AFRL-IF-RS-TR-1998-229
Final Technical Report
January 1999

MACHINE EXPLOITATION OF LANGUAGE AND VOICE INTEGRATION (MELVIN)

BBN Technologies

Marie Meteer

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-1998-229 has been reviewed and is approved for publication.

APPROVED:

DANIEL A. VENTIMIGLIA
Project Engineer

FOR THE DIRECTOR:

JOSEPH CAMERA, Deputy Chief
Information and Intelligence Exploitation Division
Information Directorate

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1999	Final Jun 96 - Oct 98

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
MACHINE EXPLOITATION OF LANGUAGE AND VOICE INTEGRATION (MELVIN)	C - F30602-96-C-0019 PE - 62702F PR - 4594 TA - 16 WU - 56
6. AUTHOR(S) Marie Meteer	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Contractor: BBN Technologies 70 Fawcett Street Cambridge MA 02138 Sub Contractor: GTE Government Systems Corporation Electronic Defense 1450 Academy Park Loop Colorado Springs CO 80910	8. PERFORMING ORGANIZATION REPORT NUMBER N/A
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/IFED 32 Brooks Road Rome NY 13441-4114	10. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-1998-229

11. SUPPLEMENTARY NOTES

Air Force Research Laboratory Project Engineer: Daniel A. Ventimiglia/IFED/(315) 330-3937

12a. DISTRIBUTION AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution unlimited.	

13. ABSTRACT *(Maximum 200 words)*

This report describes the development of a domain building speech interface to applications and databases. It describes how to build the domain model and the interface tools for a specific application with a SYBASE interface.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Natural Language, Speech Interface, Speech Input	32
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

1. Overview

Previous efforts have shown that speech recognition and natural language understanding can revolutionize the way people use complex software systems, such as those for accessing and displaying database information. Users can be more productive if they don't have to type (commonly called "fat fingering" in information) or know the complex syntax of SQL or some other database access language. Many layers of menus can be bypassed with a single expression in English, sometimes even getting at combinations of information that was impossible to access in the standard interface. Users can also learn how to use new systems more quickly if they can just say what they want, rather than having to learn the particular commands of the system.

The major drawback of spoken language interfaces, however, has been the amount of time and the expertise needed to build a language interface. The developer needed to be one with several expert knowledge skills. The developer had to be a linguist who understands the rules of English in order to write a grammar; a domain specialist, who needed to know both how the data is represented and the vernacular used to talk about it by the end user, and, an application specialist, who knew the underlying commands of the application and could link the language grammars to the application commands. Rarely could such an individual be found, so a ream of developers had to be assembled for every new domain and application.

The goal of the MELVIN project (Machine Exploitation of Language and Voice INtegration) has been to directly address this problem so that spoken language interfaces could more easily be created for new applications. The project addressed this problem in two ways:

1. Developing a core spoken language interface component that could be used across multiple domains. This core is called VISTA, Voice Interface System to

Applications, and it has been demonstrated on two different projects, MELVIN and another Rome funded project on collaborative interfaces.[1]

2. Building a ToolKit that allows developers who are not experts in linguistics or speech to build the necessary knowledge structures and link them to the database or other underlying application.

In order to demonstrate the capabilities of the spoken language interface itself, we built a demonstration system providing speech commands and data access via GTE's Temporal Analysis System (TAS). TAS is a workstation-based, analysis toolset designed to support military intelligence missions. Over the past eight years, GTE has developed sophisticated data visualization, data access, and expert systems techniques and software, and integrated them into a versatile toolset that is applicable across multiple intelligence domains, such as foreign command and control analysis, air threat tracking and analysis, and counter-drug operations. These tools automate the analysis of events over time in order to detect patterns of activity and predict future activity based upon either historical precedents or hypotheses.

The spoken language interface provides two kinds of capabilities: First, running the TAS tools, for example moving between the map and the timeline or zooming in on a location or a particular span of dates, and second, accessing information from the SYBASE database that holds the information being manipulated and displayed by TAS. A spoken language interface lets you easily integrate these two functions (e.g. "Show the aircraft takeoffs on a two day timeline starting March second"), and it lets you take advantage of context to abbreviate a command (e.g. "Show those events on the map", where "those events" are the last ones either referenced by words or selected by the mouse).

