Software Architecture and Quality
Course Code: PA1401
Assignment No: 1
Voice Driven Web System Architecture

Group member name and social id:

- Adrian Vladu (900521P736)
- Santosh Shah (870424P259)
- Zakaria Mahmud (89011011230)

Member contribution:

Name	Idea	Idea%	Documentation	Documentation%
Adrian Vladu	Introduction, Factors, Issues, Strategies, Conceptual View	37.5%	Introduction, Factors, Issues, Strategies, Conceptual View	37.5%
Santosh Shah	Factors, Issues, Strategies, Conceptual View	37.5%	Factors, Issues, Strategies, Conceptual View	37.5%
Zakaria Mahmud	Factors, Issues	25%	Factors, Issues	25%

It can also be found under https://github.com/santoshshah1/PA1410-assignment/blob/master/assignment-sources/12-13-2013/FinalAssignment1.pdf

1 Introduction

An interactive voice driven software system has a lot of potential these days because there are still many low-end phones in use. The phones might not be connected to Internet or could be used by people with little technological knowledge for various reasons, from the need of weather information to medical assistance.

The main purpose of the system is to deliver to the user a fast automatic voice response, which is dependent on the user voice or keypad press input. It is supposed to run without human intervention. The response has to be easy to understand, the voice interface should be easy to learn and, when needed, the system should provide assistance to the user.

As for the internal system perspective, the voice user interface should be easy to extend with newer data sources, other data transmission systems - VoIP and it has to accommodate more than 5000 concurrent users.

Depending on the user information that can be securely collected and processed, the system has to provide information which better matches his needs.

In the first assignment, our team has to analyze and understand the system, so that at the end of the period, it can produce a proper view of the requirements of the system, factor tables, issue cards and relevant strategies to solve those.[1] Solving the issues has to conclude with the creation of a conceptual view of the system, where strategies are implemented, keeping in mind that there must be traceability between issue cards and strategies.[2]

System requirements:

1. The system shall be compatible with multiple telephone line, i.e. support both analog and digital input.
2. The system shall support Text to Speech (TTS) and Automatic Speech recognition (ASR).
3. The system shall be able to playback pre-recordings when the web is not available.
4. The system shall be able to collect tone keypad.
5. The system should be able to distinguish spoken input for voice callers.
6. The system should provide voice input and output information.
7. The system shall provide a voice user interface to manage various interactions.
8. The system shall provide help to the user when required.
9. The system`s output should be meaningful to the user, and delivered in an explicit way.
10. The system shall personalize the requested content for different users.
11. The system shall speak text to users i.e., read prompts.
12. The system shall access and play media files via a highly reliable, distributed media server.
13. The system shall seamlessly be able to integrate with third-party components, thus extending the possible services that the system provides.

Quality Requirements

1. The system shall allow callers to get access to information without human intervention.

2. The system shall be able to deal with incorrect input (voice and tone).

3. The system shall be easy extendible with new types of information and sources of content, i.e., existing web services.

4. The functionality of the system should be quick to learn and easy to use.

5. The system shall support 64Kbps voice bit rate, 20 msec to 30 msec sample speech time, G711 codec.

6. The system shall be able to service 5000 simultaneous users (average) without affecting response times.

2 Assumptions and scope[*Addressing Indira's comment : Missing assumptions and scope*]

1. This system will have CTI or VoIP service provider in order to handle telephone calls.
2. The Internet connectivity is not seamless. Internet connections between different software components may be lost at different points of time and it may cause undesired failures or results.
3. The voice application for retrieving information from web must have a telephone number which users call to get information.
4. Technologies like ASR/TTS work perfectly.
5. The system will differentiate between a human input and any other inputs (noise, system output like TTS generated audio message).
6. The software system will be used by people with low computer literacy or who have no Internet accessibility. The users will take advantage of the newest technology for information retrieval using low-end phones without Internet access.
7. The places where this system will be deployed are developing countries with low communication infrastructure.
8. The system will be used initially to provide information on weather conditions, but it should be able to provide other type of information as transportation and promotional data in the future.
9. The system has also a broader scope - enabling people easy access to data through a NUI(Natural User Interface), that may be of vital importance in scenarios like user distress or emergency cases.
10. The user expects the system to respond in less than three seconds.

3 System Analysis

3.1 Preliminary Architecture Style [*Addressing Indira's comment : You claimed that you used architectural style but you did not elaborate what it is and why you selected it, and what are the alternatives.*]

The preliminary architecture style is a layer based architecture. For reasons of scalability and maintainability it has been changed to a N-Tier architecture style, with bidirectional information flow. User interface is the first layer which handles the user request. User input handler is used to handle numerous request and further passes it to application layer for processing. DTMF and ASR, which is external to our system is used to handle incoming user tone. The handler is also responsible to transmit processed information back to the user. TTS is used to convert the processed information back to speech. Application layer consists of a call session handler which splits the requests to per user requests and is responsible for their processing. There can be multiple simultaneous requests and the system has to know what to process for each request and process result accordingly, this is done by session handler. Then it is passed to business logic. This layer consists of validation manager, query manager and user profile manager. Validation manager validates the incoming request and passes it to query manager if it is valid. Query manager is responsible to form a valid query after parsing the user request so that it can be used to retrieve the information from persistence layer. The media server that is external to the system is also used to retrieve pre-stored audio data. User profile manager is responsible to manage each user information. A profile is created for every new user, which are stored in an internal database.

