****

**Software Design   
 Specification (SDS)**

BabbleBot

Version 1.0

Prepared by:

Group 6  
Mad DJs

|  |  |  |
| --- | --- | --- |
| * Morgan McKeithan |  | Mckeithanm15@students.ecu.edu |
| * Ashley Fraley |  | Fraleya14@students.ecu.edu |
| * Jeremiah Smith |  | Smithjere14@students.ecu.edu |
| * Eric Sengmany |  | Sengmanye14@students.ecu.edu |
| * Daehan Barnes |  | Barnesda14@students.ecu.edu |

|  |  |
| --- | --- |
| Instructor: | Dr. Kamran Sartipi |
| Course: | CSCI 3030 (Software Engineering I) |
| Date: | 10/30/2017 |

Contents

[Revisions ii](#_Toc117484243)

[1 Introduction 1](#_Toc117484244)

[1.1 Purpose 1](#_Toc117484245)

[1.2 System Overview 1](#_Toc117484246)

[1.3 Definitions, Acronyms and Abbreviations 1](#_Toc117484247)

[1.4 Supporting Materials 1](#_Toc117484248)

[1.5 Document Overview 1](#_Toc117484249)

[2 Architecture 2](#_Toc117484250)

[2.1 Overview 2](#_Toc117484251)

[2.2 Component 1..n 2](#_Toc117484252)

[3 High-Level Design 3](#_Toc117484253)

[4 Low-Level Design 4](#_Toc117484254)

[4.1 Modules Overview 4](#_Toc117484257)

[4.2 Module Specifications 4](#_Toc117484258)

[4.2.1 Module X1 4](#_Toc117484259)

[4.2.2 Module X2 5](#_Toc117484260)

[Appendix A – Group Log 6](#_Toc117484261)

Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
| --- | --- | --- | --- |
| 1.0 | Morgan McKeithan, Eric Sengmany, Jeremiah Smith, Ashley Fraley, Daehan Barnes | Complete first version of the BabbleBot Software Design Specification document | 10/30/17 |

# Introduction

## Purpose

This document provides a complete example of the first version of a Software Design Specification document for the machine learning chat bot, BabbleBot. This document is primarily based on the SRS document that was earlier prepared. In the remainder of section 1, we specify the purpose of this document, the overview of BabbleBot, and the sources used in the production of this document.

## System Overview

BabbleBot Software System runs is accessed through a website. The minimal requirements related to the user accessing the website are listed in section 2.4 (Operating Environment) of the BabbleBot SRS v. 1.0 document. The system will make use of the BabbleBot database by sending user input to the program and accessing the database to formulate a response to then send back to the website to show the user. The database will interact on the Internet with Scrub dictionary in order to gain the part of speech of new words to be entered into the database. The main purpose of BabbleBot is to educate users on sentence structure.

## Definitions, Acronyms and Abbreviations

Token – a word

Tokenize – break down a sentence into separate words

## Supporting Materials

**The following standards apply**:

J-STD-016-1995 IEEE/EIA Standard for Information Technology, Software Lifecycle Processes, Software Development, Acquirer-Supplier Agreement

IEEE-STD-P1063 IEEE Standard for Software User Documentation

**The following texts and documents have been used in the process of developing this document**:

[1] SDS for MiniThermostat template

## Document Overview

The next section of the BabbleBot SDS v. 1.0 provides the architectural view of the system. It shows the BabbleBot system broken down into subsystems and the reasons for each subsystem. Subsections of section 2 describe the subsystems and their corresponding interfaces. Section 3 provides a control view of BabbleBot and describes the details of each state of the system. Section 4 provides a low-level design of the system.

# Architecture

This section provides the architecture design of the BabbleBot software system. It includes the final version of the system component diagram which shows the different subsystems, their interfaces, and their dependencies on related subsystems. Section 2.2 discusses the different components of BabbleBot. The next section provides the Component Diagram.

## Overview../Desktop/Screen%20Shot%202017-10-30%20at%2011.24.52%20AM.png

As illustrated in the component diagram above, BabbleBot architecture consists of three main components:

1. User Interface/Website
2. BabbleBot Program
3. Database

In order to make the architecture more understandable, maintainable, and adoptable for changes it was decomposed according to the different functional areas that the system covers. The BabbleBot SRS covers the functionality in more detail of each component.