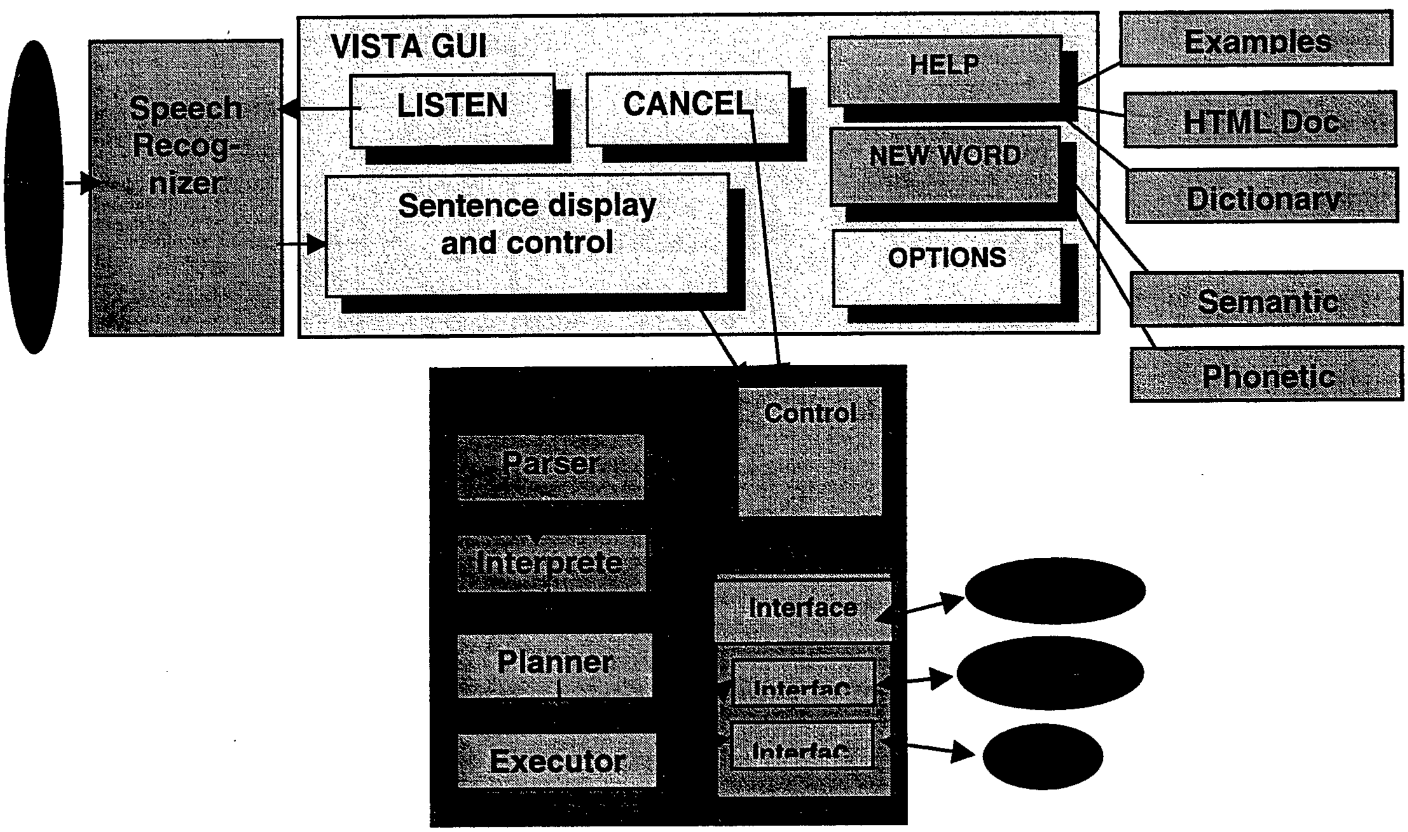
We first describe the VISTA spoken language interface in Section 2 and then describe the ToolKit in Section 3. Section 4 describes the integration of VISTA into the TAS application and the two scenarios that were developed to demonstrate the system. Section 5 outlines the milestones and major accomplishments of the project.

[1] Real Time Continuous Speech Recognition, AFRL contract F30602-94-C-0086

2. VISTA

The MELVIN speech interface system is called VISTA, Voice Input System to Applications, and as shown in Figure 1 shows the structure of VISTA. Voice input can enter the system from a microphone, or from some other source (e.g. a file of prerecorded speech). Typing and/or mouse gestures to the MELVIN user interface enhance capability without detracting from the utility of spoken input.

Figure 1 VISTA: the MELVIN Runtime System



There are three major parts to VISTA:

1. Speech Recognizer: Converts acoustic waveform into a word sequence. The recognizer is BBN's Hark™, which is a speaker independent, software only COTS speech recognition system.