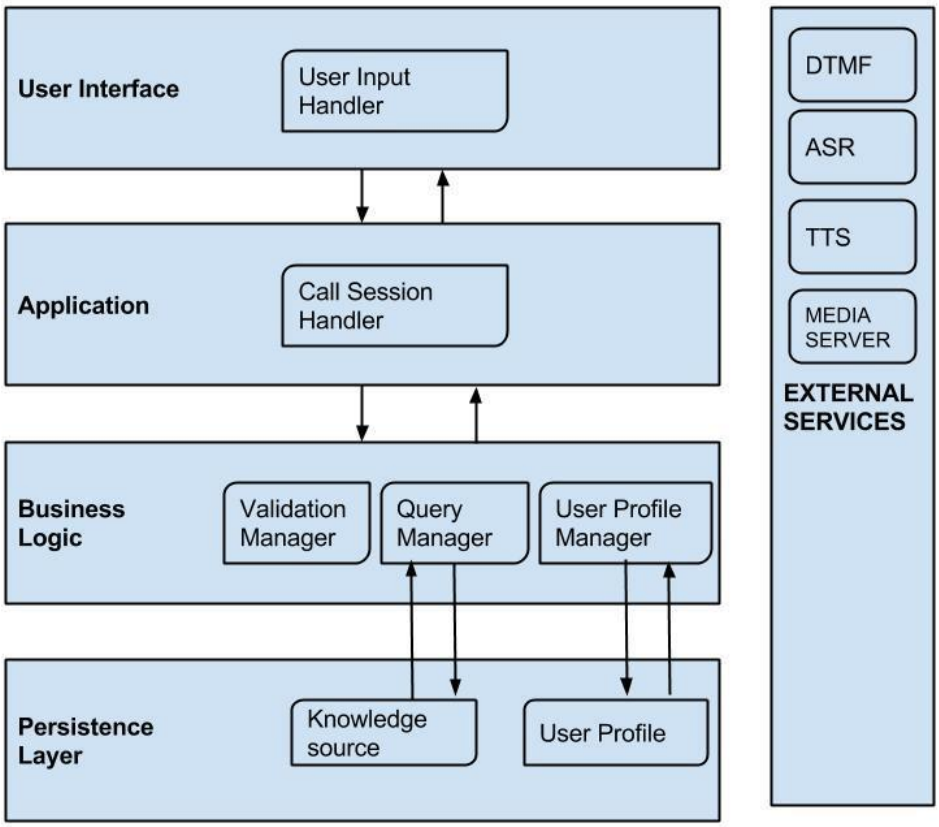
The downside of using a layered architecture style will be a decrease in the system performance, as it will add delays due to multiple functions, inter-process calls and context switches. We have considered, as alternative architecture styles, "Blackboard" and "Pipes-and-Filters".

A blackboard system is an artificial intelligence application based on the blackboard architectural model, where a common knowledge base, the "blackboard", is iteratively updated by a diverse group of specialist knowledge sources, starting with a problem specification and ending with a solution.[14] Each knowledge source updates the blackboard with a partial solution when its internal constraints match the blackboard state. In this way, the specialists work together to solve the problem. We do not use the "Blackboard"layer style, as we do know, at one moment in time, the action that needs to be executed and we do not have a partial solution to a user's request.

"Pipes-and-filters" consists of any number of components (filters) that transform or filter data, before passing it on via connectors (pipes) to other components. The filter transforms or filters the data it receives via the pipes with which it is connected. A filter can have any number of input pipes and any number of output pipes. The pipe is the connector that passes data from one filter to the next. It is a directional stream of data, that is usually implemented by a data buffer to store all data, until the next filter has time to process it.[15] We have not used "Pipes-

and-Filter", as we have considered a well defined sequence of steps for the data workflow and a part of the system has to keep track of the state(users' sessions).

Preliminary Architecture Style Diagram:



3.2 Organizational Factors[*Added more organizational factors*]

Nr.	Organizational factor	Flexibility and Changeability	Impact
O1.1	**Development budget** We have infinite financial resources.	The financial resources are not subject to change.	It has a positive impact on the amount and quality of resources consumed or used.
O2.1	**Staff skills** The staff skills are the best on the market at the moment the project starts.	The staff may encounter new software problems, which cannot be solved easily, while keeping good quality product releases.	It impacts the quality of the software, the time needed for development.
O2.2	**Employee number** We have small teams of 4-5 members, which are highly specialised on tackling different kind of software development problems.	The employee number may change depending on how much developer time is needed to develop the product.	It impacts the time needed to develop the product and also the communication strategies used, as there are major differences between how communication is handled on small teams or large teams.
O3.1	**Development time** The development time is highly restricted to one year.	The time may slightly change to accommodate minor delays in the release schedule, but no more than ±10%.	It impacts the release schedule, the effort and resources needed for the development.
O3.2	**Development resource access** We have access to the best software and hardware resources available at the moment.	The software and hardware resources evolve rapidly over time. The flexibility arises from the fact that most of the new components are backward-compatible.	It has positive impact on development time. It also impacts the staff capabilities, as the new technologies have a learning curve.

