Activity Context-Aware System Architecture for Intelligent Natural Speech Based Interfaces

Genoveva Galarza Heredero, Harsha Penmetsa, Vikas Agrawal, Lokendra Shastri
Centre for Knowledge Driven Intelligent Systems, Enterprise Technology Research Labs
Infosys Limited, Hyderabad, India
genoveva_g@infosys.com, harsha_penmetsa@infosys.com, vikas_agrawal@infosys.com, lokendra_shastri@infosys.com

Abstract

We propose a reference architecture for intelligent context-aware natural speech-enabled systems delivering complex functionality, with direct access to information, simplifying business processes and activities while providing domain-specific and task-specific depth in interactive banking, insurance, wealth management, finance, clinical, legal, telecom customer service, operations, supply chain, connected living room, and personal assistants. This system understands not just words, but intentions, and context of the interaction. We accomplish this through a marriage of speech recognition with advanced natural language processing techniques, scalable inference and semantic technologies. This architecture is expected to dramatically improve the quality of proactive decision support provided by virtual agents by enabling them to seek explanations, make predictions, generate and test hypothesis and perform what-if-analyses. The system can provide extreme personalization (N=1) by inferring user intent, making relevant suggestions, maintaining context, carrying out cost-benefit analysis from multiple perspectives, finding similar cases from organization and personal episodic memory before they are searched for, finding relevant documents and answers, and issues resolved by experts in similar situations. The architecture enables meaningfully correlating, finding and connecting people and information sources through discovery of causal, temporal and spatial relations. We present examples of demonstrations of concept in banking and insurance that we are in the process of developing.

Next Generation Digital Workspace Architecture

In the AAAI'12 workshop on Activity Context Representation we proposed a general purpose architecture (Agrawal, Heredero et al. 2012) for activity context-aware digital workspaces which provides assistance to users of existing electronic computing based assistive devices. That architecture integrated a set of native systems (operating system, productivity suites, etc.) used by the knowledge workers in their regular course of work. Working on a layer above these, a set of application-specific context capture plugins would integrate these systems with a Context Working Memory and a Guidance Engine that would use technology such as a scalable Inference Engine, Semantic Extraction Engine, Action Recognition and Simulation System to analyze the work or actions being performed by the user and offer guidance and assistance. This work is expected to build on previous long term research on CALO (Tur, Stolcke et al. 2008) led by SRI (part of which was spun-off as Siri (Gruber 2009) later popularized by Apple Inc.), HALO (Friedland, Allen et al. 2004) at Vulcan Inc., Whodini (Mehra 2012) and Cyc (Panton, Matuszek et al. 2006, Lenat, Witbrock et al. 2010).

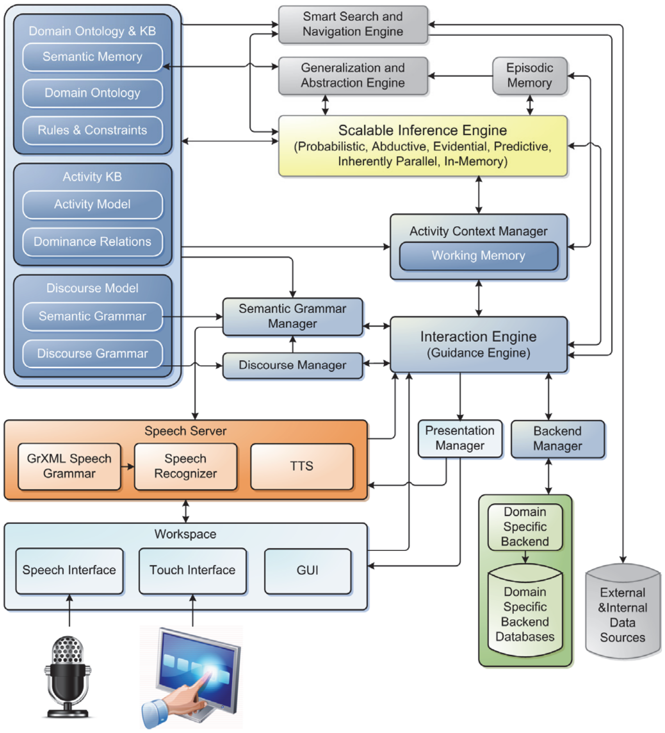


Figure 1. System Architecture for Intelligent Context-aware Speech-Enabled Natural Interfaces

Intelligent Natural Speech Based Interfaces Architecture

Extending that approach (Figure 1) we have created a customer friendly intelligent system that can provide information as well as assistance throughout activities in specific domains. We aim to create an intelligent system that is as human friendly as possible and that can be easily accessed and used by different types of users without any previous training. For that we combine speech and touch technologies with our previously demonstrated architecture for activity context-aware digital workspaces.

Through the following section we detail the key components of this new architecture and we describe one example of the application of this architecture with Moni, a Natural Speech Based Banking System and Alice, an Interactive Insurance System. The videos of this and other demonstration systems are available as supplementary materials at http://activitycontext.org/ms/ (requires credentials obtainable by emailing activitycontext@infosys.com)

Workspace: Speech Interface, Touch Interface, Graphic User Interface

The workspace module incorporates three different components that will enable the multimodal interaction between user and machine.

The **Speech Interface** enables the user to interact with the kiosk using voice commands, both simple one word commands, such as “Cancel” or “Modify” at any point of the conversations, or more elaborate sentences such as “I would like to open a savings account, please.” The system has activity models and interaction engines in place to direct the conversation in order to seek the necessary data from the user, while accommodating the possible conversational variations that a human can engage in. For example, the kiosk in a travel application should be able to understand that “I am flying to Toronto”, “to Toronto, please”, “make Toronto the destination” or “uhm, Toronto, ehhmm… yeah, that’s it” are four different valid ways of defining the destination point during a flight purchase operation.

The **Touch Interface** enables an additional path to specify actions or fill information fields for situations in which speech can be difficult to manage or simply not appropriate. For example, in the case of a banking kiosk, during the authentication of the user, the customer might not be too keen on speaking out his password or his ATM pin before a withdrawal operation in a kiosk situated in a public space. The touch interface will also facilitate the navigation among different options, letting the user press the screen to select among a set of possible options, like “Login into personal account”, “Request information” and “Open an account” in a banking related kiosk.

The **Graphical User Interface** offers a visual follow-up of the actions being performed by the system and the conversation between the user and the machine. In case of activities that require a large amount of information to be communicated to the customer, complex data or confidential data – such as the balance of one’s savings account or a detailed table with the latest transactions, that information will be displayed on the screen instead of being communicated through speech.

Speech Server

A typical speech recognition system incorporates the following components: The **Speech Server**, which acts as a central controller for the input and output speech processing resources, the **Speech Recognizer**, which using an SRGS grammar performs recognition on the audio data received from the Speech Server and generates a text string and a set of variables or tags with semantic information, and the **Vocalizer** or **Text-To-Speech Engine**, which given a text string, generates a synthesized audio output.

Speech applications communicate with the Speech Server using Media Resource Control Protocol (MRCP) and Session Initiation Protocol (SIP). A session is initiated using SIP prior to the speech recognition or synthesis. During the process of speech recognition, once the Speech Recognizer loads the assigned grammars (set via an MRCP request) it is ready to handle any recognition requests. The Speech Server receives the user’s speech using the Real-time Transport Protocol (RTP) and sends it to the recognizer, which performs the recognition, detects the end of the speech once the user stops speaking, and finally, returns the results through the Speech Server to the speech application.

The Text-To-Speech engine allows the system to generate speech prompts and responses from text strings. In a manner similar to the recognition process, the text to be synthesized to audio is sent to Speech Server in the form of an MRCP request. The TTS engine takes the text from the Speech Server and returns the synthesized audio. The Speech Server then sends this audio data to the speech application using RTP.

