Team Meeting

Jan 31

Agenda

- Code Reviews
- Configuration Strategy
- Thinking About Scaling
- Web Services

Andy agenda item - Need script to populate db… Provide a shell script that given a database name as an argument (db exists postgres has already been started) will create the tables and populate enough of the database to generate a report.

A few report messages

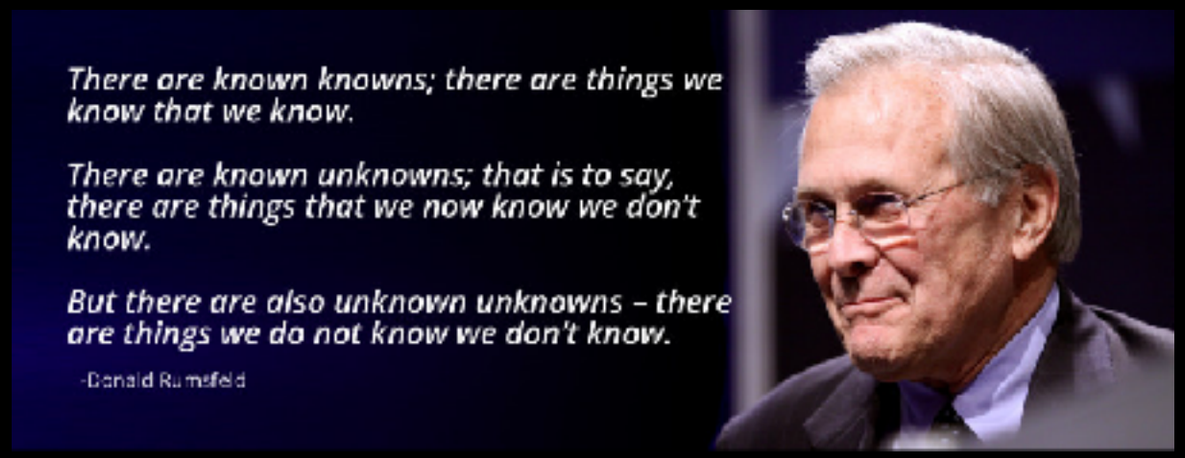
- Completed Assignment 2. This assignment took very long and felt extremely tedious.
- I will leave assignment 2 how it is right now and start working on assignment 3. Reading the instruction for assignment 3.
- I've never used psycopg2 before(although I've discovered it doesn't seem too different from the sqlite3 module, which I have used)

No real systems are fun all the time. Most systems have some portion that is not interesting that must be completed for success and many times there are obnoxious edge cases that must be handled along the way.

The term project is intended to be mostly a regrading of all of the incremental parts produced along the way. This models a strategy for dealing with small defects during a project timeline with a hard delivery date. This strategy is a bit of a gamble since success involves being able to get ahead enough in a later sprint to be able to correct the earlier code. The benefit is that the customer is more likely to remain calm since visible progress is still being made.

This is a useful insight; I don’t know a technology but I know something very much like it. That helps to assess risk (I know some of how this might work and how it might break) and where to focus on the time line.

Managing Risk



Last week we talked briefly about project timelines. I wrote up and posted my personal work breakdown for assignment 3. I still haven't had a chance to write any code for assignment 3 but it has been interesting to see some of the risks identified in my timeline being exposed.

Two of them are motivating lecture topics for today:
1 - The database should probably be configurable and the instructions don't say how… What should I do?
2 - mod_wsgi seems like a lot of work (re-session weirdness), why would I want this?

This Donald Rumsfeld quote got made fun of a lot when he first said it… but the breakdown is actually very good for anyone trying to assess risk. There are things we know could go wrong, these are easy to plan for or plan around. There are parts of our plan that we don't know enough about, something likely could go wrong there but we don't know enough to identify what could go wrong. Then there are all of the things that we don't see coming.

"Confidence, that quiet self assured feeling before you understand the situation you're in."

Code Reviews

- What do you want out of a code review?

- No code is perfect.
- Almost no code is completely irredeemable.

Computer programs encode how someone thinks about and approaches a problem. Reading other people's code will expose you to how other minds are grappling with the same challenges you've been working on. Hopefully you will enjoy the exercise. I'm hoping to get the first round of code review assignments out on Wednesday.

Starting up someone else's application can highlight configuration/executing assumptions.

Differences are the interesting part. How did I solve this problem? How did they solve this problem? What is good about their solution? What is unclear in their solution?