3.3 Technological Factors:[*Addressing Indira's comment :* **Missing many important factors, factors identified were ambiguous and not elaborated. Factor 4.1 is not architecture technology factor. What are most likely to change about the factors? Impact analysis should address what other components, or other design decision would be be affected the factors were to change.]**

Nr.	Technological factor	Flexibility and Changeability	Impact
T1.1	**Telecommunication providers** The telecommunication services(3G/4G, GSM) are offered to users by telecommunication companies.	The telecommunication providers services differ, depending on the geographical area, political regime, economical conditions and other heterogeneous factors.	It has a strong impact on the way the voice service is delivered(how the constraints can be tackled).
T1.2	**End device capabilities** Our service is delivered to users via a multitude of end devices(mobile phones, public phones) that can make a call or have Internet access.	The end devices evolve very rapidly and the capabilities as network access or compute performance have huge variations. The flexibility is given by the fact that the end devices must respect open standards.	It impacts the quality of services offered to the user, the amount of features able to be delivered. This factor has a high impact on whether to use CTI or VoIP technologies. If the end device is a telephone, CTI is used. If the end device is a smartphone and has Internet access, VoIP can be used along with CTI.
T1.3	**Network capabilities** The system components communicate over a network(wireless or wired). The phone communicates with the Voice Service over a Telephone network. The internal system components communicate over Internet.	Network speed and traffic amount increase over time, due to constant technological development.	The network change has a high impact on the way components can communicate. CTI can handle large number of calls. If the network performance is high(communication overhead is low). It is possible to have a component-based architecture which communicate over the network. It impacts the module view.
T2.1	**Infrastructure as a service (IAAS)**		

	availability Cloud computing allows us to rent compute/storage/memory resources when needed to expand our service delivery. We will use it to scale rapidly and frequently, based on our users demands.	Cloud-computing changes over time, getting cheaper and easier to use as the technology advances. It gives the flexibility of scaling up and down the amount of resources used.	The usage of cloud computing technology impacts the software architecture, the communication technologies and the level of abstraction used.
T3.1	**Disk** The amount of storage space needed for the application to be functional. A minimum of 1024 GB SSD or a HDD with high IO capability is needed.	The storage space needed for the software system will increase proportionally to the users' accesses of the system. The technologies do not change frequently, but get less expensive. Changeability in how access to disk is done is low, as the file-system technologies do not change often. The disks have to be replaced every two years, and data must be replicated.	It has a medium impact of architecture of the system, as different architecture patterns have to be followed if the data becomes huge. It will impact module view and execution view, as the persistence must be addressed directly or as a service. The performance of the system is impacted directly by the IO speed.
T3.2	**RAM** 4GB of RAM has been selected for development and a total of 128GB for servers with a possible increase or decrease depending on the system usage.	The amount of RAM will have to increase as the number of concurrent users increase. The RAM technologies evolve very slow when it comes to speed increase, but the price is decreasing rapidly.	It has an impact on module and execution view, as all programs use RAM to run. The throughput of the system increases dramatically if the amount of RAM available gets an increase. It also impacts the operating system and CPU factors, as there has to be a balance between these three forces.
T3.3	**CPU** 4 Intel Haswell processors Xeon E3 1200 on a rack should suffice. The chosen processor enables the use of virtualization, in case it is needed.	The tick-tock Intel cycle means that is a very large performance and efficiency improvement every two years. Accordingly, we will change all the hardware every two years.	It has a high impact on execution view and a moderate impact on module view. Having more than one core means multiprocessing on the same box is available. It impacts the maximum amount of RAM - this case 128GB.
T4.1	**Operating system** We chose a x64 Linux based operating system. The high	The Linux kernel changes	It has a high impact on

