COM SCI 131 Project
Twisted Places Proxy Herd
Report

Weifeng Jin, 904429456
University of California, Los Angeles

November 30, 2016

Abstract

Twisted is an event-driven networking engine written in Python and is a potential framework we would like to use in our implementation of an "application server herd", where the multiple application servers communicate directly to each other as well as via the core database and caches. This report examines the advantages and disadvantages of the framework Twisted in the process of the implementation in terms of its simplicity, efficiency, convenience, readability and reliability.

Keywords: Twisted, Python, application server herd, programming language design

1. Introduction

Wikimedia Architecture uses a LAMP platform, which composed of structures from GNU/Linux, Apache, MySQL and PHP. In order to improve and guarantee reliability and performance, it uses multiple, redundant web servers behind a load-balancing virtual router. However, with the development of web servers and new technology, we need to develop a new and completely different structure called "application server herd". Such a framework is dedicated to rapidly-evolving data, and we are using Twisted and event-driven programming to implement this structure.

In this project, for a given network structure, an application server herd of 5 servers is created. The server herd is responsible for processing location and position queries from the users. The implementation of the server herd includes error handling and massive logging requests for exchanging messages.

2. Twisted

2.1 Twisted Overview

Twisted Internet is a collections of event-oriented loops written in Python. It has the code do dispatch events to different observers and a portable API for users. Hence, it is possible for clients using Twisted to apply the same code to different loops. The main part of the Twisted is the design of a reactor, which is the main event loop that processes queries from several different clients in the same time and dispatches these queries to the corresponding processors.

Also, since Twisted is based on a "single thread-like" implementation model to enhance efficiency and reliability, the reactor does not block the processing of implementation. Multithreading in the structure might lead to problems including but not limited to synchronization concurrency, and complicated syntax.

2.2 Twisted Syntax and Language Suitability Issues

Since Twisted is written and implemented in Python, the syntax is simple and the program is relatively simple to write. However, there exists a debate about whether Python is a suitable language for the structure.

2.2.1 Type-checking

Since Python does not do static type checking, it increases the number of runtime errors. Python has a strong dynamic type checking system embedded. A dynamic type checking simplifies the process of writing a program. Dynamic typing makes it easier to use multiple interfaces. However, it increases the difficulty of the understanding of the code. Python programmers do need to look into the code to get the usage of certain types, while for languages like C++ and Java, programmers can easily see what these types are.

2.2.2 Memory Management

Since the implementation of Twisted does not require a large quantity of memory space, the issue of memory management is less obvious and important here. Also, since our application server herd is not a real-time system, we do not really need garbage collections. Thus, although lacking proper memory management mechanics, Python is still a good candidate for the structure we are going to develop.

2.2.3 Multithreading

As mentioned above, although Python has support for multithreading implementation, Twisted does not implement its program with multithreading. Instead, it uses a single core for most tasks and uses asynchronous I/O to improve the efficiency.

3. Implementation

3.1 Overview

In the programming prototype, most of the code, including different processors/handlers for IMAT, WHATSAT, AT is written in the ServerHerdProtocol class. The ServerHerdProtocol class creates a factory that initializes the protocol every time when the server is connected. The reactor (imported from twisted.internet.protocol) listens to a specified TCP port and invokes the corresponding factory every time when it is connected.

3.2 ServerHerdProtocol

The ServerHerdProtocol class is the most important part of the implementation. It has the following method declarations: connectionMade, connectionLost, lineReceived, process_failure, process_IAMAT, process_WHATSAT, process_Google, process_AT and location_update. The ServerHerdProtocol uses the LineReceiver protocol, which, after receiving message from the input, dispatches the corresponding method do deal with the input. In my implementation of the protocol, it first checks the number of connections and use the lineReceived method to dispatch the processors/handlers to deal with the input.

3.3 IAMAT Handler

The main part of the IAMAT handler is written in the method of process_IAMAT. After receiving a client message, the IAMAT handler first checks if it is a valid command by length checking. Then, it splits and stores the original message and client time stamp, then uses the location_update method to send the message to the neighbors of the current server and propagates through the entire herd. If the execution fails or the input is not a legal input, the method would return with a piece of error information in the log.

3.4 WHATSAT Handler

The main part of the WHATSAT handler is written in the method of process_WHATSAT. Similarly as process_IAMAT, after receiving a client message, the WHATSAT handler first checks if it is a valid command, splits and stores the original message correspondingly. However, I cannot get the desired result for the WHATSAT input sometimes and I do not really know the reason why…

4. Advantages and Disadvantages of Twisted

Both Twisted and Python are good candidate for the implementation and design of application server herd structure. They have various important advantages in the design process. Such as:

4.1 **Simplicity**

Since Python is a concise programming language, and Twisted has straightforward instructions and documentations, writing the program should not be too difficult and time-consuming. Also, because Python is a very popular programming language and Twisted has a well-formed community, we can easily seek help from the internet. Also, the Twisted website provides a large number of detailed coding examples, including the chatserver.py.