For this work we have used Nuance’s Speech Server version 6.2, Nuance Recognizer version 10.2 and Nuance Vocalizer for Network version 5.7.

Activity Context Manager and Working Memory

As soon as the activity that is being performed or needs to be performed has been detected by the Interaction Engine, the Activity Context Manager gets the activity specific information from the Domain Ontology, identifies the Information Needs, identifies the Priority Information using the Dominance Relations, and asks the Interaction Engine to seek this particular information.

When the information has been provided, the Activity Context Manager gets the activity specific information from the Domain Ontology and checks using constraints and rules whether the provided information enables the activity to be performed or not.

During this knowledge work, the System captures, stores, and replays the context of the activity being performed, as well as the context information of previous tasks and activities. During this performance, the Working Memory is in charge of storing in a structured form all the information relative to the process and its context, and making it available to other modules to access and modify this information.

The Working Memory contains the following necessary information:

- Initially, at the beginning of an interaction with a certain customer, the Working Memory gets populated with necessary information from previous interactions, general profile and local information, statistics, and data that can affect future interactions.
- While the user interacts with the Kiosk, the Working Memory stores all procedural information about the discourse being performed, its current state, the different dialog interactions and its responses.
- The Activity Working Memory takes care of storing all context information about current activity and previous activities being performed, the temporal location of the performance, and their input and output artifacts.
- If the customer is logged in, recognized or authenticated, the Working Memory contains all information regarding the user, important information about his previous transactions, both between customer and kiosk and between customer and bank.
- The Working Memory also stores internal procedural and contextual information, keeping track of all the internal actions and decisions of the Context-Aware System (e.g. which artifacts or modules interacted or procured information)

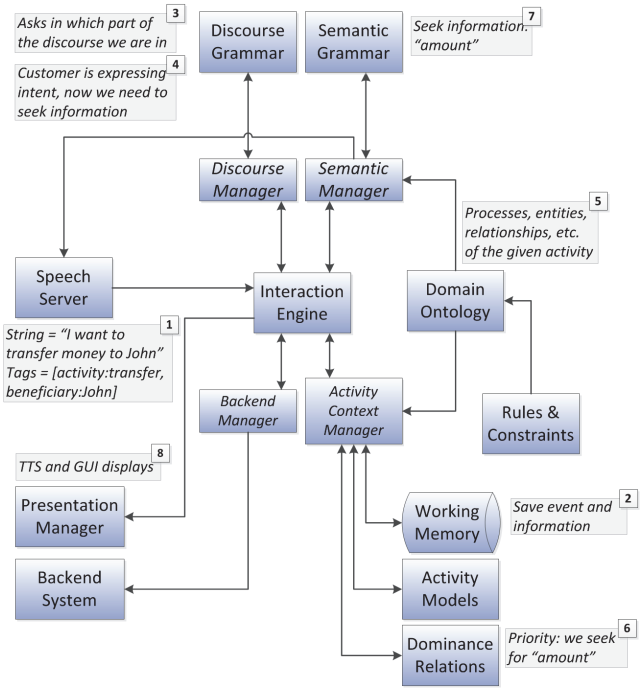

Asks in which part of the discourse we are in
3
Discourse Grammar
Semantic Grammar
Seek information: "amount"
7
Customer is expressing intent, now we need to seek information
4
Discourse Manager
Semantic Manager
Processes, entities, relationships, etc. of the given activity
5
Speech Server
Interaction Engine
Domain Ontology
String = "I want to transfer money to John" Tags = [activity:transfer, beneficiary:John]
1
Backend Manager
Activity Context Manager
Rules & Constraints
TTS and GUI displays
8
Presentation Manager
Working Memory
Save event and information
2
Backend System
Activity Models
Dominance Relations
Priority: we seek for "amount"
6

Figure 2: The Central Controller Functions of the Interaction Engine During a User Assistance Activity

All these events are centrally managed by the Interaction Manager and subsequently organized and stored in a structured form in the Working Memory along with related information.

Interaction Engine

The Interaction Engine takes a crucial part by being the central controller of the system, taking care of the storage of the context and procedural information in the Working Memory, as well as managing the interaction with all the different modules.

Initially the Interaction Engine receives the data of the events which take place in the Workspace, both received from the Touch Interface and from the Speech Interface and processed through the Speech Server.

As illustrated in Figure 2, once a particular input from the user is received and stored in the Working Memory, the Interaction Engine uses the Discourse Manager to analyze the state of the discourse between system and user and updates the Working Memory with the identified state in the Discourse Model.

After this discourse state is updated, the Interaction Engine engages the Activity Context Manager in the analysis of the current activity, determining the information needs of this activity, the already existent information and the prioritized pieces of information which the system needs to seek out in order to complete the current activity successfully.

After the Interaction Engine has sorted out what is needed from the user, it uses the Semantic Manager to generate the necessary prompts to display on the screen via GUI or in the shape of automatically generated voice. At this point, the Semantic Manager gets the Speech System ready for the expected input from the customer, by generating the necessary Semantic Grammar.

Finally, in cases where information, constraints or process details need to be fetched from a backend system, the Interaction Engine is also responsible of communicating with the Backend Manager and managing the outputs from the backend sources.

Presentation Manager

Once the Interaction Engine has taken care of generating the necessary assistance and producing the output and the prompts with the help of the Semantic Grammar, this output is sent to the Presentation Manager. This module administrates these messages and sends them to the appropriate output module for it to be displayed.

This result is presented to the user in two different forms: an abbreviated version of the output is sent to the Speech Server and the TTS module synthesizes the speech. However, this output may contain extensive or confidential details in response to the customer request, which are presented in a more elaborate way through the Workspace GUI. The process of finding optimal presentation methods for maximizing the quality of user experience continues to be an interesting area of research and development.

Discourse Manager and Grammar

For the system to be able to understand the intention of the user and track the steps till the achievement of goals of a particular request, all the possible steps of this interaction are represented in the Discourse Grammar in a domain-independent manner.

Any assistance task can be represented as the interaction of two agents, an assistant A – in this case the virtual intelligent assistant – and a customer C, a human agent, where C has a goal that he wants to achieve and A knows how to assist C in achieving it. Each agent knows certain parts of information that they need to share in order to reach the completion of this goal:

- C knows exactly what the goal is, and has knowledge about some of the necessary parameters to accomplish this goal. However, C has no knowledge about the process to follow and probably doesn't even know what pieces of information are necessary for this process.
- A has detailed knowledge about the process and the informational entities which are necessary for each step of the process. However, to begin with, A doesn't know what C's goal is, and does not have necessary pieces of information that only C can provide.

Now, the only task of A is to help C, and for that A needs to:

1. Find out from C what the goal is.
2. Once A knows the goal, and hence the process and necessary information to accomplish it, A needs to engage in a multi-modal dialogue with C in order to obtain this information.
3. Confirm whether the information needed for the activity is complete and the goal is achievable.
4. Carry out the action or provide guidance to C to fulfill this action.