	performance of the Linux kernel is very important. The Linux flavor is less important, as the code base should be portable between various flavors(Ubuntu, CentOS). The Linux kernel also has virtualization features if needed - KVM.	daily, but the stable releases are not. The operating system will be upgraded when a newer long term support version of the kernel and flavor will be available, approximately every two years.	the hardware and software used to implement the system functionalities. It has a high impact on the system performance, portability and reliability. Being an open source system, it affects in a positive way the budget. It impacts the execution view.
T4.2	**Programming language** The programming languages of choice on Linux is C++/Python. Python code also can be easily integrated with C code.	The standards do not change (POSIX).	It affects the execution view and the module view, as Python doesn't have real multi-threading support, so processes must be used to achieve real concurrency. It also impacts performance in a positive way, C code being the most efficient.
T4.3	**Database** We can use MySQL/MSSQL/MariaDB/Postgres SQL/MongoDB/CouchDB. We can also use Database as a Service, to scale rapidly.	Databases are subject to change and rapid evolution. Flexibility comes from the fact that most languages provide layers to abstract the underlying database. The Python language can abstract all of these databases through the SQLAlchemy Layer.	It impacts the execution and might have an impact on the module view. It impacts the performance of the system and the ability to handle high load.
T5.1	**Third party information providers** We support weather information providers. Yahoo Weather API is enough for the current weather information requirements.	The information providers offer a software product, which can rapidly evolve, change or degrade over time. Other providers can outgrow over time the predominant ones. Some providers offer information just for certain areas or timeframes, with different level of quality.	It has a high impact on conceptual, module and execution view. The information providers have a direct impact on the quality of service delivered to the end user. It impacts the budget, as most of the services have to be payed.

3.4 Product Factors

Nr.	Product factor	Flexibility and Changeability	Impact
P1.1	**System response time** The perceived response time for the user should be similar to a person-to-person communication scenario. The response time cannot exceed 3 seconds.	This is subject to change, as it affects the quality of service.	It impacts the conceptual, module and execution view. It also impacts, from a business perspective, the user satisfaction.
P1.2	**System availability** The system should be available for the user with an monthly uptime percentage of 90%.	The flexibility can derive from the fact that the absence of certain features can be compensated by other features.	It impacts the module view. The perceived quality of service can degrade if some features are disabled.
P2.1	**User specific restrictions** The user has a set of legal or moral regulations which must be respected.	Depending on the geographical area, political regime or religion, the response has to be created accordingly.	It impacts the module and execution view.
P2.2	**System concurrent users** 5000 concurrent users must be supported by the system. As resource are shared between the users there might be decline in service quality.	Depending on the timeframe, unpredictable event or user habits, the system be accessed in a bursty pattern, with long periods of low activity and small periods of very intense activity.	It impacts all the views. The perceived quality of service is seriously affected, as small periods of bad service quality can have serious consequences on overall system perception.
P2.3	**User incorrect input** The human user input or even hearing/understanding system's answer are error prone.	A user can press a wrong button. The user dialect/speech can be very different from the official one.	It impacts the module view and execution view.
P3.1	**Easy to use voice interface** The interface should be easy to learn and understand. Being a voice interface, there are limitations on the way it can be implemented.	A user can have different approaches to a voice interface, depending on education or level of language understanding.	It impacts the module view and execution view. It has a high impact on the user perceived quality of service.
P4.1	**Third-party integration** The software system should tap into different information providers software services.	The third-party software services do not have a common interface, contain heterogeneous data and are subject to various restrictions as query limitations, data types, data transport protocols.	The impact is very high on the conceptual, module and execution view.

P5.1	**Fault tolerance** If some subsystem fails, it should not compromise all the system, The system should be able to recover from the fault.	The faults can come in the shape of code bugs, network related errors, third-party attacks or broken hardware.	It affects the module view and execution view. the impact is very high the user experience.
P6.1	**System response personalization** The system should produce results tailored for the user profile.	The user can have a lot of differentiating factors as gender, living area, financial or marital status, end device used or the input for the system.	It has a high impact on the module view. It also affects the marketability of the whole system, as user related data is extremely precious from a business perspective.
P7.1	**System security** The security of the system is paramount, as the information received by the user might be used in very important scenarios.	The changeability of the security issues is very high, as the two opposing forces - the hacker and the security responsible - are in permanent evolution.	It has a high impact on the module and execution view, as the vulnerabilities are commonly found in the execution view. The security of the system affects the whole organization that produced the software product, so it is very important
	Functional factors[*Addressing the small number of functional factors*]		
P8.1	**User data input acquisition** The user can input data to the system using a mobile device or a fixed telephone. The user input acquisition is done by tapping to the telecommunication provider system.	Changeability: The data can be audio or tone, maybe text. Flexibility: The data transfer medium(network link layer) is abstracted by the telecommunication provider.	It impacts the conceptual view, the execution view and module view. The way data is being introduced into the system, the data types(audio, tone, text) will impact the way it is processed.
P8.2	**User data output** The weather information data is outputted to the user via a voice interface.	Changeability: The weather information is, at first, text data. Then is translated into voice data. The way it is translated and retrieved should be flexible, using media servers, local database or the mobile device(if a smartphone is used). Flexibility: The user is not aware of the way the translation is done, so it can be implemented either way.	It impacts the performance of the system, the conceptual view, module view and execution view.