2. VISTA GUI: Controls operation of speech recognition system and pass output of speech-recognition to language understanding system, allow for cancellation of system responses, editing of input and user preferences. Help KWIC (Key Word In Context) examples to show the user examples of how any word known by the system may be used, a conceptually organized vocabulary list to show the user a conceptual breakdown of words known by the system, and an HTML Online User's Manual. A "New Word" facility allows end-users to add new vocabulary.

3. Natural Language Understanding component (NLU): The Natural language understanding component has four major sub-components: (1) the parser, which determines the phrase structure of the user's request using a natural-language grammar designed for the application, (2) the semantic interpreter, which converts parse tree to meaning (including discourse effects such as the context of the topic being discussed using recent queries), (3) the planner, which converts meaning representation to a dataflow plan accessing appropriate databases and applications modules, and (4) the executor, which executes (and monitors) plan for data access and computation to meet user's needs.

2.1.1 Knowledge Sources

VISTA requires a number of knowledge sources for its processing. Figure 3 shows the relationship between VISTA (on the left), the knowledge sources (center) and the toolbox (right). The right hand box shows the tools for inputting the knowledge sources. Note that there are many more knowledge sources in the center which are used by VISTA than on input tools in the right hand side. The goal is for the user to only have to enter a small number of different knowledge sources, which are then automatically compiled into the correct format. For example, the "Lexicon Tool" lets the user add the word, put it into a grammatical class (e.g. noun, verb), and add the pronunciation all in one place. From this both the speech lexicon and NL lexicon are compiled automatically.

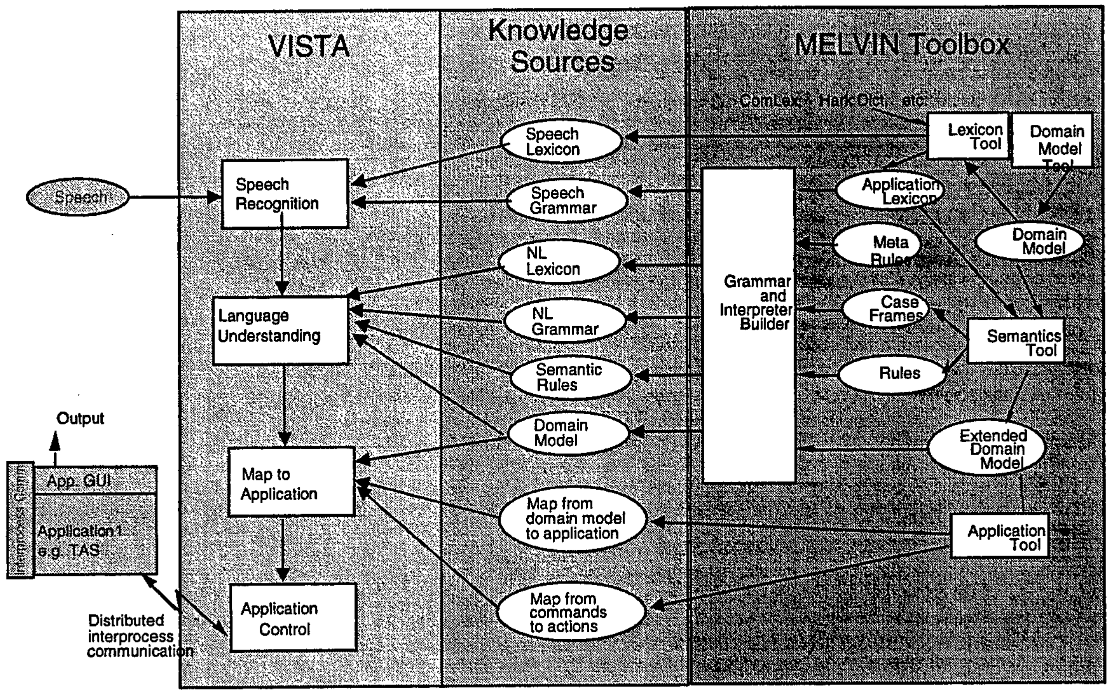


Figure 2: MELVIN Architecture

The following is a list of the knowledge sources used by each component of the system. Some of them, such as the speech lexicon and NL lexicon, contain very similar kinds of information, however the components that use them need the knowledge in different forms, so they are kept in separate knowledge bases. However, this difference is not apparent to the user, since one is generated automatically from the other. We address this issue in more detail in the tools section.

Speech Lexicon: a dictionary of the words that can be used in the system and their pronunciations.