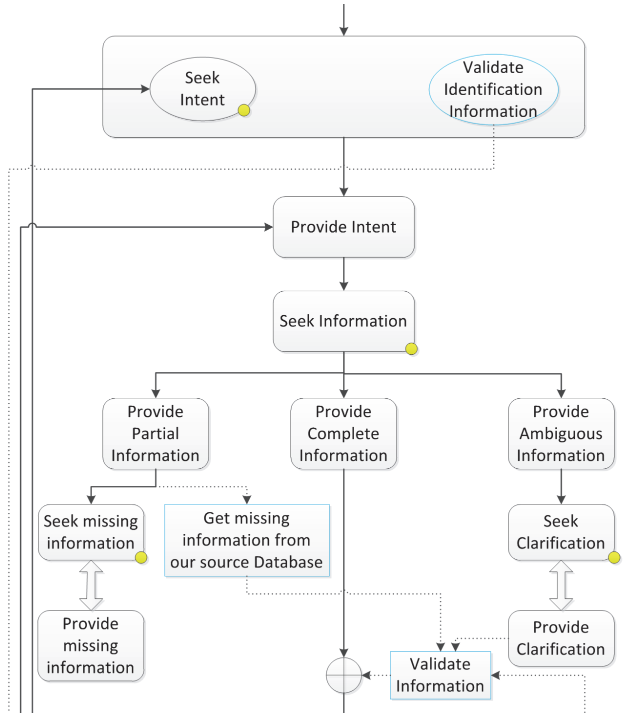

Seek Intent
Validate Identification Information
Provide Intent
Seek Information
Provide Partial Information
Provide Complete Information
Provide Ambiguous Information
Seek missing information
Get missing information from our source Database
Seek Clarification
Provide missing information
Provide Clarification
Validate Information

Figure 3: An excerpt from a Discourse Grammar state graph for "Seek Information"

The Discourse Grammar is a graph (Figure 3) that represents the possible states of the discourse between the customer (part C) and the system (in this case, part A). Each state has a unique name identifier and represents the state of an activity performed by one of the two actors. The Discourse

Grammar represents all possible states for a generic activity as well as the possible transitions between them.

For example, in the case of a simple speech assistant on a kiosk displayed in a public place, the discourse could start with a simple greeting interaction, in which the system would offer a greeting phrase and would expect the reply back from the customer to be a reply to that greeting or directly an initial intent expression. In a more complex example, the discourse state could transition to an authentication phase in which the system would ask and wait for the credentials of the user.

Another example can be seen in Figure 3, where the system state "Seek Intent" triggers the Interaction Engine to generate a particular inquiry to the customer regarding his intent. After this, the Discourse Grammar transitions to the user state "Provide Intent", while it waits for the user to manifest this intent by formulating a question or making a request. The flow of the conversation will be a sequence of transitions between system states and user states until the end of the transaction.

It is important to mention that the Discourse Grammar is completely domain independent, and any assistant – in very different contexts like banking, insurance, travel, etc. – can follow the same model as far as the speech acts related to each Discourse Grammar state are modeled in a Semantic Model.

The way we represent this state graph is with an XML document defining each state and its properties: the unique name that identifies the state, its type (whether it's an initial, final or intermediate state), the actor who performs this state (system or user), a list of the possible next states that the system can transition to, and finally a list of the necessary information slots that the system needs to take into account for uniquely identifying the speech act related to the current state.

```
<state>
   <name>seekInformation</name>
   <type>intermediate</type>
   <actor>system</actor>
   <transitions>
      <next-state>provideClarification</next-state>
      <next-state>cancel</next-state>
   </transitions>
   <properties>
      <slot>action</slot>
      <slot>information</slot>
   </properties>
</state>
```

The Discourse Manager allows the Interaction Engine to read the Discourse Grammar and manage the recording, transition and exploration of the states described in it.

Semantic Manager and Grammar

Just like the Discourse Grammar identifies the conversational state where the involved parts are in a general domain, the Semantic Grammar describes the speech acts specific to a particular domain which are used to transition from one discourse state to another.

This domain-specific module receives the dialog state, which was identified by the Discourse Manager, as well as the possible transitions to other states from the Interaction Engine, and depending on the actor involved in the future state, it generates the necessary output to feed the System.

If the previous state was performed by the assistant, the future state is to be performed by the customer. In this case, the Semantic Manager needs to generate the necessary grammar to prepare the Speech Server for the possible inputs expected from the user.

Otherwise, if the previous state was performed by the customer, the future state is an action, a reply, or a query from the assistant. The Semantic Manager understands the available information and generates the prompts which will be presented to the user via GUI and Speech.

Thus, every element described in the Semantic Grammar is directly mapped to at least one of the states defined in the Discourse Grammar.

As described above, the Semantic Grammar contains mainly two types of elements, utterances and prompts. An **utterance** is an instance of dialog that the customer can use through speech to communicate his intent, request or query, a clarification or any other piece of information, to the system. On the other hand, a **prompt** is any instance of a dialog that the system transmits to the customer via synthesized speech or visual prompting through the GUI, to provide responses, request pieces of information, ask for clarifications, inform the customer about the status of certain activities, etc., before waiting for the next input from the user.

Both utterances and prompts are of critical importance for the performance of the system, and need to be carefully designed in order to maintain the system's conversation flow. Utterances and prompts have complex dependence on each other, as in a real conversation between humans. A properly designed prompt will not only transmit the necessary information to the user, but also indirectly or directly guide him on how to provide the necessary inputs. A user whose intention is to get assistance from the system –and being conscious that the system is a machine and not a human– is expected to be a collaborative and cooperative agent and provide the needed information in an appropriate format as requested by the system.

For example if we design an assistive banking system which can help the user perform simple actions like withdrawals or transfers, after collecting the necessary information about the transaction, we will receive non-specific responses from users if we design a non-guided confirmation prompt like the following:

“You have ordered a transfer of 500 USD to John. Please confirm the operation to continue.”

The users might choose to give varied answers like *“continue”*, *“correct”* or *“confirm”*, or even *“yeah, please go ahead with the transfer”,* some of which have very clear intent and others not so clear. On the other hand, a guided confirmation prompt like the following will naturally guide the user to say the words *“yes”* or maybe *“continue”* to proceed with the operation.

You have ordered a transfer of 500 USD to John. Please say ‘Yes’ to continue or ‘No’ to cancel”

A properly designed grammar with guided prompts reduces the probability of the human customer providing feedback in a complex or unexpected way, and this will, in turn, help reduce the complexity of the utterances that need to be accounted for in the Semantic Grammar, effectively improving the overall system’s accuracy when understanding the user’s intent. Nevertheless, it is always necessary to contemplate additional ways of expression when designing the grammar of any utterance, even if the preceding prompt was clearly guided like the one in the example.

Like the Discourse Grammar, we represent the Semantic Grammar as an XML document where the different utterances and prompts are listed. The XML element `<utterance>` first references the state or states of the discourse in which this utterance can be expected from the user, identifying also the value of necessary properties. With this utterance is mapped to the discourse states, the `<srgs-syntax>` element describes the syntax of the possible phrases that the user can say during this particular state in the format of SRGS rules. After the system identifies a discourse state and the necessary properties it uses all the utterances described in the Semantic Grammar with reference to this state to put together a SRGS grammar for the Speech Server to understand the customer’s speech.

```
<utterance id="U_1">
  <reference>
    <state id="provideClarification">
      <property name="action">Transfer</property>
      <property name="information">Amount</property>
    </state>
  </reference>
  <srgs-syntax>
    <rule id="R_1">
      <item>”Transfer 500 dollars”</item>
    </rule>
  </srgs-syntax>
</utterance>
```

Similarly, the prompt element existing in the Semantic Grammar is defined with the XML tag of `<prompt>` and initially refers to the Discourse Grammar states in which this prompt will be shown. Once the state is identified we describe the syntax of this prompt by listing a set of different items from which the system can select one to display in this given situation. These slight variations of the dialog make the assistant appear more human to the user.

```
<prompt id="P_1">
  <reference>
    <state id="seekClarification">
      <property name="action">Transfer</property>
      <property name="information">Amount</property>
    </state>
  </reference>
  <prompt-syntax>
    <prompt-item>
      “How much money would you like to transfer to ”
      <entity>BENEFICIARY</entity>
      “?”
    </prompt-item>
    <prompt-item>
       “Please specify the amount to transfer”
    </prompt-item>
  </prompt-syntax>
</prompt>
```

SRGS Syntax

Speech Recognition Grammar Specification (SRGS) is a W3C standard that defines the syntax to create a speech recognition grammar. For the speech server to recognize certain sentences or structures, these have to be described in a machine-intelligible format, so the server knows what to expect the user to say. SRGS specifies two different forms:

Augmented BNF syntax (ABNF), which is a plain-text representation, and XML syntax, which in practice is the one used more frequently.