## User Interface

Babblebot will be showcased on an aesthetically pleasing platform to capture user attention and allow for easier conversing.

* Input box - the input box allows for a user to send data to the BabbleBot program
* Conversation History box (Output) – this interface comes from the BabbleBot program – it will be the history of responses that are created by BabbleBot
* Sentence structure diagram area (Output) – this interface comes from the BabbleBot program – a tree will be displayed showing the parts of speech of each word that the user inputs

## BabbleBot Program

Whether it is a question or statement, BabbleBot will read in user input, analyze it, and send out an appropriate response as well as a sentence structure diagram

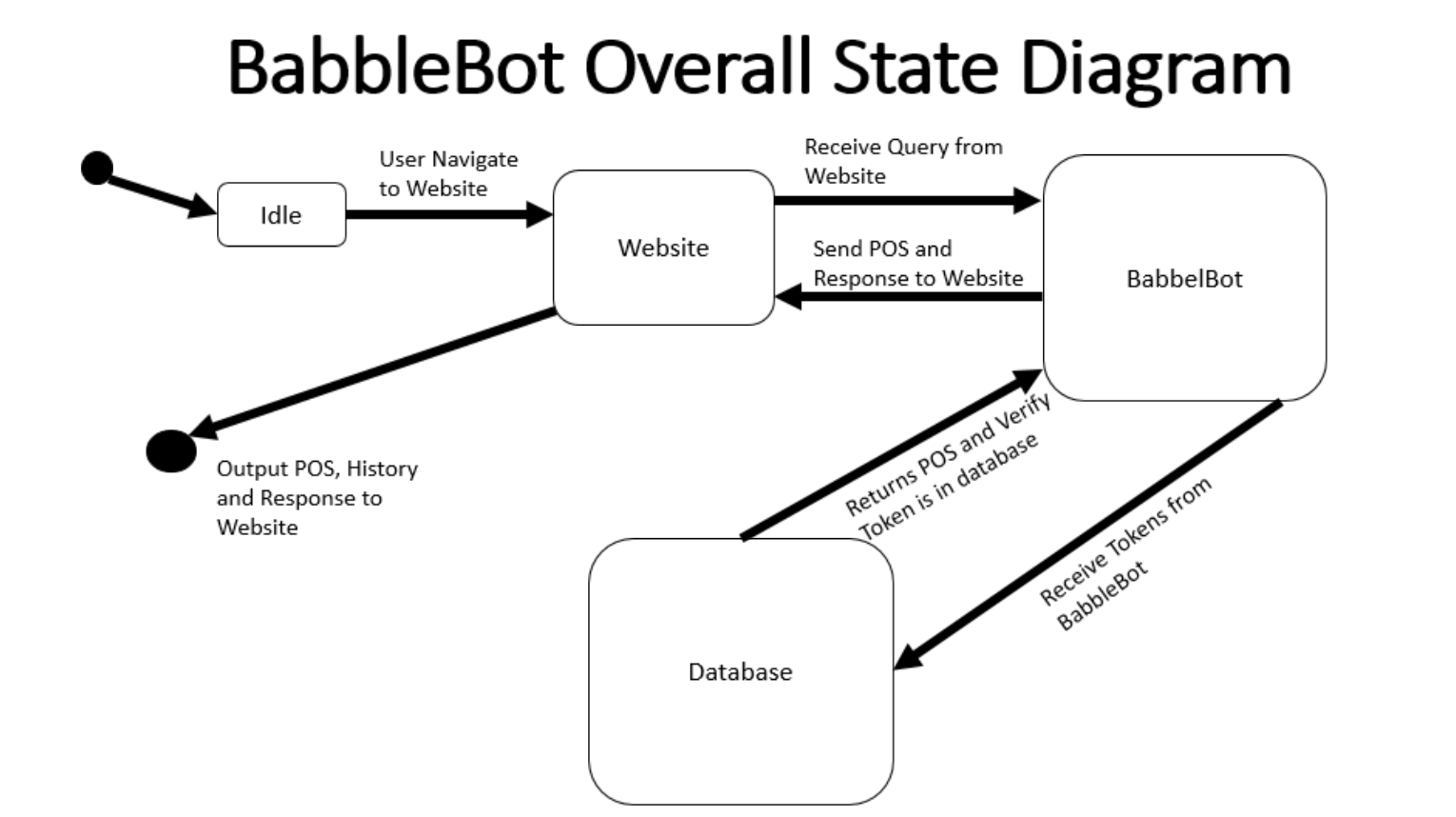
* Retreieve input from website
* Compare input with BabbleBot’s existing database
* Send output back to website

