WowWee Inc.

SwitchBot

powered by goDog

Revision 0.2 - 2014-MM-DD

Davin Sufer

Document revisions history

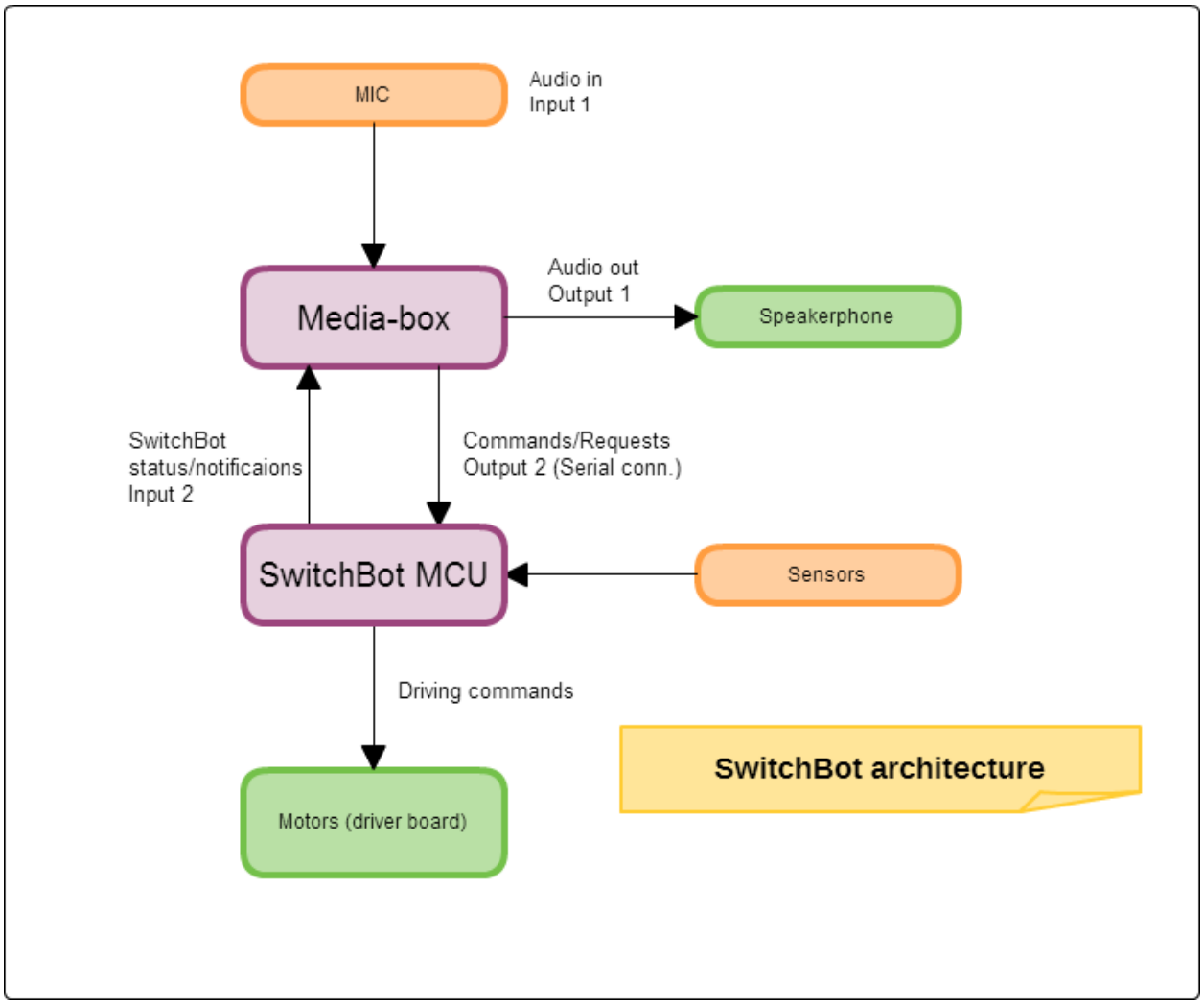
Revision number	Date	Comments
0.1	2014-11-14	First draft
0.2	**waiting approval	Additions requested by Davin and Nick

Document summary

1-Introduction

By integrating goDog to SwitchBot, our goal is to bring some **AI** to the motion capabilities of SwitchBot. To achieve this goal we need to take in consideration that in SwitchBot there's no display. We'll need then to focus on the audio output capabilities of goDog. This implies that we'll not use, for example, Web searches like "*pictures of elephants*" or *"photos of Paris"*.

The hardware setup is shown by the following diagram :



2-Requirements

2.1-SwitchBot speech capabilities

2.1.1-Generic questions

In this part we want basically to be able to talk to SwitchBot, we can for example ask questions like :

- How are you ?
- What time is it ?
- Who is Bill gates ?
- Who was George Washington ?
- What's 20% of 80 ?

2.1.2-Locations related requests

We can also ask for locations :

- Find pizza
- I want Italian food

2.1.3-Translations related requests

And finally we can ask for translations :

- Translate good afternoon to French
- Translate how are you to Italian

The output on all these cases will be pure audio, we need to take off anything related to visual display. We have to add also audio output for errors, non supported or not recognized commands and maybe some audio output while the requests are processed.

goDog already include all these features, we'll need maybe just to customize it for a pure audio output.