This grammar describes a set of rules and nested rule references to create all the dialectical options which the speech server should expect and recognize when analyzing the users' audio input. This grammar has the property of defining a set of *tags* which will be given the desired values when a certain rule is matched. This enables extraction of the semantic information necessary to process the speech act, instead of having to extract the semantic meaning of the words from brute-force natural language processing on a plain text string.

Backend Manager

After the Interaction Engine understands the intent of the user and collects the necessary parameters with the help of the Discourse and Semantic Managers it generates a query for the Backend Manager to access the backend system and databases. The Backend Manager responds to the Interaction with an output which contains the information to display to the user for a query, or a success or failure code, in case of a transaction request.

An Example of Interaction

To illustrate the interaction of the modules describe before, we describe the example of a simple dialog between a customer requesting a simple banking action and an assistant in the form of a Kiosk which implements our proposed architecture.

The customer has approached the Banking Kiosk and has logged in with his credentials through the touch interface. The Discourse Manager is set to its initial state and the Semantic Manager has generated an initial grammar which understands the initial expression of the customer's intent.

1. The customer speaks to the kiosk: *"I want to transfer money"*
2. The Speech Interface captures this dialog and sends the audio file to the Speech Server.
3. The Speech Server matches this dialog with the rules modeled in SRGS format and generates the string spoken by the user and the tag *"ACTION=Transfer"*
4. Both the text and the tag are sent to the Interaction Engine.
5. The Interaction Engine sends a request to the Discourse Manager to update the discourse state.
6. The state is updated from *"provideIntent"* to *"seekInformation"*
7. The Interaction Engine requests the Activity Context Manager for information regarding the requested activity and the necessary information to complete it.
8. The Activity Context Manager consults the Domain Ontology for the necessary entities for the action *"Transfer"*: *"Amount"* and *"Beneficiary"*
9. The Activity Context Manager decides which piece of information to seek initially and where. In this case this module identifies *"Beneficiary"* and communicates this to the Interaction Engine.
10. The Interaction Engine requests the Semantic Manager to generate a prompt for the current discourse state *"seekInformation"* and the parameters *"ACTION= Transfer"* and *"INFORMATION=Beneficiary"*
11. The Semantic Manager matches this information and identifies the prompt. It queries the Domain Ontology for any necessary domain or lexical information in order to generate the prompts which will be displayed to the user, both on the screen (*"You have requested to make a money transfer. Please specify to whom you would like to transfer the money"*) and via synthesized speech (*"To whom would you like to transfer the money?"*).
12. The Interaction Engine sends this prompts to the Presentation Manager, which redirects them to the GUI and the TTS module.
13. The text and the audio are presented to the customer.
14. The Interaction Engine sends a request to the Discourse Manager to update the discourse state.
15. The state is updated from *"seekInformation"* to *"providePartialInformation"*
16. The Interaction Engine requests the Activity Context Manager for information regarding the current activity, the existing information, and the necessary information to complete it.
17. The Interaction Engine requests the Semantic Manager to generate the necessary grammar to prepare the speech server for the customer's input for the discourse state of *"providePartialInformation",* with *"Transfer"* for the *"ACTION"* parameter and *"Beneficiary"* for *"INFORMATION"* parameter.
18. The Semantic Manager matches this information and identifies the utterance. It queries the Domain Ontology for any necessary domain or lexical information in order to generate a complete SRGS grammar and load it in the Speech Server.
19. The system is now prepared for receiving the requested information from the customer, and waits for a new dialog interaction.

Demonstration of Architecture Concepts and Principles

To demonstrate the previously described architecture, we created Moni, the personal assistant for the bank of the near future. Using a context-aware natural speech interface, Moni understands the customers' intent and provides them help, enabling significant savings in customer service costs at the bank. Moni provides service on interactive kiosks and smartphones.

Volunteers experimenting with the assistant found interacting with it very natural and easier than traditional interactive voice response systems. The assistant works independent of voice types and can adapt to different languages and accents by using different models in different geographies. For this experiment, Indian English, American English and Spanish were successfully implemented.

Variations in Utterance

When different users make a certain request or ask for help they use many variations in their choice of words. These variations are unpredictable, but through Moni's intelligent semantic processing, she could understand their intent and serve them effectively (Figure 4).

Devika: *Hello*
Moni: *Hello*
Devika: *Can you tell me my balance?*
Moni: *Your account balance is INR12597*
Devika: *What is my balance?*
Moni: *Your account balance is INR12597*
Devika: *Could you tell me how much money do I have in my account?*
Moni: *Your account balance is INR12597*

For a more human approach we implement different prompts and responses so Moni's responses have significant natural sounding variety.

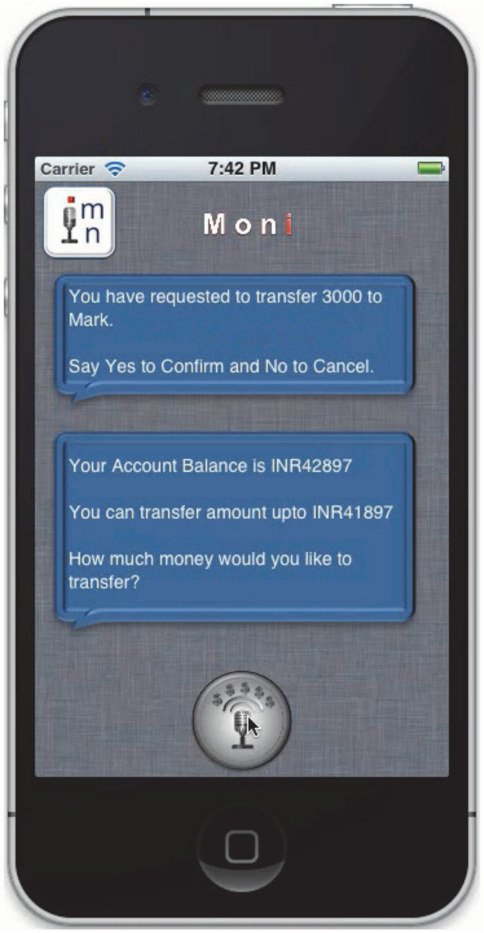
Moni: *Welcome to Bank Kiosk, how can I help you today?*
Devika: *Hello.*
Moni: *Hi.*
Devika: *Can you tell me my balance?*
Moni: *Your account balance is INR59606.*
Devika: *Can you please tell me my account balance?*
Moni: *At the moment you have INR59606.*

Modifications and Cancelations

We contemplate the possibility of customers changing their minds or accidentally giving erroneous information. For solving this problem, we implement modification and cancelation states, accessible from any other discourse state. This way, the user is able to cancel any operation easily or modify any information at any point of time.

Devika: *I'm interested in transferring some money.*
Moni: *To whom would you like to transfer the money?*
Devika: *Mark.*
Moni: *How much money would you like to transfer?*
Devika: *Sorry, make it Sanjay.*
Moni: *The new beneficiary of the money transfer is Sanjay. How much money would you like to transfer?*
Devika: *Three thousand four hundred rupees.*
Moni: *Say Yes to confirm the transfer of 3400 INR to Sanjay, and No to cancel.*
Devika: *Cancel.*
Moni: *You have cancelled the transaction. Thank you.*

Krishnan: *I need to transfer some money.*



You have requested to transfer 3000 to Mark.

Say Yes to Confirm and No to Cancel.

Your Account Balance is INR42897

You can transfer amount upto INR41897

How much money would you like to transfer?