## BabbleBot Database

The BabbleBot database consists of stored words and their parts of speech. The stored words are words that have been previously inputted to BabbleBot. After each new input, BabbleBot will add to the database in order to improve itself.

# High-Level Design

The High-Level Design section describes in further detail the interactions between the system components and their corresponding interfaces. To illustrate the dynamic behavior of the BabbleBot system, a statechart showing how the components interact with each other is shown.



In short, the website used to access the BabbleBot program becomes responsive when the user navigates to it. The website then sends the input data of the user to the BabbleBot program where the program breaks the input up into words and sends them to the database. The database sends back data to the BabbleBot program. The program then creates its responses and sends them back to the website.

# Low-Level Design

The following sections describe the low-level design of the BabbleBot software system. It breaks down previously discussed components into modules. The next section provides an overview of the modules and then the following sections go more into detail on the interface and design of each module.

## Modules Overview

BabbleBot system was decomposed to 3 independent modules. A brief description of each module is provided below:

**Name: Website Interface**

**File Name:** BabbleBot.html

**Naming Convention:** All funtions in this module will have the prefix Wb\_

**Short Description:** This module is a part of the website component. It is where the user will interact with BabbleBot using provided text boxes.

**Container Component:** This module serves as the User Interface component. It is within this model that the user will interact with the BabbleBot program and see information about the project.

**Name: Babble Bot Interface**

**File Name:** BabbleBot.py

**Naming Convention:** All functions in this module will have the prefix Bb\_

**Short Description:** This module takes user input and uses the logic programmed into the BabbleBot program to perfrom actions based on the input. Its main tasks are to store conversation history, parts of speech, and formulate responses.

**Container Component:** This module is isolated to ensure that the users to have access to the code of the program. We restrict user interaction through the use of the website module, so users are only able to type in queries. It is important to have this module separate from the rest, as it requires a large amount of code.

**Name: Database Module**

**File Name:** BabbleBot.py

**Naming Convention:** All functions in this module will have the prefix Db\_

**Short Description:** The database module is used in tandem with the BabbleBot interface. It is comprised of many different corpora of words and definitions. It is used to retrieve definitions, part of speech, and sentence structure. If a word is not found within the database, it will search the web for the definition and add it to the database.

