

User Task Automator

Himanshu Prasad¹, P. Geetha Priya², S .Manjunatha³, B.H Namratha⁴ and Rekha B. Venkatapur⁵

^{1,2,3&4} UG Student, ⁵HOD Department of CSE
K.S. Institute of Technology
Bangalore, Karnataka, India
E-mail: himanshu@outlook.in
geethapriya28.3@gmail.com, manja.js@outlook.com
nammu.harish2014@gmail.com, vb_rekha2000@yahoo.com

Abstract - We live in an era where we have powerful digital devices at our hands. In our busy lives, we can take help of technology to save time, by automating tasks of our life. But interacting with technology can be a challenge for people not very familiar with technology. Methods of interaction like voice commands are gaining popularity because it is easy for a person to just speak out what they want the device to do. Thus, we propose a program to let user specify, in natural language, an action to be performed when a certain event happens and perform it for him. A layman using a computer can't code, but many times they have a requirement to perform tasks which they wish to automate. This program acts as a platform where they can specify those tasks as commands in natural English language, and the program performs those tasks for the user. This way of combining any supported action with any supported event allows user to create large number of combinations to help them perform various tasks under various events.

Keywords: Task Automation, Natural Language Processing, Command Extraction, Event Listening

I. INTRODUCTION

A layman using a computer can't code, but many times there may be a requirement to perform small tasks which user wish to automate. It can be some routine task like starting few frequently used programs on the startup of their computer, or it may be something critical, like to be notified immediately when an important call comes. This program acts as a platform where they can specify those tasks as commands in natural English language, and the program performs those tasks for the user. Thus, it becomes very easy for the user to specify their tasks. It almost becomes like asking a real person to do tasks for them.

The software aims to support a set of events and resulting actions that it can take. The software exists as a background process on the device, always listening for any event that user may have specified. User specifies a command in natural English language, for example, "When internet connects, open Chrome browser". The software saves this command as a task. The software then proceeds to parse this command to understand the event that has to be listened for, connection of internet in our example, and the action that has to be taken on occurrence of the said event, starting the browser in this example. After this, the software continues to

monitor for the occurrence of this event. So, whenever the internet connects, the software starts Chrome for user.

The software is intended to be developed for personal computers. Although the events and actions need not be local to the device, it can be connected to web services, email for instance. Thus, it integrates the various aspects of a user's digital life and attempts to automate as many things as it can.

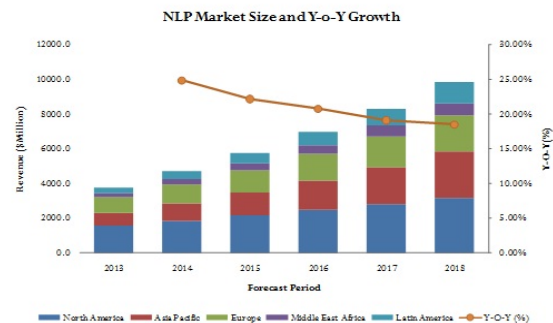


Fig.1 NLP Market Growth

The use of NLP has been growing year by year⁴. The factor behind this is the ease of use it provides to the user. The graph above speaks to the popularity of this technology. Due to popularity and ease of use, NLP based technology is implemented in the current work.

Voice recognition is the other requirement of a person who is not technology literate. A simple interface just shows the user the tasks he has added and lets him add more tasks. This lets anyone use the software easily, even if they aren't very tech literate, since the software doesn't bombard them with options.

II. SCOPE

The software can be used by anyone using a supported computer. Commands are expected in English language. Commands are supposed to be clearly specified, with an event and an action being mandatorily mentioned. The tasks will then be performed accordingly. This software is not meant to be a talking digital assistant, so user can't have a chat with it, nor can they use this to look up information.

This is used to be able to perform task on behalf of user when something happens.

III. PLATFORM

Computers are the device which most of us use for majority of tasks. But there can be small and repetitive tasks that we

have to perform, for example; it is tedious task to perform scheduled repetitive tasks, Say a task to start a program or shutdown/restart the system at a given time. So user can set the task in this software to execute the same after the events arrival. Thus, user becomes free to focus on more important task and don't have to be bothered about remembering and performing mundane tasks.

