontology whereas other IVR system provide error as their menu are static and their incapability to understand the user context.

Conclusion:
Determining the context of the user and then functioning accordingly is one of the major requirements of an IVR systems to enhance customer relationship in call centers and in organization. So in our work we analyzed the drawbacks of the existing IVR systems and introduced service-based IVR systems. We modelled the IVR system as a Finite State Machine that processes the request in stages and described the implementation details about the FSM. We see that by the use of domain specific knowledge, the lack of clarity and lack of knowledge is overcome. Also, by the use of FSM, it is easy to determine where the exception has occurred, meaning which state and the request can be processed accordingly. Hence we see that by the use of Finite State Machine in Interactive Voice Response Systems, they become more reliable and robust.

REFERENCES