Speech Grammar: a specification of phrases and sentences that will be understood by the speech recognizer. The speech grammar functions to control the search space in the recognition process.

NL Lexicon: a dictionary of the words in the application, syntactic information (e.g. singular, plural, noun, verb), and the association between the words and their meaning, i.e. their conceptual representation in the domain model.

NL Grammar: a set of rules specifying the legal phrases in the domain.

Semantic Rules: a set of rules specifying the meaning of phrases and their composition in the domain.

Domain model: a hierarchy of objects and attributes representing the things in the domain and their relationships.

Map from domain model to application: Specification of the relationships of domain model objects and objects in the application, for example the relationship of objects and attributes to the data base tables containing information about them.

Map from commands to actions: Specification of the relationship between commands defined in the language (e.g. "Zoom in") and the actions in the application that are to be carried out when that command is given. Allows a single command in language to execute a sequence of commands and database accesses in the application. For example, "Show the supply depots within 100 miles of Serajevo that have fuel valves for F-16's", requires first determining the supply depots in the appropriate locations, determining the parts available at each, and then displaying those locations.

3. MELVIN ToolKit

The MELVIN Toolkit is designed to help a developer to build the knowledge sources necessary for a speech language interface to an application using VISTA: Voice Interface System To Applications. A speech language interface has many parts. The speech sounds have to be recognized and transcribed as words, as in dictation systems. The words must then be understood to produce meanings in the context in which they are spoken. Finally, those meanings have to be translated into actions into the particular application, such as retrieving information from a database and displaying it on a screen.

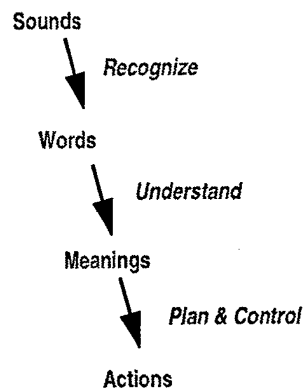


Figure 3: Steps in a Speech Language Interface

Each one of these steps requires knowledge in a particular form. The speech recognition system needs pronunciations for all of the words. Understanding requires grammatical structures and semantic categories to be associated with the words in the lexicon. The semantic categories, which represent the conceptual model expressed by the language, must be organized into a domain model representing the relationships between concepts. Finally, the concepts must be related to the application, for example associating SQL query language with the conceptual model.

3.1 User's view of the ToolKit and VISTA

While the MELVIN architecture captures the functional relationships between the knowledge sources from the components that use them in VISTA, the functional relationships are build by the MELVIN tools. The saved compiled output file of the Domain Model with its domain relationships to the source model and action applications, enables the runtime VISTA to perform recognized speech enabled actions, to be implemented though the VISTA GUI. This is shown in Figure 4. The MELVIN Toolkit is actually a completely separate program from the VISTAS (VISTA runtime application System) program as shown in the figure. This allows the user to incrementally build up knowledge sources independently of the run time VISTA system.

In the ToolKit perspective, the central knowledge source that all of the other information hangs off of is the domain model. Therefore, it is the central part of the ToolKit, as shown in the top portion of Figure 4. It allows the user to create and edit the domain model and other knowledge sources, such as the words using the "Lexical Tool" and the pronunciations, using the "Phonetic Tool". The lexical information is then connected to the domain model database source information using the "Source Model Extractor."