P8.3	**Weather information** Weather information has to be collected from external sources like weather information aggregators.	Changeability: The APIs exposed by weather providers may be differ in query and response type. Flexibility: The weather information has a fixed structure: temperature, pressure, wind speed, wind direction. This allows us to create a basic weather service abstraction, so that every weather provider can be seen in the same way.	It impacts the execution and module view, as the implementation for every weather provider can be different.
P8.4	**User input to weather data conversion** The data sent from the user must be processed into a weather information query.	Changeability: The data may change in terms of the language used, the type of query or the number of queries in a single request. The query may change from a standard one(which doesn't depend to a specific user), to a personalized one(according to historical data, geographical area, user type). Flexibility: The user cannot see the internal process, and the implementations can differ.	It impacts the conceptual view, module view and execution view. It has an impact on the performance, because a personalized response for every user cannot be easily cached.
P8.5	**The weather information query response** The query obtained by processing the user input in the system has to be relayed to the weather information system and an answer containing the relevant weather data has to be obtained.	Changeability: The query may return differently due to weather information provider updating their services. Flexibility: More than one weather information provider may be chosen.	It impacts the conceptual view, module and execution view. It has an impact on the quality of the service, because choosing one service over another may have different outputs.

3.5 Strategy Tables

3.5.1 Technology evolution
Issue Description Technology will change and evolve with time. The challenge is to easily upgrade to better software and hardware devices. **Influencing factors** T1.2. End device capabilities. T1.3 Network T2 Infrastructure as a service (IAAS - cloud computing) availability T3.1 Disk T3.2 RAM T3.3 CPU T4.1 Operating system T4.5 Third party information providers
Strategies: **Strategy 1: Introduce interface and components** Divide the system into components, so that change is limited to one component. A localized change will not affect the other parts of the system. **Strategy 2: Encapsulate domain specific data** Encapsulate the domain specific data so that the data flow management becomes simpler and more maintainable. **Strategy 3: Encapsulate General purpose hardware.** Encapsulate the general purpose hardware so that it hardware change doesn't have a huge impact on the system functionality. **Strategy 4: Use and respect existing standards** Standards as POSIX, code style guides and design patterns shall be used where available. **Strategy 5. Module based architecture** Design module based architecture so that each module handles a specific task. Any change then would only impact the respective module. Separation of concern can be achieved through this. [17] **Strategy 6 :Layered based architecture** Layered based architecture makes the point of impact due to technology change very clear. Usually technology change would affect a particular layer.

Related Strategies:
Strategy 7: Use of messaging queues.
This is not the case, as the system requires a close to real-time performance, and, if queues are used, messages are consumed by workers and there are no guarantees on the time taken to execute a task.

3.5.2 Information retrieval time from third-party information providers

Issue Description

The system responds to user input with information retrieved from the third party providers. The information quality and the response time will determine user perception of the entire software system.

Influencing factors

T5 Third party information providers
P4 Third party integration

Strategies:

Strategy 8: Introduce interfaces and components for retrieving information
Develop components for domain specific data and develop a component third party integration. Use abstract factories and create drivers(specific information retrieval implementation) for each content provider.

Strategy 9: Cache the information retrieved from third-parties
Add a caching mechanism, which can store data in memory(RAM) or a database. The data becomes easier to retrieve, as there is no network or third-party overhead.

Related Strategies:
Strategy 10: Retrieve and store all the information from third-parties periodically.
We do not consider this strategy, as the amount of the data stored would be very large and it becomes a bottleneck for our system in terms of performance and resource consumption. The network overhead while we retrieved data from third-parties will decrease to zero, but the lookup overhead(searching in the data stored) will increase significantly.

3.5.3 High Load

Issue Description

The software system is required to support 5000 users. Although it is not a high load, it can increase over time.

Influencing factors

T1.2. End device capabilities.
T1.3 Network
T2.1 Infrastructure as a service (IAAS) availability
T3.1 Disk
T3.2 RAM
T3.3 CPU
T4.1 Operating system

Strategies:

Strategy 11: Have the system run on powerful hardware so that it can handle huge load.
Strategy 12: Make the system distributed so that it can distribute load.[3] Split the system into components, so that load balancers can be added in front of them.
Strategy 13: Use the best algorithms available. In order to change the algorithms as they evolve, they shall be implemented as separate components. See Strategy 1.

Related Strategies:
No related strategies.

3.5.4 Error Handling
Issue The software system can and will get undesired input. These inputs may origin from various sources. One example would be requesting weather report for a place which doesn't exist . **Influencing factor** P2.3 User incorrect input
Strategies: **Strategy 14:** Add a separate module to validate user input **Strategy 15:** Fallback to a voice based menu
Related Strategies: **No related strategy**

3.5.5 Security
Issue Third-party integration introduces security issues **Influencing factor** P4 Third-party integration
Strategy 16:Employ least privilege [16] The system should be open with least privileges so that request are only entertained to the limit they should be. **Strategy 17 :Keep it simple [16]** Difficult design and architecture are hard to understand. So in order to understand and trace the sytem it is wise to keep the design simple.
Related Strategies: **No related strategy**

