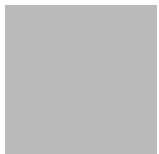





Daniel J. Lauk :: Software Engineer :: Paul Scherrer Institut

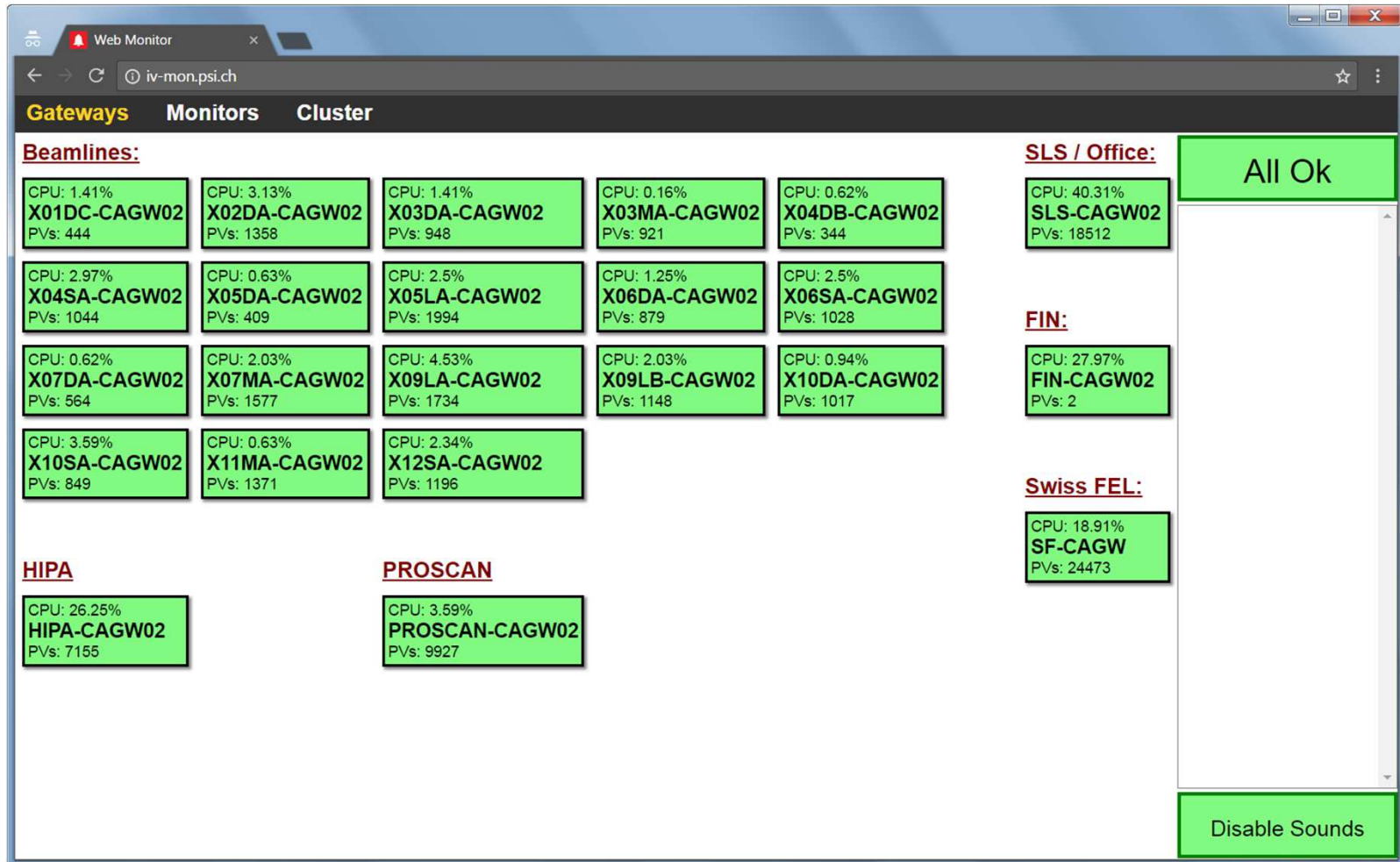
Lessons learned implementing a Channel Access gateway in Python with pyuv

EPICS Collaboration Meeting, 2017-05-18, Osaka




- 
- Written in C#
 - Running on VMs
 - Windows Server 2012R2
 - 1 CPU 2 cores
 - 2 GB RAM
 - 60 GB HDD
 - Central configuration through web interface (cached offline by gateway)
 - Search locations (like EPICS_CA_ADDR_LIST)
 - Access control
 - Live monitoring
 - Watchdog
 - Running 23 gateways in production (some more for testing)


Live gateway monitoring



Why write another gateway?