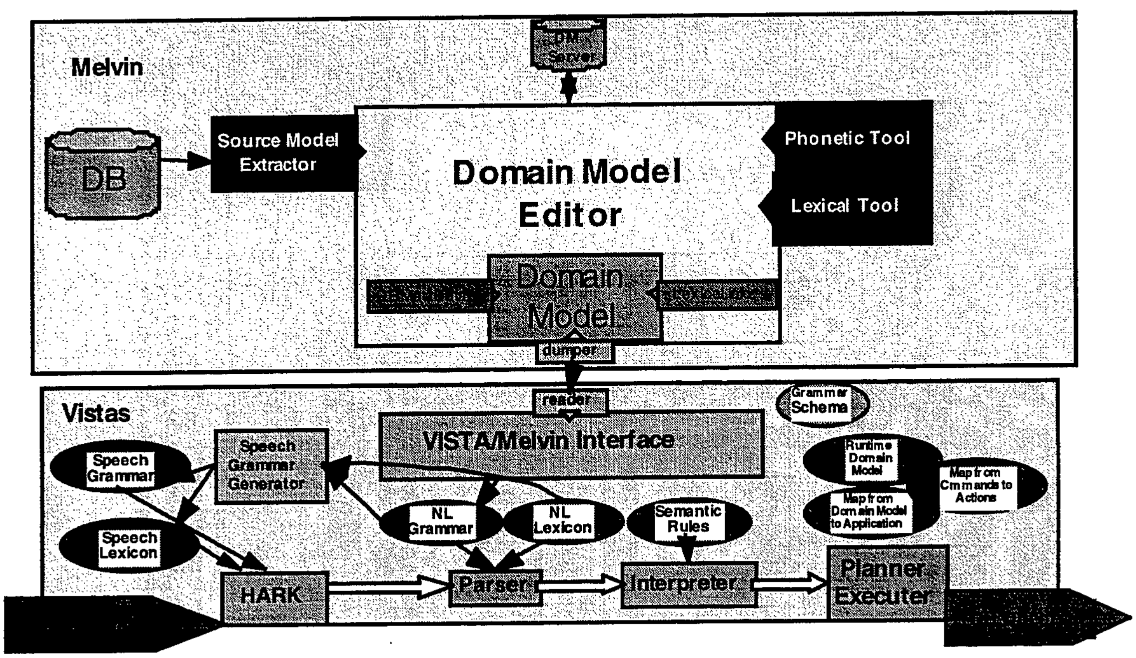


Figure 4 User's view of the ToolKit and VISTA

All of the knowledge that is created using the ToolKit is dumped into a file when completed (using the "dumper") so that it can be read into the VISTA system (VISTAS) (using the "reader"). This saved domain file is compiled into the correct form to be used for recognizing and understanding user's queries and commands.

Details of the architecture are described in the Software Design Document and its operation is described in full in the User's Manual. A full example of how to develop the speech application interface using the toolkit is shown in the User Manual.

3.2 Overview of steps to create a new application

The toolbox is designed to build a new application from scratch. The following are the major steps. These steps are described in detail in the MELVIN ToolKit User's Manual.

1. Develop target sentences for the domain

    - Compile a set of sentences that reflect the kinds of queries and commands the end user will need. This will both help direct the developer in what to add and provide a test suite to monitor progress. It will also be used to tune the speech grammar. It is important that this set be representative, but it does not need to be complete. Make sure that all of the major categories of things in the hierarchy are mentioned and that the most frequent variations of how questions are asked are included (e.g. at least one example of "Show all the..." and one of "Display the...").
    - It will speed up further work if the developer spend some time first understanding the database so that he knows what information is there and making sure the database is in the correct format, that is primary and foreign keys are correctly marked. It may turn out that information needs to be added, for example if user's tend to ask about troop locations by their commanding officers (e.g. "Show me where Lt. Smith's squadron is") then that information has to be included in the database.

2. Load source model from database
   - Load the database into the domain model editor. This is the "source model" and will include all of the information accessible to the end user from the database
3. Load in the core domain model.
   - This is the set of concepts that are common to all applications. All of the concepts that are added should be subconcepts of this set.
4. Build the domain model.
   - Extend the core domain model to the new domain by adding new subconcepts and attributes. This work should be guided by the target sentence list and the knowledge of what is in the database.
5. Add Lexical information
   - For each concept that will be explicitly mentioned, add the word and its pronunciation.
6. Connect the Source and Domain Models
   - For each concept, indicate what database concept it corresponds to.
7. Save out the knowledge in the VISTA format.
8. Load into VISTA, compile, and test.

As shown in Figure 5, this is not a strictly sequential operation. The first three steps can be done in parallel and the next three can be interleaved, which allows the develop to focus on one set of concepts and add in all of the knowledge needed for those (for example, all of the words, etc. needed to talk about aircraft events), and then move to another part of the domain.