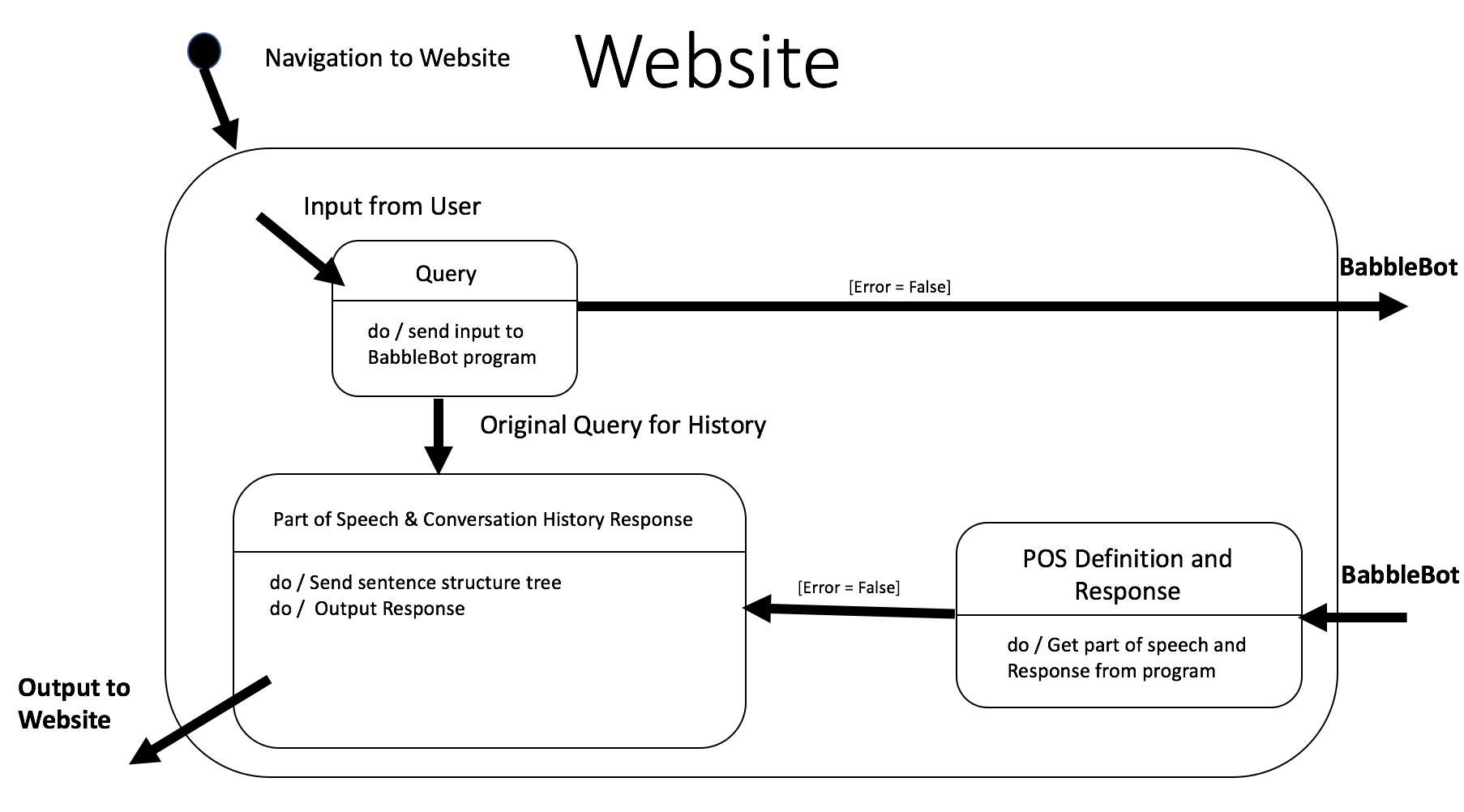
**Container Component:** This module is a part of the BabbleBot interface. BabbleBot uses this module to formulate the response it wants to return to the user.

## Module Specifications

The following sections provide a more detailed overview of module interface specifications and its design. To specify the interfaces we list the module internal state variables and then provide the semantics for its access functions. In the design section we provide a list of internal components and describe the design of each component using statechart diagrams.

### User interface/Website Module

The website module provides the system with the most important functionality to the user – it allows users to access BabbleBot with ease. The website uses the traditional HTML, CSS3, and Javascript programming languages. User input will go directly to the BabbleBot program and output will come directly from the BabbleBot program.