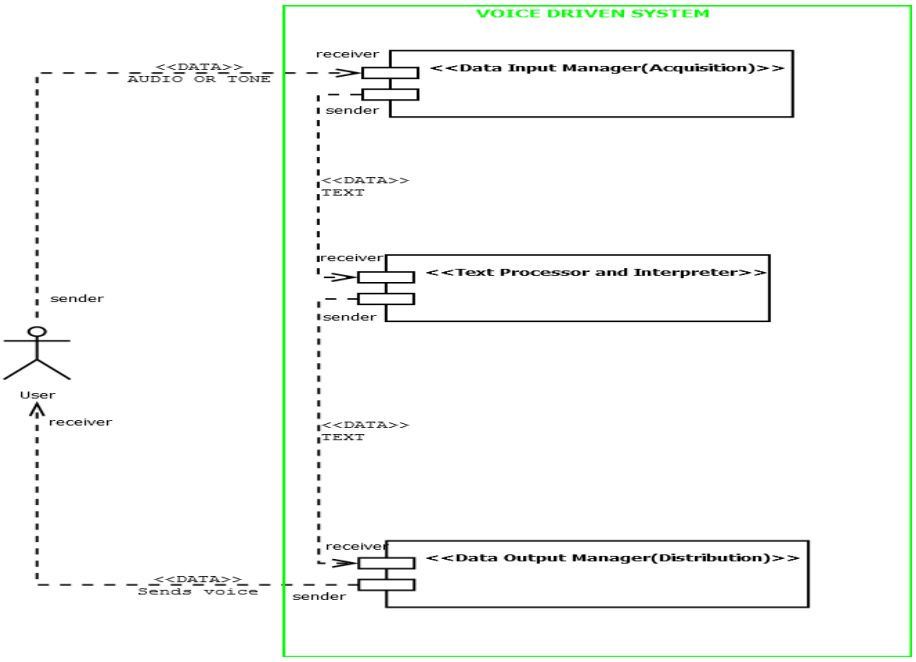
3.5.6 System dependencies to the external environment
Issue Description The system should not depend on the telecommunication providers, so that it is portable and the development should be decoupled of the existing technologies, as they change very often. **Influencing factors** T1.2. End device capabilities. T1.3 Network
Strategies: **Strategy 18:** Introduce buffer components. This is very necessary if the data flow is not continuous. It can hold data temporarily before it can moved to the other components of the system for further processing.
Related Strategies: **No related strategy**

4 Conceptual View

4.1 Strategy Mapping to Conceptual View

4.1.1 Iteration 1:

Strategy	**Component**	**Description**
Strategy 1: Introduce interface and components Strategy 18: Introduce buffer components.	Data Input Manager(Buffer)	The architecture is divided into components. Data input manager is one of the imprortant componnet.It is the data entry point of the system, which gets the data from the telecommunication system and transforms it into text, implementing automatic speech recognition or tone to text mapping. The text data it is sent to the Text Processor and Interpreter.for further processing. It is also known as buffer as it store data for certain duration before it passes for further processing.
Strategy 1: Introduce interface and components	Text Processor and Interpreter	Text processor and interpreter is one separate component. After it receives the text data from the Data Input Manager, it will process the query and return the weather information requested as a text. The text will be sent to the data Output Manager for further processing.
Strategy 1: Introduce interface and components Strategy 18: Introduce buffer components.	Data Output Manager(Buffer)	After receiving the business critical information(weather data) as text, it converts it to audio data using text to speech recognition. It then relays the audio data to the user end device, using the telecommunication system. It is also known as buffered response as it store data for some time before sending response back to the user.