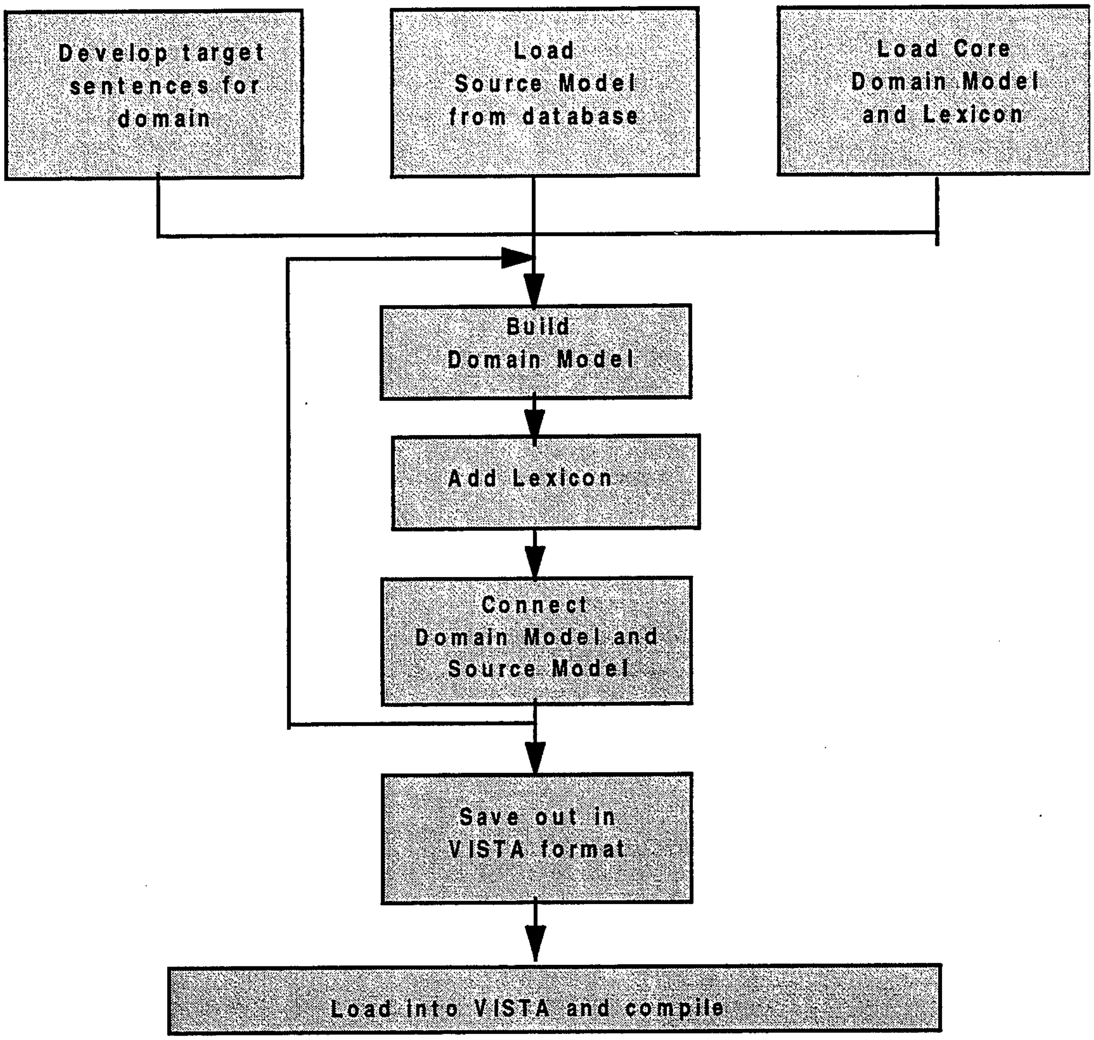


Figure 5 Toolkit Workflow

4. Integration of VISTA and TAS

The initially developed MELVIN application of VISTA was with TAS (Temporal Analysis System), an operational Air Force Research Laboratory product. It was selected as a good case study due to its significant user-base in diverse operational environments. The integration of VISTA and TAS was a significant portion of the project, since we used this initial integration to create the core spoken language system. Furthermore, TAS was used in a variety of domains, so we were able to work on portability of domains without initially addressing the issues of different applications. We implemented interfaces for two different databases. One focused on a fictitious strategic air scenario and the other on a scenario involving an uprising and secession of a state (in which the "Montana militia" secedes from "Atlantis"). We were also able to show the utility of a speech interface to actual user's of TAS at USSPACECOM, Colorado Springs, Co in June 97.

In this section, we first provide and overview of TAS and then describe the two domains, strategic air defense and regional conflict.

4.1 TAS

The Temporal Analysis System (TAS) is a workstation-based, analysis application designed to support military Situation Analysis intelligence missions. Over the past eight years, GTE has developed TAS providing sophisticated data visualization, data access, and expert systems techniques, and integrated them into a versatile toolset that is applicable across multiple intelligence domains. TAS is a versatile application. It emphasizes the study of events as a function of time to determine patterns of behavior. TAS tools aid the analyst in determining the potential situation(s) at hand by automating much of the analysis process. It digests incoming message traffic from various sources and types to detect patterns of activity and predict future activity based upon either historical precedents or hypotheses.