TABLE 1 EXISTING PC SOFTWARE

| Software | Platform | Memory Usage | Input Method | Services |
|-----------|----------|--------------|---------------|--------------------------------|
| RoboTask | Windows | 5 MB | Dropdown Menu | Event driven command execution |
| Automator | Mac OS X | 25 MB | Menu option | Storable workflow |
| Cortana | Windows | 7 MB | Voice | Instant answer / Results |

IV. EXISTING WORK

Following are some existing software that relate to our idea:

A. Automator for Mac OS X

Automator¹ can either record an action performed by user or create a workflow based on its options. But the glaring gap is that every time the user needs to run these tasks, he needs to execute them manually. The software doesn't support the ability to listen for a particular event and then execute tasks automatically. Also, it doesn't support Windows, which has the most market share for desktop computer

B. RoboTask



RoboTask² is a software that closely aims to what we propose. It provides a multitude of actions, and building on the shortcomings of the other software that we had a look at previously, Automator, it provides a list of events that will automatically trigger the execution of particular task.

But even this software has some gaps left that can be filled. It has a very restricted list of actions and events, and they are mainly work oriented. It is good from workflow point of view but doesn't extend to other personal tasks. Also, even this software has no companion phone component. And it is a paid software, costing \$119.95 for a personal license at the time of writing. That converts to just a little above by the current exchange rate, which is difficult for many people to afford in our country.

C. Digital Assistants

Digital assistants mostly help on phones and can perform small tasks for us like playing a song or starting an app or setting a reminder. But they have very limited set of tasks and all of the tasks are performed immediately. The tasks cannot be set for future nor can they be associated with an event whose occurrence should automatically set the task in motion. And most of them don't have a desktop component. Only Cortana³, developed by Microsoft, exists on both their computer OS and the phone OS. But it also doesn't support creating tasks and events. Digital Assistants are mainly aimed to provide live information, which is not the scope of this project. So, they only provide a small subset of the features of this software.

D. Shortcomings in existing work

1. No universal solution.
2. Existing solutions can be expensive.
3. The existing solutions bombard the user with options, which can overwhelm a layman and he may not be able to explore or understand all the available options.
4. No way to specify the task using natural language.
5. Limited number of tasks.
6. No way to interact with voice.
7. Mostly productivity oriented, less support for personal tasks.

V. OUR SOLUTION

Considering the existing solutions that already exist in the market, our software fills the gap left by these products and improves and extends their existing functionality.

A. Voice Based Interaction

Most of the existing software interacts with complicated interface. They bombard user with options, which are hard

to find, in an interface that is hard to navigate. It might be especially difficult for people not used to technology that much.

This software aims to solve the problem by providing a voice based interaction option. The user can press the microphone icon to record his command. Maximum length of the recording is 15 seconds. After that the record audio is processed and the speech is converted to text, which is automatically inserted into the text box. Even further confirmation to accept or cancel can be done by the use of voice.

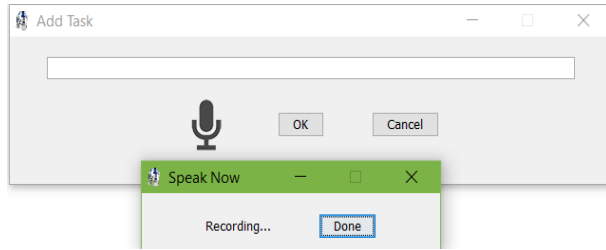


Fig.2 Voice Interaction

Even though voice recognition technology has come a long way, it's still not foolproof. Keeping that in mind, the software shows the command in an editable text box. This allows the user to correct any mistakes happening in the recognition process.

B. Natural Language Command System

This software doesn't overwhelm user with a long list of actions. Instead, the user can specify their command, which is a set of actions to be performed and events that will trigger them automatically. The software then extracts this data from the given command and creates the task. Thus, a layman with not much technical knowledge can also easily create tasks without digging through a lot of options.

For this system to work, it is making use of NLP (Natural Language Processing). Natural language processing is a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between computers and human (natural) languages. As such, NLP is related to the area of human-computer interaction. Many challenges in NLP involve: natural language understanding, enabling computers to derive meaning from human or natural language input; and others involve natural language generation. We need to overcome these challenges in order to parse our user command and extract the necessary details from it.

