Table of Contents

[TL:DR 1](#_Toc68953695)

[Blog 1 1](#_Toc68953696)

[(Giving a Voice to Rasa with Botium Speech) 1](#_Toc68953697)

[Blog 2 3](#_Toc68953698)

[(How to build a voice assistant with open source Rasa and Mozilla tools) 3](#_Toc68953699)

[Blog 3 5](#_Toc68953700)

[(webkitSpeechRecognition and Rasa) 5](#_Toc68953701)

## TL:DR

Lugedes läbi saadetud blogid, jäi mulle kõige enam silma teine blogi, kus kasutati Mozilla toolsí ([Mozilla DeepSpeech](https://github.com/mozilla/DeepSpeech)/ [Mozilla TTS](https://github.com/mozilla/TTS)). Mulle tundub et POCí raames võiks antud tehnoloogiaga edasi minna. Kuna ta on open source, Rasa enda tiimi poolt testitud ja on kergelt muudetav vastavalt meie vajadustele.

Täpsemad blogide kokkuvõtted on iga blogi sissekande lõpus.

## Blog 1

## ([Giving a Voice to Rasa with Botium Speech](https://floriantreml.medium.com/giving-a-voice-to-rasa-with-botium-speech-1c65a7513bdf))

Botium Speech Processing is a unified, developer-friendly API to the best available free and Open-Source Speech-To-Text and Text-To-Speech services.

Tegemist on võrdlemisi otsekohese lahendusega, tundub et S2T ja T2S lahendused on iseseisvad, ehk vajadusel saame antud juhendist soovi korral ainult T2S lahendust kasutada.

Näites kasutatud kriitilised nõuded:

* Git client
* Docker and Docker-Compose

Architecture

1. User speaks into a microphone
2. A Speech-To-Text service translates into text **(Botium Speech Processing)**
3. An NLU engine extracts information out of the text **(Rasa)**
4. A dialogue engine builds text response **(Rasa)**
5. A Text-To-Speech service translates into spoken text **(Botium Speech Processing)**
6. User listens to the audio file

Diagram

Description automatically generated

Botium Speech Processing comes with a reasonable default configuration.

* [MaryTTS](http://mary.dfki.de/) for Text-To-Speech (Written in Java)
* [Kaldi](https://kaldi-asr.org/) for Speech-To-Text (Shell, C++, natuke pythonit)

Lisa [moodul](https://github.com/codeforequity-at/botium-speech-processing/tree/master/connectors/rasa) mis lisab RASAĺe S2T ja T2S võimekused, kasutades Socket.io liidest.

Botium koosneb kolmest põhiosast: Kaldi (S2T), MaryTTS (T2S) ja Sox for audio file proccessing.

[Defaultina](https://github.com/codeforequity-at/botium-speech-processing) on 2 keelt olemas: Inglise ja Saksa.

Täiendavaid keele mudeleid saab allalaadida antud [link](https://kaldi-asr.org/models.html)śt.

Kaldi mudeli treenimise [tutorial](https://kaldi-asr.org/doc/kaldi_for_dummies.html).

Lahenduse ülevaade:

Antud lahendus tundub olevat üsna mõistlik aga ei oleks minu esimene valik, kuna

1. On open source, mis on postiviine
2. Ma ei oska öelda kui palju saame muuta/ luua oma komponente + ma ei ole kindel kui suur on hetkene kasutajate arv (online toe olemasolu on tähtis).
3. Tundub kirjeldusest et Botium speech processing on pigem läinud lihtsamat teed pidi, et vajab vähem settuppi ja et saaks võimalikult kiiresti töötava asja püsti aga see-eest on vähem mugavndamis võimalusi. Mis on antud projekti puhul tähtis omadus.

## Blog 2

## ([How to build a voice assistant with open source Rasa and Mozilla tools](https://blog.rasa.com/how-to-build-a-voice-assistant-with-open-source-rasa-and-mozilla-tools/))

Antud lahendus kasutab ainult open source tööriistu.