TAS is used operationally by multiple military commands and agencies within the DoD for a wide variety of applications including foreign command and control analysis, air sovereignty, criminal investigation, and counter-drug. TAS enables an analyst to visualize and analyze large volumes of data for the purpose of monitoring and correlating events, assessing situations and predicting future activities. TAS incorporates expert systems technology and provides a knowledge base that is user-maintainable in an environment, where the paradigms for activities are constantly changing. The Tools within TAS to facilitate the analysis of large volumes of data are timelines, maps, query tools, keyword dictionary and the Knowledge-based Prediction Analysis Situation Assessment (K-PASA) expert system.

In order for the MELVIN VISTA system to drive TAS, BBN/GTE developed a macro language interface within TAS. It is described in detail in the Software Design Document. This interface allows speech commands to be recognized into the TAS tools environment. The interface tools include: setting timeline dateframes or asking questions about historical activities. This language specification enabled the development team to translate spoken requests into statements that can be passed to the TAS application via an interface such as a socket or command line argument.

4.2 Scenario 1: Strategic Air Defense

The initial scenario was chosen for the following reasons: 1) It is realistic in the sense that it parallels probable real-world foreign activity, 2) It uses real data taken from "official" sources, 3) It is unclassified, and 4) It contains "known" data and, by so doing, removes the domain information variable from the equation and allows the engineers to focus on implementing and validating new technology in the intelligence arena.

The scenario data indicates that a high-level of communications by various headquarters elements and flight activity by strategic aircraft occurred during the period from late June through early July, 1996. The analyst has been tasked to determine whether the communications and aircraft activity was random, or related to an actual strategic alert, or an exercise. The keys to this determination are the types of communications detected (codewords, other), the temporal relationships between the various communications and the observed flight activity, the units involved (i.e. those who control the strategic aircraft and missiles), the type of flight activity observed (practice bombing, etc.), and communications events following the flight activity which might indicate the termination of an exercise.

4.3 Scenario 2: Regional conflict

After the initial scenario was in place, it was decided to create a new scenario, both to test the portability of the system and to provide a demonstration that was more relevant in the post-cold war period. GTE developed a scenario based on a Five-day, non-nuclear regional conflict (e.g., Chechnya) primarily involving aircraft assets. The scenario describes a revolt by state of Montana (Red Force) against Atlantis Federation (Blue Force). It contains several phases, including build-up, deployment, and simulated air combat. Blue Forces are represented by U.S. Air Force units and aircraft; the Red Forces are represented by fictitious Russian Air Force units and aircraft.

The following are some sample queries and command for the Regional Conflict Scenario

1. Show the map.
2. Make the start date March first.
3. Set the duration to five days.
4. Show geopolitical events.
5. Show a three day timeline starting on March first.
6. Show Montana geopolitical events on March first.
7. Add Montana terrorism events.
8. Show Montana aircraft deployments.
9. Show those events on the map.
10. Raise the map.
11. Go to Montana.
12. Set resolution to high.

13. Raise the timeline.
14. Show JCS codewords.
15. Show codeword events from CENTCOM to AMC and ACC.
16. Show all comms to three sixty sixth wing on the timeline.
17. Show Atlantis bomber takeoffs.
18. Show those events on the map.
19. Zoom out.
20. Show Elsworth Air Force base.

5. Summary of Milestones and Accomplishments

In this section we list the milestones from the original proposal, all of which were completed in the course of the project, and then look at the accomplishments of the project throughout the two years.

5.1 Milestones

5.1.1 Milestones for Year 1

- Evaluate a selection of noise reduction microphones and make recommendations on which are most useful in an office environment.
- Perform an analysis of existing speech recognition and natural language understanding systems and tools, to determine which are most appropriate for use in the MELVIN system.
- Design and document initial MELVIN runtime architecture and module interfaces to allow a voice interface for both natural language query and commands.
- Design and document initial MELVIN ToolKit architecture and module interfaces to allow a non-linguistic expert to build, configure or modify the voice interface.
- Work with GTE to design speech interface to TAS that is compliant with the initial architecture
- Work with GTE to perform a task analysis with potential end users (current TAS users)
- GTE will extend TAS as necessary to comply with the initial architecture
- Develop the first version of MELVIN tools
- Develop the first version of MELVIN runtime environment

- An initial GUI and demonstration version of MELVIN for TAS, using tools where available. Demonstrate both open microphone and "click to talk" modes of interaction. Leave system behind at Air Force Research Laboratory at Rome, NY.