The given query is sent over the internet to be processed to an agent trained with a set of queries related to the supported features. It extracts the intent of the user, which refers to the action. It then looks for a defined set of parameters, which become our event. Finally, the result is returned in a JSON

format, which is then inspected and accordingly a task is added to the software's task list.

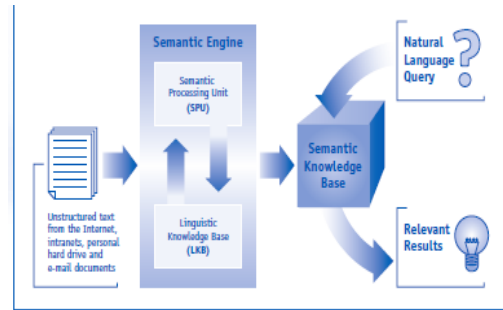


Fig.3 NLP Architecture

```

{
  "id": "e5a392d5-aefd-4292-bafa-782b07d14a9f",
  "timestamp": "2017-05-03T18:53:06.161Z",
  "lang": "en",
  "result": {
    "source": "agent",
    "resolvedQuery": "in 10 minutes remind me to call my mom",
    "action": "",
    "actionIncomplete": false,
    "parameters": {
      "battery_level": "",
      "internet_connected": "",
      "internet_disconnected": "",
      "percentage": "",
      "reminder": "call my mom",
      "time": "00:33:05"
    }
  },
  "contexts": [],
  "metadata": {
    "intentId": "926fbac9-e675-458b-aff8-078075fcbf0c",
    "webhookUsed": "false",
    "webhookForSlotfillingUsed": "false",
    "intentName": "alarm"
  },
  "fulfillment": {
    "speech": "",
    "messages": [
      {
        "type": 0,
        "speech": ""
      }
    ]
  }
}

```

Fig.4 Sample JSON

C. Event Action Model

This software follows an event-action approach. It supports some events which it can wait for to happen, and actions which are to be taken when the aforementioned event occurs. Thus, by having a feature set of events and actions, the user can possibly create a lot of combinations, thus giving them freedom to automate a lot of tasks. And rather than following a preset sequence, it allows the user to create functionality of his or her choice.

The software keeps running in the background all the time. At the time of launch, or whenever the task list is modified, it reads the task file that it maintains. The tasks are stored in an XML format. It reads each and every task, find the event associated with that task, and launches a new thread listening for that task. The threads then keep waiting for that event to occur. As soon as that happens, the associated action is performed and the particular task is removed from the task list.

D. Modular Structure

The software has a modular structure, which makes it both flexible and powerful to use and easy to develop. The set of supported events and actions are not linked to each other. Thus, rather than being restricted to a particular functionality, the user is free to create his or her own functionality by combing any of the actions with any of the events. Thus, rather than limiting a user’s options, the software gives them more freedom to create their own use.












- >  CopyFile.java
- >  DeleteFile.java
- >  Downloader.java
- >  InternetDisconnects.java
- >  Kernel32.java
- >  ListenerForNetwork.java
- >  ListenerManager.java
- >  MoveFile.java
- >  Power.java
- >  SpeechRecognition.java
- >  Start.java

Fig.5 Few Components of The Software

The modular structure also helps in the development of the software. The actions and events are split into their individual classes. And we have created a standard interface for them to interact. Thus, this gives the developer the power to easily add and remove feature sets, which makes them like “pluggable” components. We can even change the complete implementation of a feature without breaking anything. This allows for easy and rapid development of new feature sets, thus allowing us to easily increase the number of events and actions that the software supports.

VI. ADVANTAGES

While developing this software, special care has been taken to make it as easy to use as possible, and fill the gaps that other software is not filling.

A. Universal Solution

All of the present solutions are very limited in the number of tasks they perform. In these systems, the number of tasks is limited and they focus on productivity related work. Some solutions can automate very specific tasks accurately. But most of them don’t support other tasks that a layman can use. Thus, this software is designed to support all tasks a normal person may require in his day to day life, including both professional and personal tasks.

B. Cost Effective

Existing solutions can be expensive for many of the users to afford. To allow more users to make use of our software, we

have developed it as a freeware and provide service based on no cost. Furthermore, the software doesn’t need any special hardware or software to run, thus incurring no extra cost to the user.