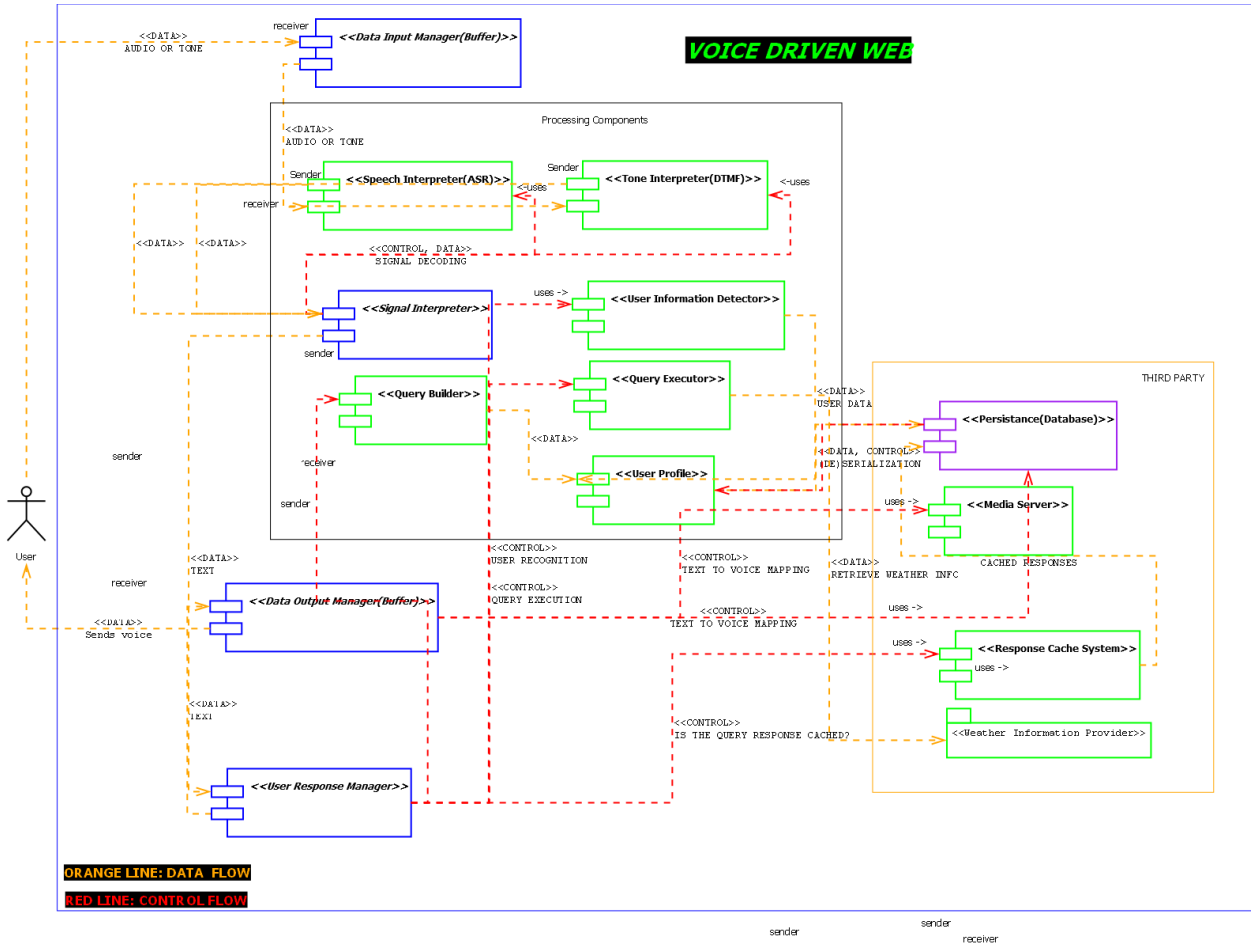


Iteration 1 Conceptual View image: http://goo.gl/41Iz3f

4.1.2. Iteration 2

Strategy	**Component**	**Description**
Strategy 18 - Introduce buffer component Strategy 1: Introduce interface and components	Data Input Manager(Buffer)	The architecture is divided into components. Data input manager is one of the improrant componnet.It is the data entry point of the system, which gets the data from the telecommunication system and transforms it into text, implementing automatic speech recognition or tone to text mapping. The text data it is sent to the Text Processor and Interpreter.for further processing. It is also known as buffer as it store data for certain duration before it passes for further processing
Strategy 9: Cache the information retrieved from third-parties	Response cache system.	Recently fetched information and/or mostly fetched information can be cached in the component.
Strategy 1: Introduce interface and components Strategy 5. Module based	Signal Interpreter	All the information related to signal processing come under signal interpreter. This module receives the text data from the Data Input Manager, it will send the data for processing to the Tone Interpreter. If the Tone Interpreter return succesfully, data is sent to User Response Manager. Otherwise, it is an audio

architecture		data and it will trigger the Speech Interpreter to process the data. The returned text is then sent to User Response Manager.
Strategy 1: Introduce interface and components	Speech Interpreter	This component will get an audio data and it will translate it into text data. Speech Interpreter is a separate component.
Strategy 17 :Keep it simple [16]	-	The division of components , modules and groups have made the architecture design simple and elegant. As a result it can be easily understood by all.
Strategy 1: Introduce interface and components	Tone Interpreter	The component will decide if the data received is tone and it will translate the data into text. It is a component too.
Strategy 1: Introduce interface and components Strategy 5. Module based architecture	User Response Manager	All the things that has to be done for response are taken care by response manager. This module willl handle the interaction between User Information Detector, Query Builder, Query Executor, Response Cache System. It uses the components depending on the user input. After it gets the weather information data from the cache system, or the query executor, it relays it to the Data Output Manager.
Strategy 18 - Introduce buffer component	Data Output Manager	After receiving the business critical information(weather data) as text, it converts it to audio data using text to speech recognition. It then relays the audio data to the user end device, using the telecommunication system. It is also known as buffered response as it store data for some time before sending response back to the user.



https://github.com/santoshshah1/PA1410-assignment/blob/master/assignment-sources/12-13-2013/Conceptual-View-1.png

Iteration 2 Conceptual View image

4.2 Conceptual View

The conceptual view is focused on a first top view of how the system can be designed/implemented taking into consideration the requirements.