- 
- It's fun
 - Problems with the existing gateway: One certain bug...
 - Occurs very rarely
 - Not reproducible
 - Logging everything is not feasible
 - Different approach (technology stack, programming language, async. APIs...)
 - Encourage new/different ideas
 - Maybe certain bugs will be avoided at all
 - Maybe the bugs will be easier to track down
 - Last, not least: Learn something

The Twelve Networking Truths


- 
- RFC 1925
 - Published April 1st 1996

“Some things in life can never be fully appreciated nor understood unless experienced firsthand. Some things in networking can never be fully understood by someone who neither builds commercial networking equipment nor runs an operational network.”


“It is more complicated than you think.”


<https://www.ietf.org/rfc/rfc1925.txt>

The mission, should you choose to accept it...

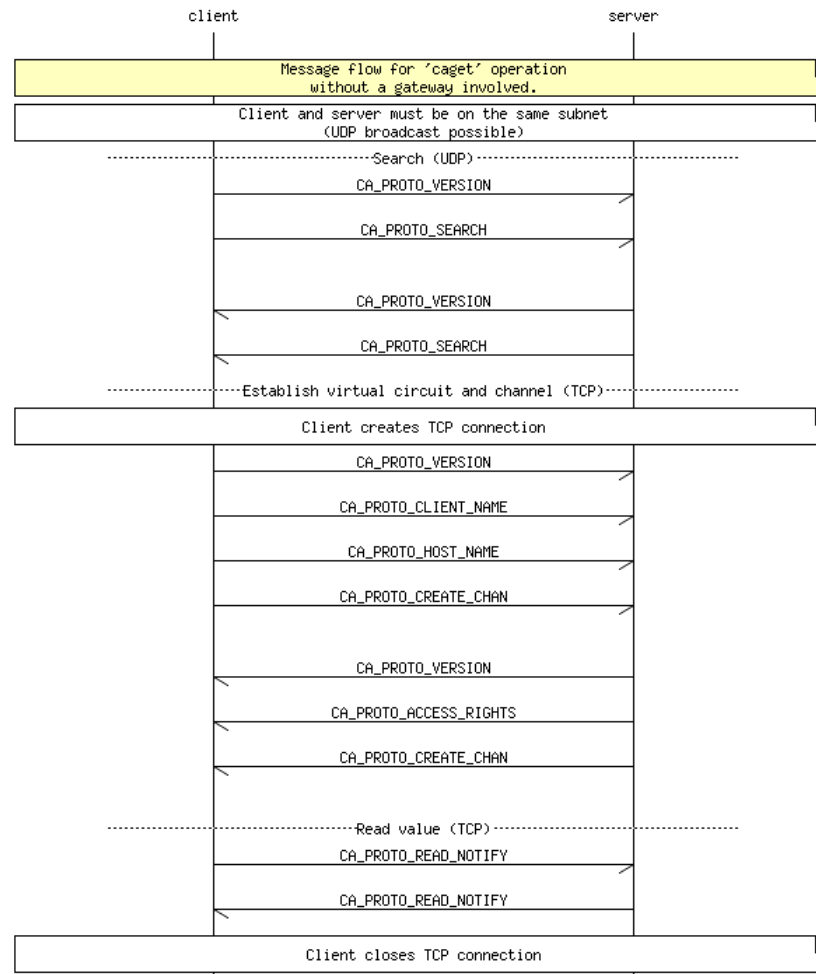
- 
- Develop a **prototype** gateway
 - Use language of your choice
 - No need for production hardening (i.e. leverage pareto principle)
 - Only 1 requirement: Must support read operation (i.e. caget)
 - Optional: Try supporting multiple clients

Why Python?

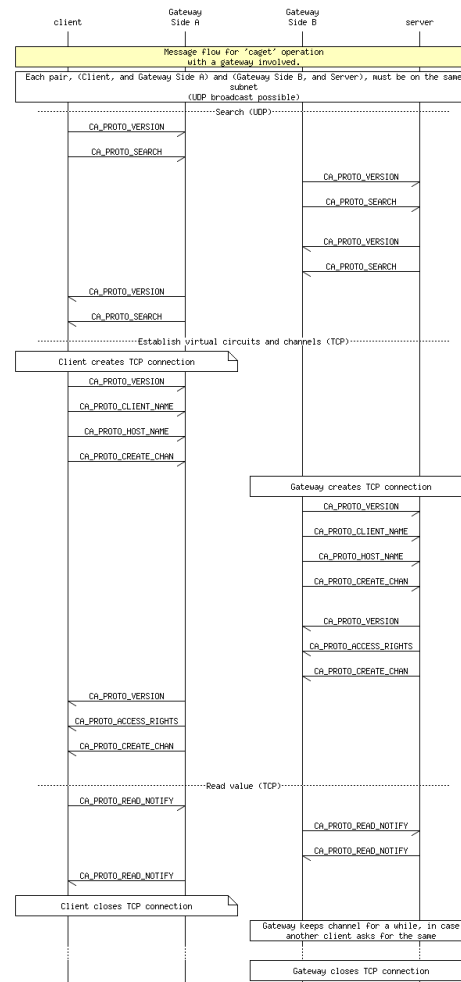
- 
- Alternatives, that I considered (I'm a language nerd) :
 - Rust
 - Go (golang)
 - Erlang
 - Haskell
 - F#
 - Personal familiarity
 - Syntax
 - Standard library
 - various 3rd party libraries
 - Well-known language (in general and at PSI)
 - Multi-Platform (just in case)
 - Good support (tools, community, documentation)