2.2-SwitchBot motion capabilities

In this part we need to control SwitchBot motion through voice commands. The format of a voice command will be a concatenation of two parts

1. the command
2. command params (optional)

As a starting point we can have the following commands :

- ❑ **move forward number_of_steps**
- ❑ **move backward number_of_steps**
- ❑ **turn left number_of_steps**
- ❑ **turn right number_of_steps**
- ❑ **dance**
- ❑ **stand up**
- ❑ **track**
- ❑ **lean forward**
- ❑ **lean backward**

We can think about the possibility to provide a sequence of commands on one shot, like :

"forward 5, left 2, forward 5, right 3"

goDog should provide for each voice command an output through the serial port to the switchbot microcontroller. The voice commands needs to be parsed to byte packets as follow :

<START_BYTE> <command_code> <command_param> <END_BYTE>.

2.3-SwitchBot behaviour

In this part we need to define some handlers for the events coming from SwitchBot microcontroller. We want, in this part, to mimic some natural behaviour, for example :

- If he falls over, he will say " oh ow !!"
- If the battery level is too low, he well say " battery low, ...!!" *
- …

(we need to add maybe more features here …)**

* This feature is not implemented in the version 1.0.

2.4-SwitchBot status

The android module should be able to get some status information regarding the robot. This information is used to extend the robot capabilities or for debugging purposes. This information can be related to the robot posture :

- kneeling
- standing up
- laying down
- transitioning

For the transitioning state we need to get the start and end positions.

The information can also be related to the robot joints. In this case the application should get the position of each joint, namely :

- torso
- left & right thigh positions
- left & right calve positions
- left & right pulley positions

3-Protocol

Following is the protocol that will be used to format the communication data between goDog (android media-box) and the SwitchBot microcontroller.

3.1-Packets

All the communication data between the android module and the microcontroller (commands, notifications, requests, params) needs to be wrapped with a START_BYTE and END_BYTE.

START_BYTE	**0xFE**
END_BYTE	**0xFF**

Commands

Mnemonic	**hex**	**params**
MOVE_FORWARD	**0x10**	number of steps
MOVE_BACKWARD	**0x11**	number of steps
TURN_LEFT	**0x20**	number of steps
TURN_RIGHT	**0x21**	number of steps
BODY_CON	**0x30**	con codes/mnemonics - 0x01 **DANCE** - 0x02 **STAND_UP** - 0x03 **TRACK** - 0x04 **LEAN_FORWARD** - 0x05 **LEAN_BACKWARD**

Example 1 : voice command = " **move forward 5**"
parsed command = **0xFE - 0x10 - 0x05 - 0xFF**
Example 2 : voice command = " **dance** "
parsed command = **0xFE - 0x32 - 0x01 - 0xFF**

3.2-Requests

We define a request as a command sent from the android module to the microcontroller to get some information like SwitchBot status or the battery level. There's no parameters when sending a request. The size of all the packet requests is 3 bytes (<START_BYTE><request code><END_BYTE>).

Mnemonic	**hex**
GET_BATTERY_LEVEL*	**0x41**
GET_STATUS	**0x42**
GET_POSE_TOP	**0x43** request to get torso, left & right thighs positions
GET_POSE_BOTTOM	**0x44** request to get left & right calves, left & right pulleys positions.

* this feature will not be implemented in the version 1.0.

3.3-Notifications

We define a notification as an event coming from the microcontroller to the android module. This event is triggered after a request from the android module or directly from the microcontroller to inform it, for example, that the battery level is too low (**NOTF_BATTERY_LEVEL**) or that SwitchBot falls over (**NOTF_STATUS**) … etc. The notification **NOTF_VOICE_RECORD** is generated by a push button or an IR sensor to alert the android module to start recording the voice command.

Mnemonic	**hex**	**data**
NOTF_VOICE_RECORD	**0x50**	"no data"
NOTF_BATTERY_LEVEL**	**0x51**	0..20
NOTF_STATUS	**0x52**	- **0x01 STANDING** - **0x02 KNEELING** - **0x03 LAYING_DOWN** - **0x04 TRANSITIONING ***
NOTF_POSE_TOP	**0x53**	3 bytes of data : - **BYTE 1** : torso position - **BYTE 2** : left thigh position - **BYTE 3** : right thigh position
NOTF_POSE_BOTTOM	**0x54**	4 bytes of data: - **BYTE 1** : left calve position - **BYTE 2** : right calve position - **BYTE 3** : left pulley position - **BYTE 4** : right pulley position

***** This notifications (TRANSITIONING) will include two more bytes for the start and end positions. The packet will be as follow:
<START_BYTE> <NOTF_STATUS><TRANSITIONING><start_position><end_position><END_BYTE>

** This feature will not be implemented in version 1.0.

4-Project timeline & deliverables

Date	Deliverables
Nov 21st, 2014	- project plan - project RTM (Requirements Traceability Matrix)
Dec 2nd, 2014	ver 0.1 (5% of the requirements)
Dec 16th, 2014	ver 0.2 (50% of the requirements)
Dec 23rd, 2014	ver 0.3 (80% of the requirements)
Dec 30th, 2014	ver 1.0 (final version)

5-Affectations

Developer	Deliverables
Dean Weber	- Customized goDog module for SwitchBot - Server level development/configuration
Nick Morozovsky	- Microcontroller firmware - SwitchBot device (hardware)
Raouf Bensalem	- Android USB driver for the microcontroller - Behaviour handlers - Project documentation