All the components involved will respect the SPR and will thought of as software services - this way changes in one module should not interfere with the prospect of changes in the other ones.[5][6]

The following components are thought of to reach the goal:

1. The mobile device which the user has. It is the user hardware interface. From our perspective, it is a black box which has memory, the ability to send data and receive data

to/from the telecom company system. The data may be transmitted using basic telecom protocols or VoIP.[7]

2. The telecom company system who provides communication services. It is a black box which relays data from the user mobile phone and back. The data is transferred using telecom/VoIP. This black box can abstract the method used. The interface provides methods to get raw user data and send to the user raw data in the same single session.
3. The Automatic Speech Recognition system should take as input the raw data from the telecom system and transform it into relevant text. The string result shall be passed to the Text Interpreter system.[8]
4. The Text To Speech shall take as an input a string, transform it in raw audio data and pass the result to the telecom system input stream.[9]
5. The Text Interpreter system shall receive text and map it to relevant data using the knowledge sources available. The string result shall be sent to the Text To Speech system.
6. Knowledge sources - these sources may be local or Internet based data sources of relevant information. These should have a common interface for querying data. Example: weather information providers, medical assistance, etc.

The components above must exchange messages that can be differentiated between different user sessions. It may be the case that between the Telecom system and TTS/ASR should be a load balancer.

List of abbreviations:

PSTN - Public Switch Telephone Network
CTI - Common Telephone Integration
DTMF - Dual Tone Multiple Frequency
TTS - Text To Speech
ASR - Automatic Speech Recognition
SRP - Single Responsibility Principle

Alternatives to the design

1. Distributed architecture can be replaced by centralised powerful server(powerful hardware). It makes much sense when the product doesn't have a large number of users. Our product doesn't use it because we think it is very likely that the number of user will increase over time.
2. The product would have been much more efficient without the use of interfaces, module based architecture, SRP principles etc. Tightly coupled system are the the most efficient systems . But it is very hard and requires a lot of effort to accommodate change in this

system. Changes are inevitable and, besides that, ASR has not yet reached its maturity and it is bound to change more than others.[13]

3. The product will be more robust if there is a system administrator management interface, so that an administrator can change factors like allowing access only to some users depending on the country, phone number or make runtime knowledge source mappings to certain users etc.

7.2 Conclusion drawn

Though the architecture method was very short and was performed with limitations like lack of resource , we were able to find out the weak spot in the architecture. Assignment descriptions played a major role in creating scenarios

Designing the software architecture of the system above has given us a very good insight on the problems that we have to solve to ensure the system may evolve over time, is easy to maintain, and above all, meets the user requirements.

Coming up with a very good solution may interfere also with the limited time and also with the financial resources at our disposal.

A good architecture can evolve to a better one, over time, if we get better knowledge of the system and user requirements - it comes with experience.

References

[1]. L. Bass, P. Clements, and R. Kazman. Software Architecture in Practice, Third Edition. Addison-Wesley Publishing Co., Reading MA, 2012.

[2]. C. Hofmeister, R. Nord, and D. Soni. Applied Software Architecture. Addison-Wesley, Reading MA, 2000.

[3]. Load balancer you need to know by Keneeth Hess. http://www.serverwatch.com/trends/article.php/3937981/5-Load-Balancers-You-Need-to-Know.htm . Last accessed : 09-12-2013.

[4]. Application Programming Interface (API) By David Orenstein.

http://www.computerworld.com/s/article/43487/Application_Programming_Interface . Last accessed : 11-12-2013.

[5]. Single responsibility principle. http://www.oodesign.com/single-responsibility-principle.html . Last accessed : 12-12-2013.

[6]. Service Oriented Architecture : SOA Features and Benefits http://www.opengroup.org/soa/source-book/soa/soa_features.htm . Last accessed : 12-12-2013.

[7]. Common VOIP architecture. http://www.titandatacom.com/attvoip.pdf .Last accessed : 12-12-2013.

[8].Voice Recognition, Author: Jim Baumann

http://www.hitl.washington.edu/scivw/EVE/I.D.2.d.VoiceRecognition.html Last accessed : 12-12-2013.

[9] Allen, Jonathan; Hunnicutt, M. Sharon; Klatt, Dennis (1987). *From Text to Speech: The MITalk system. Cambridge University Press.*

[10] Public switched telephone network. http://en.citizendium.org/wiki/Public_Switched_Telephone_Network .Last accessed : 12-12-2013.

[11] Speech technology in telephone. http://www.dmoz.org/Computers/Speech_Technology/Telephony/ .Last accessed : 10-12-2013.

[12] Voice frequency. http://www3.alcatel-lucent.com/bstj/vol39-1960/articles/bstj39-1-235.pdf .Last accessed : 12-12-2013.

[13] Speaker independent connected speech recognition.

http://www.fifthgen.com/speaker-independent-connected-s-r.htm .Last accessed : 11-12-2013.

[14] Black boards.
ftp://reports.stanford.edu/pub/cstr/reports/cs/tr/86/1123/CS-TR-86-1123.pdf .Last accessed : 11-12-2013.

[15] Pipe and filter.
http://www.dossier-andreas.net/software_architecture/pipe_and_filter.html .Last accessed : 11-12-2013.

[16] Security Architecture and Design/Systems Security Architecture
http://en.wikibooks.org/wiki/Security_Architecture_and_Design/Systems_Security_Architecture .Last accessed : 12-12-2013.

[17] Separation of concern.
http://en.wikipedia.org/wiki/Separation_of_concerns Last accessed : 12-12-2013.