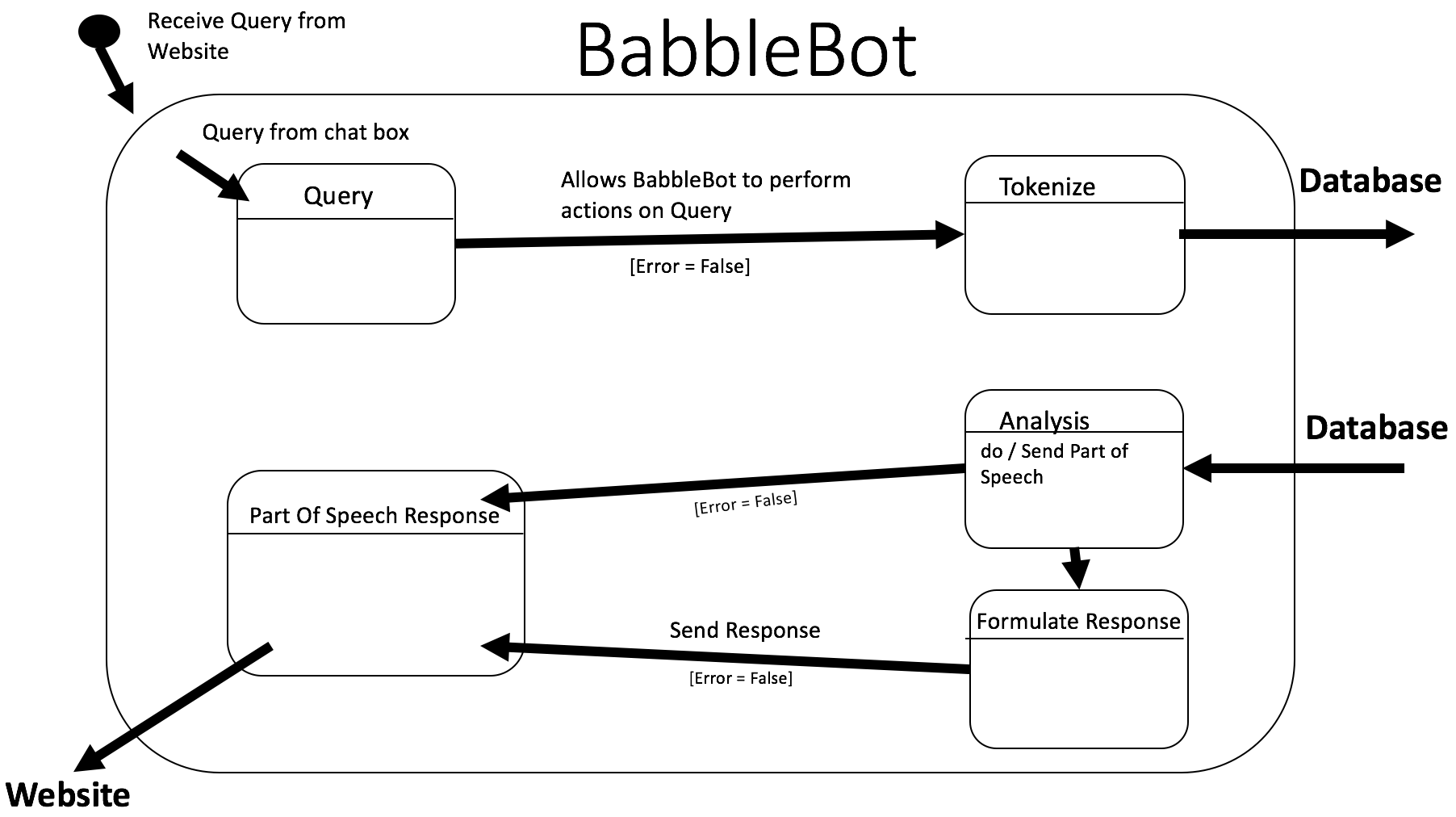
**

### BabbleBot Program Module

The BabbleBot program is the most important module of the system. It directs data traffic where it should go. It also is able to put “part-of-speech” tags on words as well as create a response for BabbleBot to send to the user of the website.

|  |  |  |
| --- | --- | --- |
| **Used External Modules:** | NLTK (Natural Language Tool Kit) | |
| **Used External Data Type:** | token | |
| **Internal State Variables:** | sentence | |
| **Exported Function(s)** | | **Description** |
| word\_tokenize | | word\_tokenize separates the words of a given input (sentence). |