C. Ease of use

A key advantage it provides is the ease with which the software can be used, even by a person not well versed in technology. It takes input as a normal English sentence. The input is accepted as voice also which can be helpful for physically challenged people. Thus, the interaction of the user with the software is a breeze.

D. Flexibility

Rather than fixing a particular functionality for user, the software simply provides a set of supported features. Since it is not a tailor-made software, user can add events and actions as per need. The user is then free to combine them in the way he or she wants, thus creating their own use case. This gives the software a lot of flexibility in terms of what it can do.

TABLE 2 BUILDING BLOCKS

| | | |
|------------------------|---------------------|--|
| Computation | Hardware (Computer) | New generation processor, RAM meeting minimum requirements to run OS |
| | Software(Computer) | Windows OS, 7 or later, Java Runtime 1.7+ |
| Command Identification | | Using NLP |
| Communication | | Using Internet with decent speed |
| Services | | Taking the input in natural language, analyzing it and serving them when the specified event occurs. |

E. Multi Platform Support

None of the solutions that relate to our idea are multi-platform. They exist only for a particular platform. We made the choice of the language of this software to be Java, the reason being portability of code. Leaving few snippets of code that make use of some OS specific functionality, the rest of the code is readily portable to other platforms.

F. Batch Processing

Rather than the user adding tasks one by one, this software supports taking the input from a text file which contains predefined commands. The software parses the file and processes each of these commands and adds them as tasks.

Thus, the user can at once add lots of tasks rather than specifying them one after the other.

VI. RESULT AND DISCUSSIONS

The resulting software provides automation of the user tasks. It starts in background, listening for all the events for the added tasks. It only allows a single instance of the software to run at a time. It features a simple interface, displaying the existing tasks and gives 3 options, to add task, to add multiple tasks from a file, and to delete tasks. Adding task is done easily through voice recognition, and then the command is processed by NLP agent and stored as a task. It also asks for any extra information, if required to perform that particular task.

TABLE 3 PERFORMANCES

| | |
|-------------------------------------|--------------|
| Average RAM Usage | 26 MB |
| Disk Space Usage | 8 MB |
| Average Processor Usage | 5% |
| Average Startup Time | 500 ms |
| Average Time for Speech Recognition | 6 seconds |
| Average Time for NLP Processing | 2 seconds |

TABLE 4 FEATURE COMPARISONS

| FEATURE | EXISTING SOFTWARE | | | UTA |
|------------------------|-------------------|----------|---------|-----|
| | Automator | RoboTask | Cortana | |
| Voice Interaction | No | No | Yes | Yes |
| Cross Platform Support | No | No | No | Yes |
| Batch Processing | No | No | No | Yes |
| Event based | No | Yes | No | Yes |

VII. CHALLENGES ENCOUNTERED

1. Making sure that most of the code is portable
2. Processing the user input using Natural Language Processing

3. Including voice based interactions
4. Training the NLP agent
5. Running the tasks in background effectively and efficiently
6. Creating a standard interface for events and actions, allowing for easy addition of features

VIII. FUTURE WORK

This software aims to be a universal solution for the automation and thus its features can be extended to include more event and action set in future. Smartphones can also be supported in future, with the ability of interaction between the computer software and the phone app.

IX. CONCLUSION

This software aims to simplify user's lives by giving a platform to automate their tasks. Thus, users don't have to worry about lots and lots of routine tasks, they can set it once in this software and forget about it. Then they can focus on other important tasks that demand their attention.

This software aims to provide better functionality than any other solution currently present in the market. It involves NLP and makes sure the user can easily specify the input as natural language rather than being overwhelmed with various options. And it makes the interaction voice based, which makes it even easier to use, even for the non-technical population.

X. ACKNOWLEDGEMENT

Authors would like to thank VGST (Vision Group on Science and Technology), Government of Karnataka, India for providing infrastructure facilities through the K-FIST Level I project at KSIT, CSE R&D Department, Bengaluru.

REFERENCES

- [1] Automator for Mac OS X. (www.macosautomation.com)
- [2] RoboTask (www.robotask.com)
- [3] Cortana (<https://support.microsoft.com/en-in/help/17214/windows-10-what-is>)
- [4] <http://www.marketsandmarketsblog.com/natural-language-processing-market.html>
- [5] <https://api.ai>
- [6] <https://cloud.google.com/speech>