Figure 4: Moni for iPhone interacting with a user whose intent is to transfer money

Moni: *To whom would you like to transfer the money?*
Krishnan: *Geeta.*
Moni: *How much money would you like to transfer?*
Krishnan: *Hmmm… change that to Ravi.*

Moni: *The new receiver of the money is Ravi. How much money would you like to transfer?*
Krishnan: *Two thousand rupees.*
Moni: *Say Yes to confirm the transfer of INR2000 to Ravi, and No to cancel.*
Krishnan: *Sorry, transfer nine thousand rupees.*
Moni: *No, you can only transfer up to INR8597.*
Krishnan: *Cancel.*
Moni: *Transaction cancelled. Thank you.*

Guided Information Seeking

The customer can offer the necessary information in multiple forms. While they can choose to directly convey all the information needed at once while expressing the intent to Moni, but the customer may not know exactly what data is important to communicate. Querying the Activity Model, the system knows what pieces of information are required and in what order to ask, to ensure that the information collection is complete. The following transcripts, as well as Figure 5, illustrate the cases of the customer providing the complete set of necessary information on the very first interaction, the customer providing their intent along with partial information, and finally the customer providing only their intent and the assistant seeking information for all the information slots.

Providing Intent and All Information at Once

Devika: *I want to transfer three thousand rupees to John.*
Moni: *You have enough balance in your account to transfer INR3000 to John. Please say Yes to confirm and No to cancel.*
Devika: *Yes.*
Moni: *The money has been successfully transferred.*

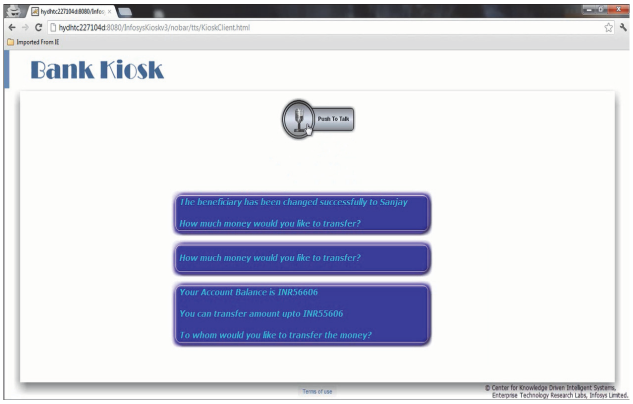
Providing Intent and Partial Information

Devika: *I would like to send some money to Mark.*
Moni: *How much money would you like to transfer?*
Devika: *Three thousand.*
Moni: *You have requested to transfer INR3000 to Mark. Say Yes to confirm and No to cancel.*
Devika: *Cancel*
Moni: *Transaction cancelled. Thank you.*
Devika: *I am interested in transferring some money.*
Moni: *Please tell me to whom should I transfer the money.*
Devika: *Mark.*
Moni: *How much money would you like to transfer?*
Devika: *No, I meant Sanjay!*
Moni: *The new receiver of the money is Sanjay. How much money would you like to transfer?*
Devika: *Three thousand four hundred rupees.*
Moni: *You have requested to transfer INR3400 to Sanjay. Say Yes to confirm and No to cancel.*
Devika: *Sorry, transfer ninety thousand rupees.*
Moni: *No. You can only transfer up to INR55606.*
Devika: *Cancel*
Moni: *Transaction cancelled. Thank you.*

Support for Various Activities

There are certain activities, like requests for information, queries, money transfers, etc., which can be supported through a smartphone interface. However, other more complex activities which may involve physical interaction (opening accounts, withdrawing or depositing money, etc.) need to be supported through physical kiosks. Moni identifies these operations and advises the user to resort to additional support when unable to provide help. In the transcript below we illustrate how these other activities can be handled.

Krishnan: *I want to withdraw some money.*
Moni: *I am unable to assist you with that request on a phone. Please visit our nearest Kiosk. Here is a map and list of our Kiosks.*


hydhtc227104d:8080/Infos
hydhtc227104d:8080/InfosysKioskv3/nobar/tts/KioskClient.html
Imported From IE
Bank Kiosk
Push To Talk
Terms of use
© Center for Knowledge Driven Intelligent Systems, Enterprise Technology Research Labs, Infosys Limited.

The beneficiary has been changed successfully to Sanjay
How much money would you like to transfer?

How much money would you like to transfer?

Your Account Balance is INR56606
You can transfer amount upto INR55606
To whom would you like to transfer the money?

Figure 5: Moni in the Kiosk version interacting with a user for a money transfer

At the Kiosk

Krishnan: *I want to withdraw some money.*
Moni: *Please specify the amount you want to withdraw.*
Krishnan: *Nine thousand rupees.*
Moni: *You want to withdraw INR9000. Say Yes to confirm and No to cancel.*
Krishnan: *Cancel.*
Moni: *Transaction cancelled. Thank you.*
Krishnan: *I wish to withdraw five hundred rupees.*
Moni: *You want to withdraw INR500. Say Yes to confirm and No to cancel.*
Krishnan: *Change that to a thousand rupees.*

Moni: *You want to withdraw INR1000. Say Yes to confirm and No to cancel.*
Krishnan: *No.*
Moni: *Transaction cancelled. Thank you.*
Krishnan: *Can I withdraw some money?*
Moni: *You can withdraw a maximum of INR55606.*
Krishnan: *Please tell me if I can withdraw some money.*
Moni: *You can withdraw up to INR55606.*
Krishnan: *I want to deposit a check.*
Moni: *Please place the check in the deposit slot. Thank you very much.*
Krishnan: *I would like to deposit seven thousand rupees.*
Moni: *Please place INR7000 in the deposit slot. Thank you very much.*
Krishnan: *Ciao.*
Moni: *Bye. Please visit again.*

The videos of this and other demonstration systems are available as supplementary materials at http://activitycontext.org/ms/ (requires credentials obtainable by emailing activitycontext@infosys.com)

Alice – The Insurance Agent

For a demonstration in a different domain, we are working on creating Alice, Insurance Agent, who can empower customers to get insurance service in fewer steps and find better deals personalized for them. Through intelligent semantic processing and inference systems, Alice will be able to proactively find information, simplify business processes and save customer service costs. The following transcript shows how Alice will be able to assist John's requests by using speech interaction and visual support on John's iPhone screen:

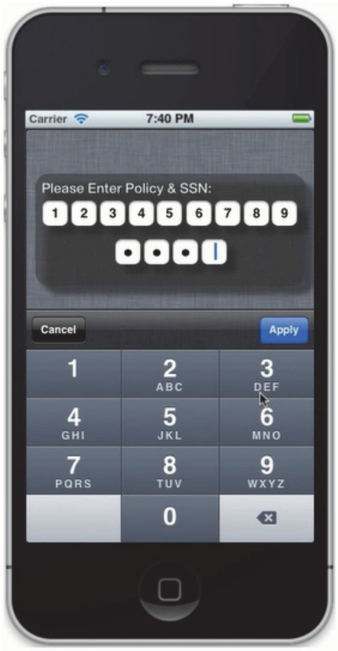
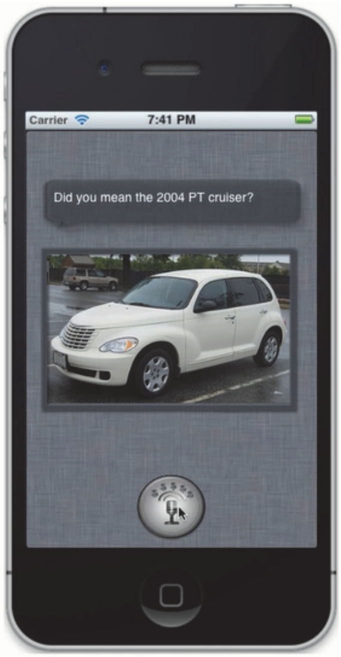


Figure 6: User login into Our Insurance application

Alice: *Welcome to Our Insurance! If you are already an Our Insurance customer, please enter your policy number and the last four digits of your Social. If you do not have this information, please wait a few seconds and I will connect you to a representative.* (See Figure 6)