Configuration

- Why do we want software to be configurable?
- What should be configurable?
- Who needs to be able to modify configurations?
- What are reasonable expectations for the user doing configuration?
- How do we attack the design challenge?

Let’s prime with a few questions…

Common Solutions

- Hardcode anyway
- Configuration variables in a single source file
- Required parameters at application start time
- Use a configuration file
  - INI style?
  - XML style?
  - JSON style?
  - Invent something new?

We could ignore the configuration problem and just hardcode the data. A lot of research code does this; it is one of the reasons why research code is a nightmare to actually work on/with.

We could tell the user to edit a source file and set the values. This is easy for the programmer but exposes the internals of our program to the user. If they do something to misconfigure the system, the program crashes instead of generating a configuration error message.

We could handle all of the configuration through parameters, that is what the earlier assignments did. This assumes the program will be started in a way that accepts parameters and that the user is willing to key in all the parameters at every launch.

We could use a configuration file, which is what a lot of applications actually do. The configuration file is usually stored at a well known location or passed in as a parameter. The “-D <path>” when starting postgres is the parameter that points the software to the correct configuration. We’ll need a configuration file in our context due to how mod_wsgi is configured to launch our applications.

If we introduce a configuration file, then we introduce the problem of defining the configuration file format and writing code to at least parse that format. What are your thoughts on the file format we should use? What are the configuration parameters we should expect?

Decision will be included in post-lecture notes and appear in the next revision of the LOST requirements document.

Thinking About Scale

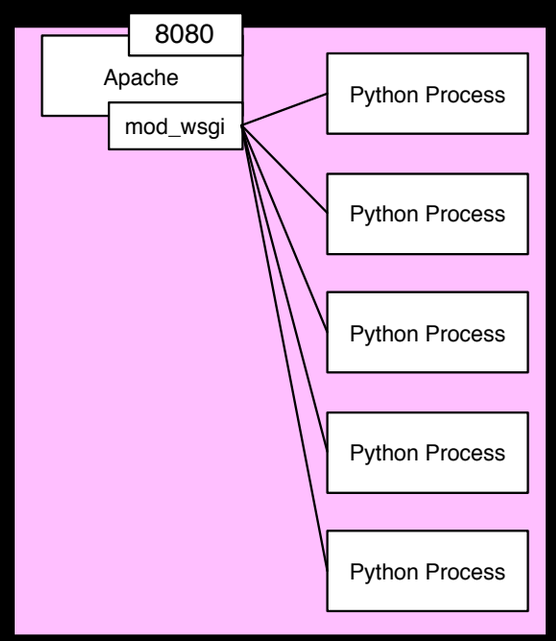
- Thinking about rate - thought exercise
  - 1000 requests per second
  - 10 milliseconds per request
- Thinking about volume - case study
  - 15 TB of data per day
  - 100 Mbps link
- Concurrency helps but brings other challenges

In both cases, you have a bad time. The system simply cannot keep up. In the first case, you could try to use a more efficient data structure or algorithm, you might be able to win. In the second case, you would need specialized hardware to be installed which could be challenging.

Rather than have one process do all of the work, the work can be spread over several processes. In some cases, adding concurrency is pretty trivial. In other cases, this can be really hard since there may need to be agreement between independently executing code. Operating systems and the parallel course should both go into these challenges in much more detail that we can in this class.

mod_wsgi

- Python is interpreted and can be a little slow
- Python is single threaded... even with python threads
- Apache/mod_wsgi use C to do a bunch of preprocessing
- A pool of Python processes are kept to service requests

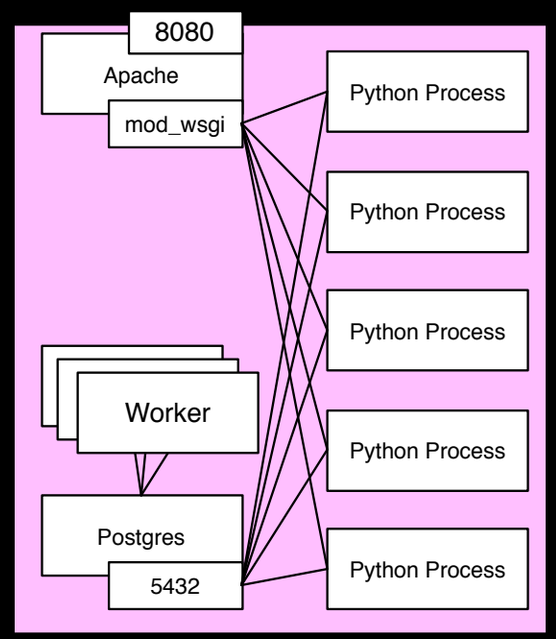


The main value add for mod_wsgi is to decrease latency and increase concurrency within a single host. mod_wsgi uses a pool of python processes. A pool of 5 processes on a computer with more than 4 cores should allow 5 requests to be processed in parallel. Each process, however, is trapped in its own memory space and can't see the memory of the other processes. If data needs to be shared, that needs to occur outside of the python process memory space. mod_wsgi may also restart python processes at anytime.