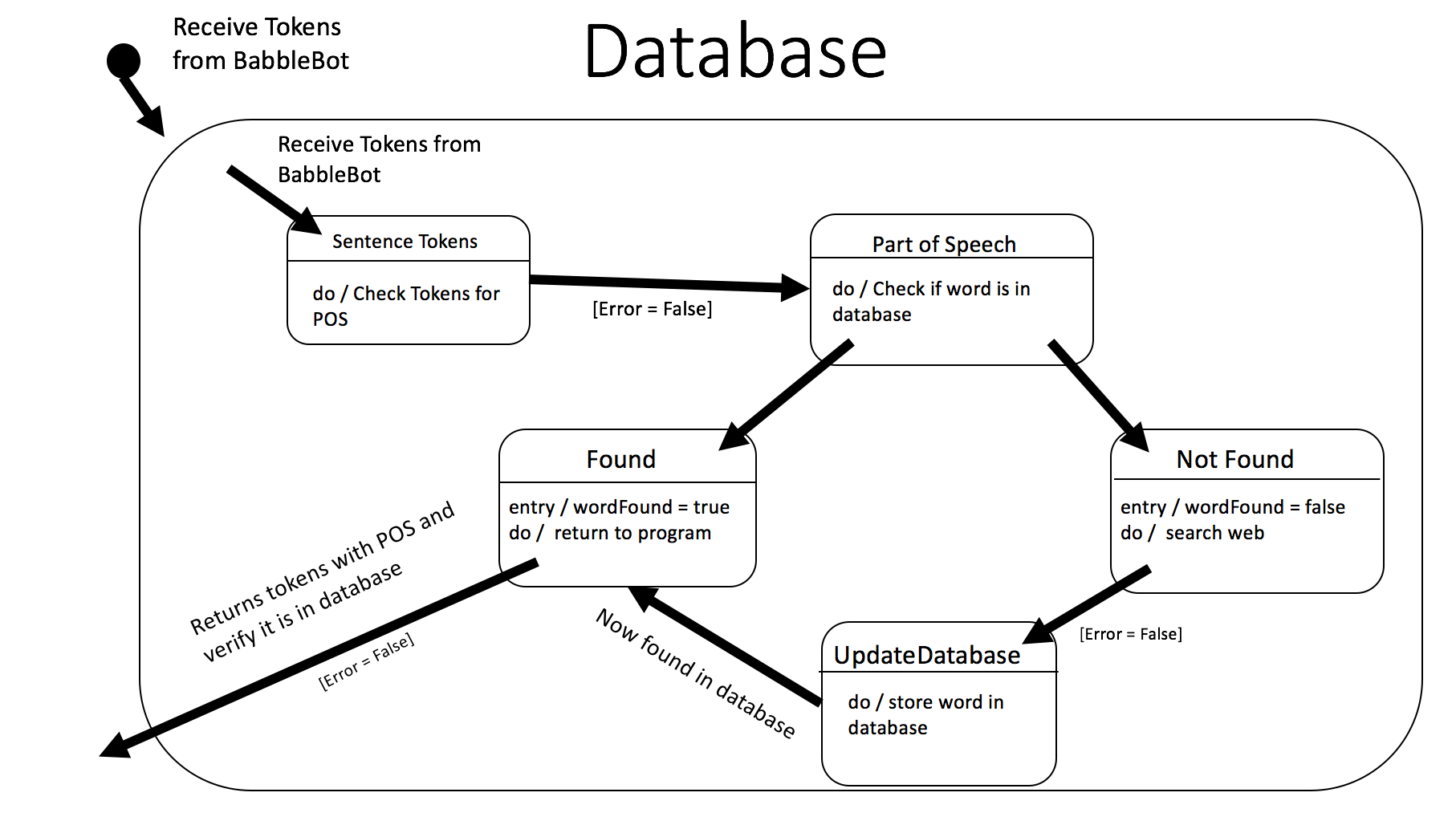
|  |  |
| --- | --- |
| **Internal Function(s)** | **Description** |
| getInput  sendOutput | getInput will get the input from the user from the website.  sendOutput will send BabbleBot’s response to the website. |



### BabbleBot Database Module

The database is a collection of words along with their part of speech. The database works very closely with the BabbleBot program so the program will actually use functions to update the database.

|  |  |
| --- | --- |
| **Internal State Variables:**  ***Internal Functions:***  getDefinition()  updateDatabase() | contains(): Boolean  pos(): string  **Description**  Searches the web for new defintion – input word to search  Update database adds a new word and its part of speech to the BabbleBot database. Inputs would be the new word and the part of speech to add. |



Appendix A – Group Log

GroupMe discussion – October 18, 2017: Split SDS into sections

GroupMe discussion – October 21, 2017: Discuss any problems/updates

Group meeting – October 28, 2017: work on further programming of BabbleBot and figure out how to appropriately fill out section 4 of the SDS

Group meeting – October 30, 2017: finalize StateCharts