Alice: *Thank you! Good morning, John. How can I help you today?*
John: *Alice, I would like to add my son Adam to my auto policy.*
Alice: *Sure, I can help you with that. Does Adam live with you?*
John: *Yes.*
Alice: *We need Adam's personal information starting with his Date of Birth, and Social Security Number.*
John: *His date of birth is November fourteenth, 1995, and Social Security is nine eight five two two four six five one.*
Alice: *Please let me know if this information is correct.*
John: *Yes.*
Alice: *And what is Adam's driver's license number?*
John: *N-200 12345678*
Alice: *Here is what I got. Is this correct?*
John: *Yes, it is.*
Alice: *Which car will he be primarily driving?*
John: *He will be driving the oldest car.*
Alice: *Did you mean the 2004 PT cruiser?* (See Figure 7)

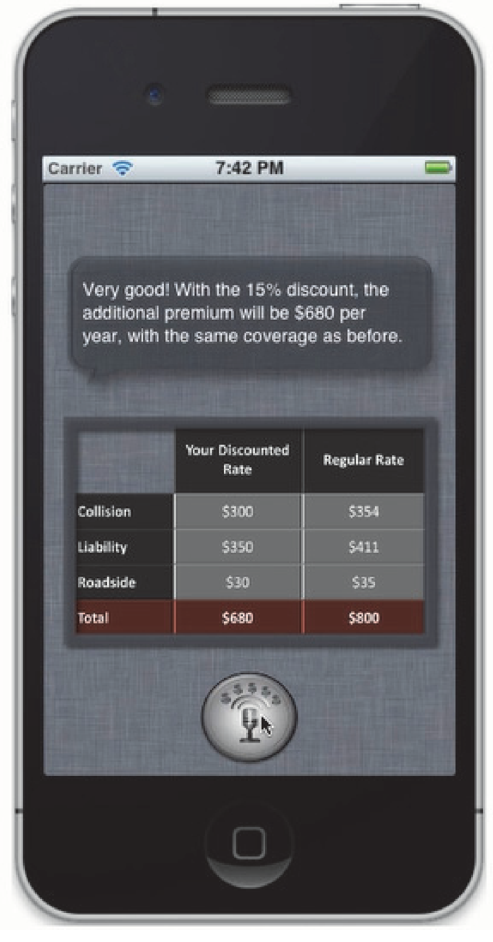


Did you mean the 2004 PT cruiser?

Figure 7: Visual Feedback for the Customer

Notice the MPIE Inference Engine based on SHRUTI (Shastri and Ajjanagadde 1993, Shastri 1999, Shastri 2007) can be used to connect database information to unstructured information through temporal inference.

John: *Yes, that's the one.*
Alice: *Great! I can get you a 15% discount if your son has a B average or better on his grades.*
John: *My son is a straight A student.*
Alice: *Very good! With the 15% discount, the additional premium will be $680 per year, with the same coverage as before. We will need a copy of your son's transcript. Should I send you an email address where you can send a copy of his last transcript?* (See Figure 8)
John: *Sure, do that.*
Alice: *Is there anything else I may help you with?*
John: *Alice, I want to save some money. I am considering dropping everything except liability on this car.*
Alice: *What is the condition of the vehicle?*
John: *The car is in a very good shape, no rust, never been in an accident.*



Very good! With the 15% discount, the additional premium will be $680 per year, with the same coverage as before.

	Your Discounted Rate	Regular Rate
Collision	$300	$354
Liability	$350	$411
Roadside	$30	$35
Total	$680	$800

Figure 8: Showing Complex Information Visually with a Short Speech Response

Alice: *You will save $300 a year by dropping collision and theft coverage. But I see that the car's replacement value is $6000. In view of this, you may want to reconsider dropping collision and theft coverage.*

Here we use the power of an inherently parallel inference engine to evaluate what-if scenarios by creating hypotheses, testing them, and maximizing the financial utility for the customer and the insurance company. We are able to question a command from the customer based on the power of this inference (Narayanan, Bailey et al. 2007, Shastri 2007) and reason on the episodic memory (Shastri In Revision).

John: *I think I'll keep the car coverage the same, I did not realize the car value was still 6000 dollars.*

We are able to make an inference that the customer does want to keep collision using the abductive inference engine (Hobbs 2004, Hobbs 2006).

Alice: *OK. We'll leave the coverage the same. Is there anything else I may help you with?*
John: *No, thanks.*
Alice: *OK. I'll add your son to the policy effective immediately.*
John: *Great.*
Alice: *Thank you very much for using Our Insurance for your insurance needs. I look forward to serving you again!*

The videos of this and other demonstration systems are available as supplementary materials at http://activitycontext.org/ms/ (requires credentials obtainable by emailing activitycontext@infosys.com)

Maturity Model for Intelligent Speech-enabled Systems for the Enterprise

Based on our experience in research and development of the novel solutions described herein, we have create a maturity model for intelligent speech-enabled systems for the enterprise (Figure 9). The four levels described in the maturity models require progressive higher levels of technology maturity and natural language processing maturity. These levels are consistent with levels of maturity needed for smart search (Figure 10) where the higher levels require the scalable inference engine shown in Figure 1.

Level 1 in our maturity model is the standard legacy DTMF tone-based phone support systems where the system is used primarily as a method of routing callers to human telephone support agents.

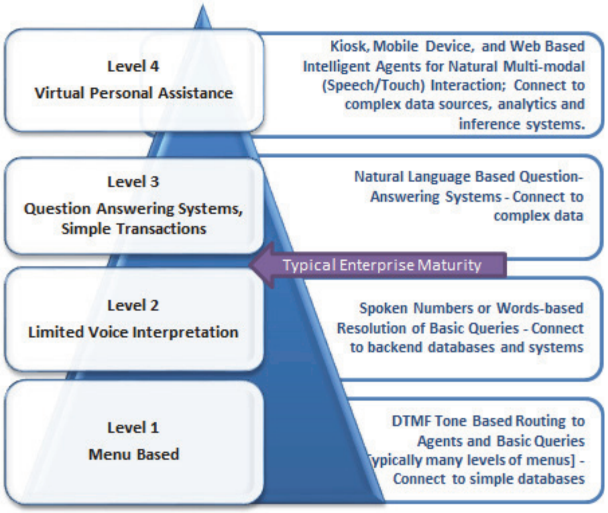

Level 4
Virtual Personal Assistance
Kiosk, Mobile Device, and Web Based Intelligent Agents for Natural Multi-modal (Speech/Touch) Interaction; Connect to complex data sources, analytics and inference systems.
Level 3
Question Answering Systems, Simple Transactions
Natural Language Based Question-Answering Systems - Connect to complex data
Typical Enterprise Maturity
Level 2
Limited Voice Interpretation
Spoken Numbers or Words-based Resolution of Basic Queries - Connect to backend databases and systems
Level 1
Menu Based
DTMF Tone Based Routing to Agents and Basic Queries
ypically many levels of menus] - Connect to simple databases

Figure 9: Maturity Model for Intelligent Speech Enabled Systems

Level 2 represents the state of the art in Interactive Voice Response systems available for phone support at most large enterprises where the user can ask questions and provide input in pre-defined formats or utterances.

Level 3 represents the upcoming wave of systems including Siri (SRI and now Apple), Nina (Nuance), Sophia (Taptera) and Lexee (Angel Labs), where natural language based questions are answered and activities of authentication, navigation and basic transactions are enabled, and there are already recommendations for going deeper and broader in scope (Markowitz 2013).

Level 4 in our maturity model represents the class of system architectures described in this paper, where multiple levels of complex reasoning are done to predict possible scenarios and deliver high value for customer and the enterprise, helping customers decide better. Such systems make intentional, causal, referential, spatial and temporal inferences using a combination of abductive (Shastri 2000, Shastri and Wendelken 2000, Garcez, Gabbay et al. 2007) and predictive, evidential and deductive reasoning, supporting what-if analysis and hypothesis generation and evaluation.

