## Модуль Скайп-Аватар

Модуль будет считывать текст всех входящих мгновенных сообщений , отфильтровывать необходимые и перенаправлять их на определённый контакт. Этот определённый адрессат может быть настоящим живым человеком или логером (бот или аватар, который просто записывает информацию).

Среди отфильтрованных сообщений могут быть и команды, которые могут быть выполнены другими модулями. Позже можно начать использовать специальные фильтры или дополнительные жесты или просто тригеры, присоедененные к событиям различных сенсоров или групп сенсоров.

Так же на следующем этапе этот модуль должен получать изображение, звук и видео и распознавать их или находить подобные в интернете.

**Пример модуля:**

#Вначале подключаем необходиме для работы этого модуля другие модули

import sys

import Skype4Py

import time

import os

import re

#Подключить нашего аватара к запущенному Скайпу

def OnAttach(status):

print "API Attachment status: %s" % skype.Convert.AttachmentStatusToText(status)

attached = False

# Пытаемся подключиться

if status == Skype4Py.apiAttachAvailable:

#skype.Attach()

while not attached:

try:

# Если скрипт внесен в Public API allowed programs - коннектимся

skype.Attach(Wait=False)

attached = True

# Если нет - ждем, пока внесут, а пока идем лесом

except:

pass

#Присоедениться к нужному чату если неполучилось подождать полсекунды и попробовать снова

def GetChat(chat):

print "Receved: %s %s via Skype" % chat.Body, chat.Sender.FullName

try:

idChatMessage = skype.Command("GET CHAT mfeudor CHATMESSAGES")

skype.SendCommand(idChatMessage)

print 'TERMINAL>>', idChatMessage.Reply

return int(idChatMessage.Reply.split()[-1])

except:

time.sleep(0.5)

return GetChat()

#Отфильтровать нужные сообщения.

def SendChats(tar\_user):

chats = skype.Chats

for c in chats:

# c.Messages is a tuple, to be able to sort it,

# convert it into a list

msg\_list = list(c.Messages)

msg\_list.sort(message\_timestamp\_cmp)

for Message in msg\_list:

if Message.Sender != ' ':

# Фильтр сообщений через поиск по регулярному выражению (шаблону)

r = re.search (r'Test (.+)', Message.Body)

if r:

msg = r.group(1).strip()

for member in Message.Chat.MemberObjects:

skype.SendMessage(tar\_user, msg)

# Обработчик сообщения на основе статуса сообщения.

def OnMessageStatus(Message, Status):

if Status == 'RECEIVED':

print(Message.FromDisplayName + ': ' + Message.Body+ ': ' + Message.Sender.FullName);

if Status == 'SENT':

print('Myself: ' + Message.Body);

if Status == 'SENDING':

print('Sending: ' + Message.Body);

if Status == 'READ':

print('Read: ' + Message.Body);

#Начало основной программы

# send init: username, port, state

skype = Skype4Py.Skype()

# Регистрируем слушателей

skype.OnAttachmentStatus = OnAttach

# Callback на входящее сообщение

skype.OnMessageStatus = OnMessageStatus

if not skype.Client.IsRunning:

print 'TERMINAL>> Starting Skype..'

skype.Client.Start()

print 'TERMINAL>> Connecting to Skype..'

#skype.Attach(Wait=False)

skype.Attach()

profile = skype.CurrentUserProfile

print 'TERMINAL>> Skype name: %s' % skype.CurrentUser.Handle

print 'TERMINAL>> Skype credit balance: %d' % Balance()

print 'TERMINAL>> Port: %s' % port

print 'TERMINAL>> State: %s' % state

print 'TERMINAL>> Proxy: %s' % started\_proxy

print 'TERMINAL>> MoodText: %s' % profile.MoodText

print

print 'TERMINAL>> SET AUTOAWAY OFF'

skype.SendCommand(skype.Command('SET AUTOAWAY OFF'))

# Отправить информацию о включении Аватара адрессату

skype.SendMessage('adress\_user', 'Start Avatar for Read Messages')

#Основной цикл скрипта-Аватара в котором он находиться всё своё основное время, фильтрация #срабатывает по прерыванию на входящее сообщение

mocpSongTitle = ''

currTitle = ''

while True:

SendChats('hueputalo2')

#SendChats('nosce\_duma')

time.sleep(10)

# (stat, currTitle) = commands.getstatusoutput('mocp -Q %title')

currTitle = 'Time > ' + str(time.time())

if currTitle != mocpSongTitle:

if currTitle != '':

mocpSongTitle = currTitle

profile.MoodText = mocpSongTitle

else: profile.MoodText = ':-)'

Полезно для организации правильной архитектуры проекта разбить функции на несколько групп:

1. Объектно-ориентированные (Иницилизация значений)
2. Машины состояний (Операторы if и case)
3. Приравнивание-пересылка (копирование и передача областей памяти и АЛУ)
4. Парсинг (поиск по регулярным выражениям)
5. Внутренние таймеры (Временная диаграмм)
6. Внешние события (Обработка прерываний)
7. Шина D-bus

**Эти 7 уровней и составляют новую методологию проектирования. На практике можно ограничиться 3-4 уровнями, причём крайний уровень включает в себя все последущие вплоть до 7. Полезно в наименовании функций точно указывать номер уровня и перед компиляцией программы строить полное дерево функций по признаку уровень с целью проверки соответствия выбранной иерархии.**

**Полученное дерево, можно выводить в виде скрипта для встроенного языка Python програмы PyMol.**

**Обработка естественного языка**.

Семантический парсер-переводчик будет испольован в этом разделе на 8 уровне.

1. Семантический Парсер Правда <http://www.prawda.newmail.ru/download.html> Так как у меня есть договорённость с автором о безвозмездном использовании этого продукта. **Основные направления превода Английский-Русский, Эсперанто-Русский, Можно добавить Испанский-Русский и Немецкий-Русский.**
2. По поводу семиотического анализа и прагматик веб. Все только начинают думать. Видимых стандартов нет.

**Использование нейросетей, марковских цепей и контекстно свободных грамматик.**

<http://research.microsoft.com/en-us/news/features/speechrecognition-082911.aspx>  
hybrid context-dependent “deep” neural networks Hidden Markov Model (CD-DNN-HMM)

**Open-source пример**<http://nlp.stanford.edu/software/stanford-dependencies.shtml>

**Система извлечения фактов**<http://inventionmachine.com/>

**Дистанционное обучение**  
<http://dret.net/lectures/>

При Хрущёве власти боролись за системы коммуникации. Сейчас у всех есть мобилка. Теперь пошла мода на телеприсутствие, но оно не даёт контроля. Телеприсутствие должно быть целиком дистанционно управляемым. Устройство должно быть вещь в себе, все функции должны быть удалённые или автономные. Хорошо бы получать максимум сенсоров от производителей цифрового контента с хорошей фильтрацией.