5.1.2 Milestones for Year 2

- Complete initial design and implementation of all MELVIN tools, using feedback from government, military users, and GTE.
- Document MELVIN ToolKit for system administrators
- Have GTE or military personnel test the ToolKit to create, use, configure, and modify a voice interface to TAS, or to another database system or intelligence data handling system.
- Demonstrate the system interfacing with audio input from a source other than a microphone
- Use the tools, and demonstrate the resulting runtime system interfacing directly with SYBASE {SOW 4.1.4}
- Revise MELVIN tools, using feedback from government, military users, and GTE.
- Revise MELVIN ToolKit documentation for system administrators
- Conduct an evaluation to examine the usability and utility of MELVIN, involving an appropriate user community
- Document MELVIN runtime system for system administrators
- Demonstrate the full functionality of MELVIN, using a demonstration scenario {SOW 4.1.5, 4.1.5.1}
- Preliminary test/demonstration {SOW 4.1.5.1}

- Train government personnel {SOW 4.1.6}
- Deliver MELVIN prototype

5.2 Accomplishments

5.2.1 Kickoff : June 1, 1996

5.2.2 By December 30, 1996

- VISTA and TAS integrated
  - Implement socket interface to SYBASE
  - Implement alpha version of socket interface to TAS
- Initial Demonstration Scenario
  - Began testing of queries to SYBASE from demonstration scenario
  - Began testing of TAS display parameter commands from demonstration scenario
- Draft Software Design Document
  - Circulated sections on VISTA and Interface for review
- Preliminary Toolbox Design
  - Draft architecture for toolbox
- Microphone testing

5.2.3 By June 30, 1997

- Completed new scenario
  - GTE provided scenario, database, and sample queries

  - BBN provided all language and speech knowledge sources
- Added on-line user help and documentation
- Used web browser to make background info on scenario available to users
- Completed testing of VISTA for hands off and hands on demonstration
  - Improved both language coverage and speech recognition performance
- Showed VISTA/TAS to USSPACECOM TAS users and managers at Colorado Springs
  - Three sessions of demonstrations/user trials
  - Generated lots of enthusiasm about speech interfaces
- MELVIN Toolbox
  - First cut domain model and source model tools
  - Began design of lexicon/semantic tool

5.2.4 By December 31, 1997

- Took VISTA / TAS demonstration to DODIIS (all trade show expenses paid by BBN marketing, including labor and travel)
- Delivered VISTA system to AFRL
- Implementation of Draft Tookkit
  - Implemented Domain Model Editor
    - Work done in conjunction with BBN's LogWeb project
  - Implemented Source Model Extraction Tool
    - Uses JDBC to automatically pull information from TAS (SYBASE) database

- Implemented tools to add lexical information
  - Tools use examples rather than asking users for syntactic information
  - Pronunciation tool automatically connects with HARK 100,000 word dictionary
- Implemented connection between ToolKit and VISTA

- Wrote draft of MELVIN ToolKit User's Manual

5.2.5 By September 30, 1998

- Completed full implementation of Toolkit
- Completed User's Manual
- Delivered system to AFRL
- Trained AFRL personnel in how to use Toolkit

6. Conclusion

The MELVIN project tackled a extremely challenging problem that is well beyond the current state of the art: How to build a speech interface with natural language understanding that can be ported to new applications and domains. Within a two year time frame, we accomplished the following:

- Proof of principle system with one application (TAS) and two domains
- Use of VISTA on one other Rome Funded project (Real Time CSR: Voice Collaboration)
- Development of a Java-based portable ToolKit for building new applications.
- Application of the toolkit to the second domain using TAS.

While we have "proved the principle", there is still more work to be done, both in testing and tuning the existing system and extending the capabilities of the basic technology. At the very least, we need to do the following:

- Identify applications that are speech ready
- Train AFRL personnel in using ToolKit
- Work with them to refine toolkit and bring up new applications
- Work on extending spoken query/command to true mixed initiative dialog both at the workstation and over the telephone

Our ultimate goal is natural interaction with complex applications in order to make those applications quicker and easier to use, bringing more timely information to the people who make critical decisions in the DoD.

MISSION OF AFRL/INFORMATION DIRECTORATE (IF)

The advancement and application of information systems science and technology for aerospace command and control and its transition to air, space, and ground systems to meet customer needs in the areas of Global Awareness, Dynamic Planning and Execution, and Global Information Exchange is the focus of this AFRL organization. The directorate's areas of investigation include a broad spectrum of information and fusion, communication, collaborative environment and modeling and simulation, defensive information warfare, and intelligent information systems technologies.