Correlation with Smart Search Maturity

The levels of maturity levels in intelligent speech-enabled systems require strong support and have direct dependence

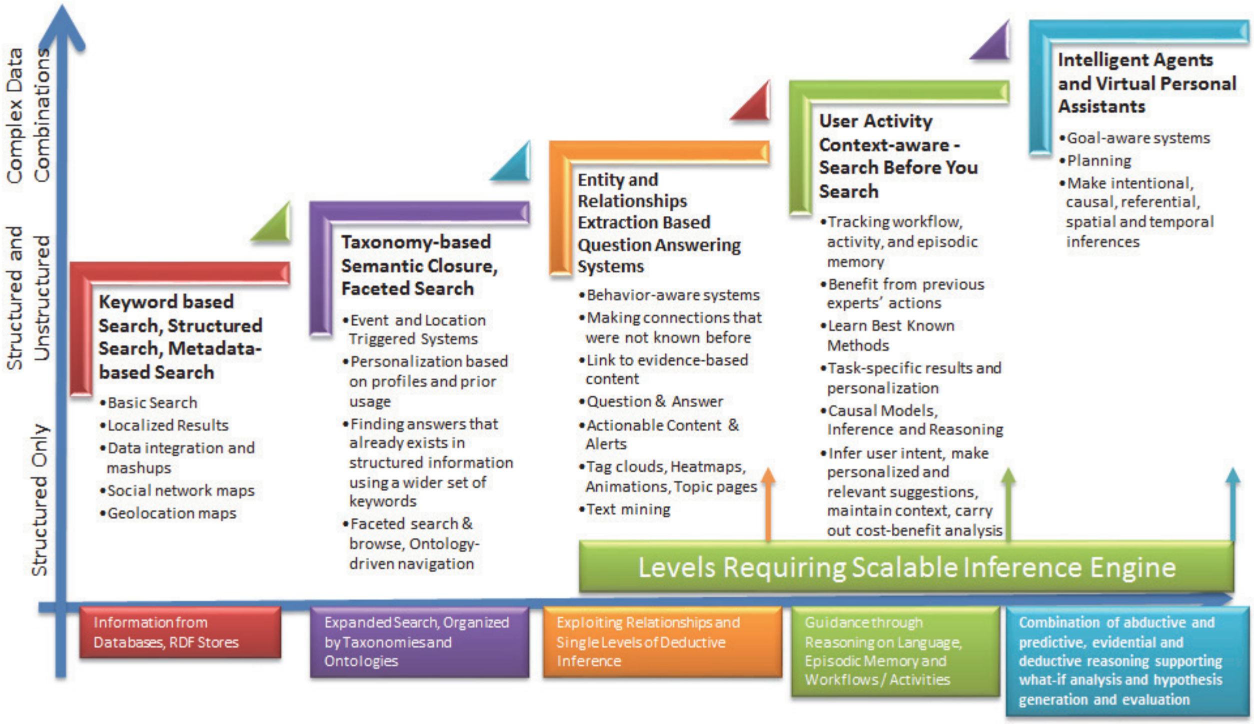

Complex Data Combinations
Structured and Unstructured
Structured Only
Keyword based Search, Structured Search, Metadata-based Search
•Basic Search
•Localized Results
•Data integration and mashups
•Social network maps
•Geolocation maps
Taxonomy-based Semantic Closure, Faceted Search
•Event and Location Triggered Systems
•Personalization based on profiles and prior usage
•Finding answers that already exists in structured information using a wider set of keywords
•Faceted search & browse, Ontology-driven navigation
Entity and Relationships Extraction Based Question Answering Systems
•Behavior-aware systems
•Making connections that were not known before
•Link to evidence-based content
•Question & Answer
•Actionable Content & Alerts
•Tag clouds, Heatmaps, Animations, Topic pages
•Text mining
User Activity Context-aware - Search Before You Search
•Tracking workflow, activity, and episodic memory
•Benefit from previous experts' actions
•Learn Best Known Methods
•Task-specific results and personalization
•Causal Models, Inference and Reasoning
•Infer user intent, make personalized and relevant suggestions, maintain context, carry out cost-benefit analysis
Intelligent Agents and Virtual Personal Assistants
•Goal-aware systems
•Planning
•Make intentional, causal, referential, spatial and temporal inferences
Levels Requiring Scalable Inference Engine
Information from Databases, RDF Stores
Expanded Search, Organized by Taxonomies and Ontologies
Exploiting Relationships and Single Levels of Deductive Inference
Guidance through Reasoning on Language, Episodic Memory and Workflows / Activities
Combination of abductive and predictive, evidential and deductive reasoning supporting what-if analysis and hypothesis generation and evaluation

Figure 10: Maturity Model for Smart Search

on maturity levels in enterprise smart search technologies available in the industry.

Level 1 is keyword based search (Page, Brin et al. 1999), including structured search, and metadata-based search where information is retrieved from databases, rank indices RDF stores, driving data integration and mashups, social network maps and geolocation maps.

Level 2 requires taxonomy-based semantic closure, and faceted search organized by taxonomies and ontologies delivering event and location triggered systems, personalization based on profiles and prior usage, finding answers that already exists in structured information using a wider set of keywords (Ding, Finin et al. 2004), ontology-driven navigation, and knowledge graphs (Bollacker, Evans et al. 2008).

Level 3 includes Entity and Relationships Extraction (Panton, Matuszek et al. 2006, Lenat, Witbrock et al. 2010, Fan, Kalyanpur et al. 2012) based Question Answering Systems (Chu-Carroll, Prager et al. 2006) that exploit relationships and machine learning to deliver behavior-aware systems, making connections that were not known before, link to evidence-based content (Ferrucci, Brown et al. 2010, Ferrucci 2012), surface actionable content & alerts with tag clouds, heat-maps, animations, and topic pages through text mining. This level of smart search can be used to support Level 3 of our Intelligent Speech-enabled Systems (Figure 9)

Level 4 includes User Activity Context-aware (Laha, Galarza et al. 2011, Laha, Shastri et al. 2012) search that provides answers before a user looks for information, by providing guidance through reasoning on language, episodic memory and workflow of activities. Such systems track workflows, activity, and episodic memory (Shastri 2001, Shastri 2001, Shastri 2002, Shastri 2007), extract benefit from previous experts' actions, enable learning of Best Known Methods (BKMs), provide task-specific results and personalization, with causal models, inference and reasoning for assisted cognition (Kautz, Etzioni et al. 2003). Level 4 systems can infer user intent, make personalized and relevant suggestions, maintain context, and carry out cost-benefit analysis. These systems can support close to Level 4 in our Intelligent Speech-enabled Systems maturity model (Figure 9)

Level 5 includes Intelligent Agents and Virtual Personal Assistants (Myers, Berry et al. 2007) which use a combination of abductive, predictive, evidential and deductive reasoning supporting what-if analysis (Murdock, McGuinness et al. 2006) and hypothesis generation and evaluation delivering goal-aware systems, with planning through intentional, causal, referential, spatial and temporal inferences . These systems support Level 4 and above in our Intelligent Speech-enabled Systems maturity model (Figure 9)

Conclusions and Summary

In this proposal we have described a detailed system architecture for activity context-aware intelligent speech-enabled systems, and illustrated it through one example implemented in banking and another example being implemented in the insurance domain. These smart workspaces and playspaces will support contextual interaction through multi-modal interfaces combining speech, touch and gesture, with context-aware services that:

1. Let us seamlessly move between applications and devices without having to remember or copy what we did earlier, without having to explicitly carry, transfer, and exchange activity context.
2. Present a framework for managing, meaningfully dividing, tracking and enabling distributed team work, connecting us to insights of experts within the organization and beyond, contextually recommending experts from within and outside an organization
3. Quickly find and show us directly related information and answers to questions based on what we mean, in the context we need it, with access to the source, quality and how the information was derived, providing offers in retail, suggestions in knowledge work and guidance in the form of personal assistants with
    a. Fast scalable proactive delivery of relevant and timely information about situations and events
    b. Provision of transparent access for users to the reasoning and data underlying the systems predictions and recommendations.
    c. Facility for the user to seek explanations for observations and perform complex what-if analysis by generating hypotheses and testing them
    d. Enable smart, semantic search through interpretation of terse (highly contextual) dialogs, commands and queries
    e. Dramatically improve the system's ability to extract information from and understand information in natural language sources such as blogs, social media etc.
4. Proactively show us steps others have taken in meaningfully similar situations before,
5. Offer proactive, task-specific and personalized advice and highlight steps others took before in meaningfully similar situations (contextual knowhow).
6. Help us reason and decide faster, with greater confidence.