New challenges exist for data that must persist across requests (e.g. session and asset data)

mod_wsgi + database

- RDBMS provides ACID
- Atomicity - all or nothing
- Consistency - data always remains valid*
- Isolation - concurrent activity could have occurred serially
- Durability - completed work stays complete



Databases can be used to help address challenges with concurrency and accessing a reliable single view of the data across several concurrently running processes. Traditional RDBMS solutions provide several features that are extremely helpful.

Atomicity - if several things are done in one transaction all of them will be committed or none of the transactions will appear to have occurred.

Consistency is limited to the constraints that have been articulated to the database. For these constraints, the database will verify that all of the checks pass before allowing a transaction to commit.

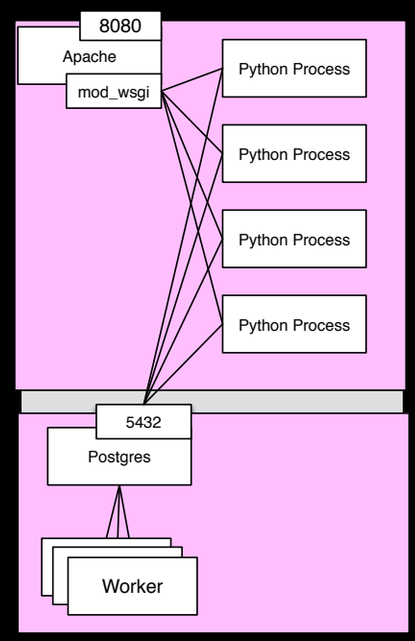
Isolation keeps concurrent users from interfering with one another. If several concurrent transactions are executing, the results will match some serial order in which the transactions could’ve occurred.

Durability is the property that once the database confirms that work has been committed that result is not lost.

All of these guarantees have a computational cost. No-SQL solutions will trade away one or more of these guarantees to get performance. If you work on a system that uses a No-SQL solution, you should probably think about which specific guarantees you need and wether or not your No-SQL solution provides them.

Multiple Hosts

- One host may not be enough
- Need to make decisions about where to separate the work between processes and nodes
- A common pattern involves a load balancer, application layer, service layer, and database layer



It could be that there is too much load for a single server. Multiple servers may be used to solve this problem… adding design problems about where and how to segment the work.

A common pattern is to have a load balancing layer (route network traffic without looking at the requests), that communicates with an application layer (web server and web application code), a service layer (functions that are needed for the web application code), a database layer (where all of the persistent data is stored). These architectures are really heavy and slow but can also make reliability easier to achieve. One of the biggest disaster projects I saw tried to use this architecture; the project failed in large part due to the additional complexity and latencies of the approach.

The Service Layer...

- Generally I think service layers are overhead
- Web services however can be useful for automation
- Generally provides an abstraction like functions

I usually don't provide service layers when building web applications. A service layer is to some extent another distributed application to build between the data sources the customer cares about and the user interface that the user cares about; middleware is hard to show to the customer in a way they feel good about.

Service oriented architectures (SOA) were all the rage for a little while… Data owners would provide functions to access their data. Any group in the enterprise would then use those functions to back their own applications. There are a host of performance problems and engineering problems that made this not happen.

Web services can still be useful however since providing an interface that looks like a function is much easier to use than screen scraping web pages.

RESTful Webservice

- HTTP
- Client/Server
- Stateless
- Uniform Interface

All web services run on top of HTTP. WSDL type services support a bunch of features that run counter to the HTTP design, RESTful web services are fairly compatible.

URIs are used to identify resources. Responses are documents, typically something like JSON or XML.

Assignment 5

- Assignment 5 hasn't been written yet, but I expect to post by Friday
- LOST requirements document will also need to be updated prior to assignment 5 posting
- Assignment will be to implement the 5 web service calls in the documentation