4.2 Event-Driven Model

Twisted is a well-developed and carefully-designed structure for an event-driven model. Twisted implementation uses a lot of inheritance classes and we have to write the handlers by our own. We can modify the handlers or classes with our own need, which provides freedom for the clients and programmers.

However, Twisted + Python does have some drawbacks in the design process, such as:

4.3 Dynamic Type Checking

Although dynamic type checking enables programmers to write the code faster, it also makes the reader more difficult to understand the intention of the programmer, and makes the maintenance work hard to progress.

4.4 Error Handling and Debugging

Since Twisted and Python do not have proper IDEs to support the programmer, it is quite difficult to debug and find related errors sometimes. When I was doing the project, sometimes the compiler would yell at me "Hey the program does not work, Here's what the problems are …" with a lot of library errors which I do not understand. Also, since the Twisted structure is quite complicated and multi-leveled inheritance, it takes a lot time to find where the "true" bug is.

5. Testing the Implementation of Twisted

Python does have some support for testing the program and it is important to test the programs during development since Python has dynamic type checking and lazy-evaluation – as mentioned above, sometimes these "advantages" might turn into drawbacks to make the debugging process extremely painful.

For testing the program, I simply use the queries(inputs) in the course website. First I sends and IAMAT client query to server Alford to test if the message is successfully sent to the neighboring servers. As mentioned above, I must confess that I cannot get the same results for the WHATSAT command.

6. Comparison between Twisted and Node.js

Node.js is an open-source, multi-platform JavaScript runtime environment for development of various instruments and applications. Node.js can also be used to develop a "application server herd" in our practice. Similar as Python, Node.js also provides event-driven API and real-time applications, a large number of corporations, including IBM, Microsoft and Yahoo are using Node.js for their server-end applications.

6.1 Single Threading

Both Node.js and Twisted(Python) implements the application using single threading rather multi threading. They also uses a unblocked I/O to support concurrent linking requests and calls. The drawback of both implementation is that they cannot deal with multicore systems.

6.2 Development Issues

Unlike Python, which lacks a dedicate IDE to help with illustration, visualization and debugging, Node.js has various tools to support its development, including desktop IDEs and even online compilers. Although Node.js is a relatively new language comprated to Python and Twisted, it has gained a lot of popularity and recognition over recent years.

However, Node.js does not offer packages like ServerFactory and SeverProtocol, which are useful in our implementation and supported by Twisted. In terms of developing the same application in Node.js, we need to define our own classes and create new objects correspondingly, which adds to the difficulty of development.

6.3 Performance

Several online resources have claimed that Node.js has better performance than Python in terms of server-end applications. The main reason for the improvement in performance is due to the use of V8 engine. V8 is an open-source JavaScript engine developed by Google.

Instead of real-time interpretation, it compiles JavaScript code to machine code before executing it. Also, it uses a lot of optimization techniques, including inlining, elision and inline caching to improve the performance.

6.4 Twisted vs. Node.js

All in all, although Node.js has a lot of great features for out application server herd, it is a relatively new language and lacks extensive community support. In this sense, Twisted is a more mature framework than Node.js and the Twisted community has a lot more support and resources to offer since it has a much longer history of development.

Also, in the sense of development, although Twisted + Python lacks some ideal debugging tools for the ease of development, it has more extensive library support for server-end applications than Node.js, which might be a great help in developing our applications.

7. Conclusion

In the project, I use Twisted to create a proxy server herd application. The inter-server communications are designed for a large quantity of quickly-changing data, especially the GPS location information. Also, in the application we use the Google Places API to support our research. From the discussion and analysis above I think we can safely conclude that Twisted is a proper candidate for our application of an "application server herd" architecture.

Python and Twisted have a lot of great features and disadvantages to support our implementation (although some of them have some drawbacks), including dynamic type checking, concise syntax, extensive community support, memory management, single threading and event-driven model design. The disadvantages of the Twisted framework include difficulty in debugging, lack of proper garbage collection designs, and support of multicore systems. Compared with the disadvantages mentioned above, these advantages are far more important and should be takin into consideration of the process of choosing a suitable framework. The reasons for that have been discussed extensively in the report.

As for the comparison between Twisted + Python and Node.js, I would argue that overall Node.js might be a better candidate here for the server-end application. However, since I do not have much time to dig in the details of Node.js application and I am not familiar with the language, this claim might be inaccurate and biased. Although Node.js is extremely popular on the internet for this kind of application these days, I would still agree that Twisted is a suitable candidate for the project since Twisted is a more mature framework and provides more library built-in protocols, it might be easier to develop our application in the structure of Twisted since we have more access to the community support and documentation records.

8. Referneces

[1] Twisted website: http://twistedmatrix.com/trac/

[2] Fall 2016 COM SCI 131 Project Specification: http://web.cs.ucla.edu/classes/fall16/cs131/hw/pr.html

[3] Wikipedia page description of Node.js: https://en.wikipedia.org/wiki/Node.js

[4] The Switch: Python to Node.js: https://journal.paul.querna.org/articles/2011/12/18/the-switch-python-to-node-js/

[5] Twisted Introduction: http://krondo.com/an-introduction-to-asynchronous-programming-and-twisted/