We are working with our partners for creating such smart workspaces and playspaces for specific domains.

Acknowledgments

We thank Karl Nika for interesting discussions on the insurance domain agent, and his conception of the demonstration example. We thank Bhanu Verma for providing the ground work and software engineering needed for implementing the banking scenario on Windows Phone using Microsoft's APIs.

References

Agrawal, V., G. G. Heredero, H. Penmetsa, A. Laha and L. Shastri (2012). Activity Context Aware Digital Workspaces and Consumer Playspaces: Manifesto and Architecture.

Bollacker, K., C. Evans, P. Paritosh, T. Sturge and J. Taylor (2008). Freebase: a collaboratively created graph database for structuring human knowledge. Proceedings of the 2008 ACM SIGMOD international conference on Management of data, ACM.

Chu-Carroll, J., J. Prager, C. Welty, K. Czuba and D. Ferrucci (2006). A multi-strategy and multi-source approach to question answering, DTIC Document.

Ding, L., T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. Doshi and J. Sachs (2004). Swoogle: a search and metadata engine for the semantic web. Proceedings of the thirteenth ACM international conference on Information and knowledge management, ACM.

Fan, J., A. Kalyanpur, D. Gondek and D. Ferrucci (2012). "Automatic knowledge extraction from documents." IBM Journal of Research and Development **56**(3.4): 5: 1-5: 10.

Ferrucci, D. (2012). "Introduction to "this is watson"." IBM Journal of Research and Development **56**(3.4): 1: 1-1: 15.

Ferrucci, D., E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg and J. Prager (2010). "Building Watson: An overview of the DeepQA project." AI magazine **31**(3): 59-79.

Friedland, N. S., P. G. Allen, G. Matthews, M. Witbrock, D. Baxter, J. Curtis, B. Shepard, P. Miraglia, J. Angele and S. Staab (2004). "Project halo: Towards a digital aristotle." AI magazine **25**(4): 29.

Garcez, A. S. d. A., D. M. Gabbay, O. Ray and J. Woods (2007). "Abductive reasoning in neural-symbolic systems." Topoi **26**(1): 37-49.

Gruber, T. (2009). Siri, a Virtual Personal Assistant—Bringing Intelligence to the Interface, Jun.

Hobbs, J. R. (2004). "Abduction in natural language understanding." Handbook of Pragmatics: 724-741.

Hobbs, J. R. (2006). "The Origin and Evolution of Language: A Plausible, Strong-AI Account." Action to Language via the Mirror Neuron System: 48-88.

Kautz, H., O. Etzioni, D. Fox, D. Weld and L. Shastri (2003). "Foundations of assisted cognition systems." University of Washington, Computer Science Department, Technical Report, Tech. Rep.

Laha, A., G. Galarza and V. Agrawal (2011). "Support Systems for Knowledge Works: A perspective of future of Knowledge Management Service."

Laha, A., L. Shastri and V. Agrawal (2012). Activity-context Aware Computing for Supporting Knowledge-works. AAAI'12 Workshop on Activity Context Representation: Techniques and Languages. Toronto, Canada, AAAI Press.

Lenat, D., M. Witbrock, D. Baxter, E. Blackstone, C. Deaton, D. Schneider, J. Scott and B. Shepard (2010). "Harnessing Cyc to Answer Clinical Researchers' Ad Hoc Queries." AI Magazine **31**(3): 13-32.

Markowitz, J. A. (2013). Beyond SIRI: Exploring Spoken Language in Warehouse Operations, Offender Monitoring and Robotics. Mobile Speech and Advanced Natural Language Solutions, Springer**:** 3-21.

Mehra, P. (2012). "Context-aware computing: beyond search and location-based services." Internet Computing, IEEE **16**(2): 12-16.

Murdock, J. W., D. L. McGuinness, P. P. da Silva, C. Welty and D. Ferrucci (2006). Explaining conclusions from diverse knowledge sources. The Semantic Web-ISWC 2006, Springer**:** 861-872.

Myers, K., P. Berry, J. Blythe, K. Conley, M. Gervasio, D. L. McGuinness, D. Morley, A. Pfeffer, M. Pollack and M. Tambe (2007). "An intelligent personal assistant for task and time management." AI Magazine **28**(2): 47.

Narayanan, S., F. D. D. Bailey, D. Jurafsky, L. Shastri and R. W. I. Would (2007). "A Computational Model of Metaphoric Reasoning About Action and Event Descriptions."

Page, L., S. Brin, R. Motwani and T. Winograd (1999). "The PageRank citation ranking: bringing order to the web."

Panton, K., C. Matuszek, D. Lenat, D. Schneider, M. Witbrock, N. Siegel and B. Shepard (2006). Common sense reasoning–from Cyc to intelligent assistant. Ambient Intelligence in Everyday Life, Springer**:** 1-31.

Shastri, L. (1999). "Advances in Shruti—A neurally motivated model of relational knowledge representation and rapid inference using temporal synchrony." Applied Intelligence **11**(1): 79-108.

Shastri, L. (2000). Types and quantifiers in SHRUTI–a connectionist model of rapid reasoning and relational processing. Hybrid Neural Systems, Springer**:** 28-45.

Shastri, L. (2001). "A computational model of episodic memory formation in the hippocampal system." Neurocomputing **38**: 889-897.

Shastri, L. (2001). "From transient patterns to persistent structure: A model of episodic memory formation via cortico-hippocampal interactions." Behavioral and Brain Sciences.

Shastri, L. (2002). "Episodic memory and cortico-hippocampal interactions." Trends in cognitive sciences **6**(4): 162-168.

Shastri, L. (2007). "A neural architecture for reasoning, decision-making, and episodic memory: Taking a cue from the brain." Proceedings of the 3rd International workshop on neural-symbolic learning and reasoning, NeSy **7**.

Shastri, L. (2007). "SHRUTI: A neurally motivated architecture for rapid, scalable inference." Perspectives of Neural-Symbolic Integration: 183-203.

Shastri, L. (In Revision). "From Transient Patterns to Persistent Structures: A model of episodic memory formation via cortico-hippocampal interactions, ." Behavioral and Brain Sciences.

Shastri, L. and V. Ajjanagadde (1993). "From simple associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony." Behavioral and Brain Sciences **16**: 417-417.

Shastri, L. and C. Wendelken (2000). Seeking coherent explanations—A fusion of structured connectionism, temporal synchrony, and evidential reasoning. Proceedings of the Twenty-Second Conference of the Cognitive Science Society, Philadelphia, PA, ed. LR Gleitman & AK Joshi.

Tur, G., A. Stolcke, L. Voss, J. Dowding, B. Favre, R. Fernandez, M. Frampton, M. Frandsen, C. Frederickson and M. Graciarena (2008). The CALO meeting speech recognition and understanding system. Spoken Language Technology Workshop, 2008. SLT 2008. IEEE, IEEE.