After exploring the currently available options: CMUSphinx, Mozilla DeepSpeech, Mozilla TTS, Kaldi, we decided to go with Mozilla tools - [Mozilla DeepSpeech](https://github.com/mozilla/DeepSpeech) and [Mozilla TTS](https://github.com/mozilla/TTS). Here is why:

* Mozilla tools come with a set or pre-trained models, but you can also train your own using custom data. This allows you to implement things quickly, but also gives you all the freedom to build custom components.
* In comparison to alternatives, Mozilla tools seem to be the most OS agnostic.
* Both tools are written in Python which makes it slightly easier to integrate with Rasa.
* It has a big and active open source community ready to help out with technical questions.

Mozilla DeepSpeech is a speech-to-text framework which takes user input in an audio format and uses machine learning to convert it into a text format which later can be processed by NLU and dialogue system. Mozilla TTS takes care of the opposite - it takes the input (in our case - the response of the assistant produced by a dialogue system) in a text format and uses machine learning to create an audio representation of it.

* Diagram

  Description automatically generated

Antud lahendus kasutab sammuti RASA SocletIO lahendust.

there is a lot of room for improvement, especially at the STT and NLU stage. How can you improve it? Here are some ideas for you:

* Pre-trained STT models are trained on a quite generic data which makes the model prone to mistakes when used on more specific domains. Building a custom STT model with Mozilla DeepSpeech could lead to a better performance of STT and NLU.
* Improving the NLU could potentially compensate for some of the mistakes made by STT. A rather simple way to improve the performance of the NLU model is to enhance the training data with more examples for each intent and to add a spellchecker to the Rasa NLU pipeline to correct some smaller STT mistakes.

Lahenduse ülevaade:

Antud lahendust peaks tõsisemalt kaaluma, kuna

1. Ta kasutab ainult open source tööriistu
2. Kuna on Rasa enda poolt loodud, siis on suurem tugi/ integreeritus Rasa´ga.
3. On suurem hulk kasutajaid ehk online tugi on suurem
4. Saame luua enda custom componente.
5. Me vist peame oma eesti keele mudeli ise treenima aga see peaks olema tehtav ([link](https://medium.com/analytics-vidhya/deep-speech-train-native-languages-with-transfer-learning-part-0b01-13d24f1337c7))

## Blog 3

## ([webkitSpeechRecognition and Rasa](https://blog.rasa.com/webkitspeechrecognition-and-rasa/))

Antud tehnoloogia põhikomponent on [Web Speech API](https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API), mis töötab ainult Google Chrome´i ja Edge´ga. API koosneb nii T2S (Speech synthesis) ja S2T (Speech recognition) komponentidest. Antud API´t kasutades jagatakse andmeid kolmanda osapoolega, mis võib olla osade klientide puhul *deal breaker*.

Hetkel on see API eksperimentaalne ja kui interneti ühendus kaob ära siis bot enam ei tööta.

A picture containing text

Description automatically generated

Antud lahenduses luuakse keskserver nimega API. Mis võtab ühendab Rasa´t ning frontEnd javascripti, kus tuleb tulevad teksti failid. Keskmise proxi implementeerimiseks kasutavad [FastAPI](https://fastapi.tiangolo.com/).

Web Speech API [ei toeta eesti keelt](https://stackoverflow.com/questions/14257598/what-are-language-codes-in-chromes-implementation-of-the-html5-speech-recogniti) ning mulle tundub et teda ei ole võimalik ka treenida.

Lahenduse ülevaade:

Kuigi antud lahendust on võrdlemisi lihtne implementeerida siis ma pigem ei kasutaks seda kuna tal on üsna palu piiranguid mis ei kattu antud projekti nõuetega.

1. Ei ole open source
2. Jagab osaliselt andmeid kolmanda osapoolega (Kindlasti ei sobi PPA/ kaitseressurside ameti/ Kaitseväe seda sorti projektide puhul)
3. Toetab ainult Chromeí ja Edgeí.
4. Ei toeta eesti keelt.