- 
- pytest
 - Write tests in a simple way
 - Python file should be called test....py
 - Tests are simply functions (no class hierarchy) with name test_...
 - Instead of fancy helper methods just use **assert**
 - pylint (static analysis)
 - Pedantic (generates **lots** of error messages and warnings)
 - It's actually right (most of the time)
 - yapf (code formatting)
 - Takes care of nearly all «unimportant» messages from pylint
 - It actually gets formatting right (most of the time)
 - flake8 (code complexity analysis)
 - conda (virtual python environments)
 - pyuv (asynchronous I/O through libuv)

Conversation without a gateway



Conversation with a gateway



So I wrote a test...

- Using pytest
- Make sure, I understand, how generators work
- Only showing 1 of the tests here (font size)

```
import pycagw.helpers
```

```
def test_id_generator():  
    """Test id_generator."""  
    expected = [4, 6, 8, 10, 12, 4, 6, 8, 10,  
                12, 4, 6, 8, 10, 12]  
    for e, a in zip(expected,  
                    pycagw.helpers.id_generator(4, 12, 2)):  
        assert a == e
```

...and a module to be tested

- In file pycagw\helpers.py:
- (Docstrings omitted to fit on slide)

```
def id_generator(start: int=0, max_id: int=0xfffffffff,
                 step: int=1):
    next_id = start
    while True:
        yield next_id
        next_id += step
        if next_id > max_id:
            next_id = start
```

- But running **pytest** gives **ImportError** for **pycagw.helpers**

Hunh?!



https://media2.fdncms.com/thecoast/imager/bundy-al-bundy/u/zoom/1088729/al_gif-magnum.jpg

- I used one of the two commonly recommended directory layouts:

```
pycagw/                (project)
  README.md
  doc/
  pycagw/              (package)
    helpers.py
    tests/
      test_helpers.py
```

- Google to the rescue: Add (empty) `__init__.py` files in each directory
- Running `pytest` discovers the tests and reports success!
- Shame on me: I actually knew about `__init__.py`...

Then I wanted to run it

- Wrote more tests. Made them pass.
- Wrote some basic entry point for running a gateway process.
- Not ready, yet. Just a smoke test.
- But running `python pycagw\cli.py` gives **ImportError** on `pycagw.cagw`
- Could be a syntax error in the file... but `pylint` doesn't complain.
- Let's double check: Start Python interpreter interactively (REPL)


```
>>> import pycagw.cli  
>>>
```

...What?! No ImportError???


Hunh?!




https://media2.fdncms.com/thecoast/imager/bundy-al-bundy/u/zoom/1088729/al_gif-magnum.jpg

- 
- Google to the rescue: Make it a **proper** package
 - **Install** the package in **development** mode
 - Run `python setup.py devel`
 - OK, now it starts up without ImportError
 - It breaks only shortly after that, but that's fine (only a smoke test)

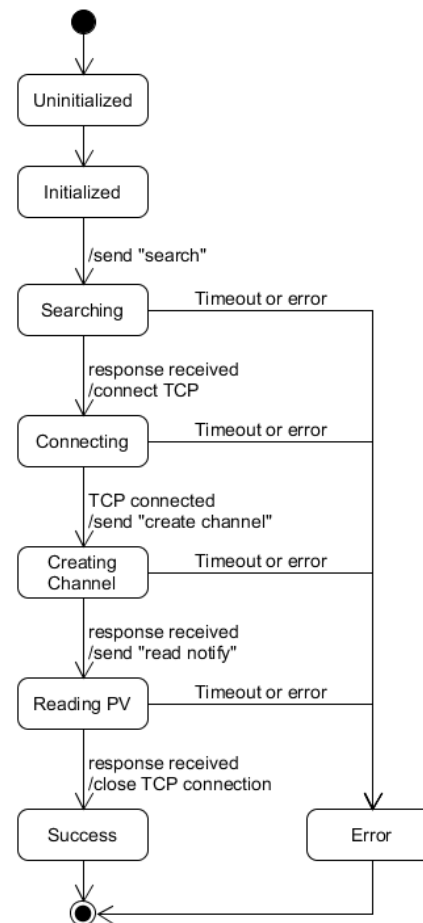
Things learned about Python

- 
- Python type hints (PEP 484)
 - Support IDE's autocomplete feature
 - Improve API documentation
 - No type checking at runtime
 - Generators (yield keyword)
 - Building a larger Python system is more involved than I thought
 - Directory structure
 - Make a proper package

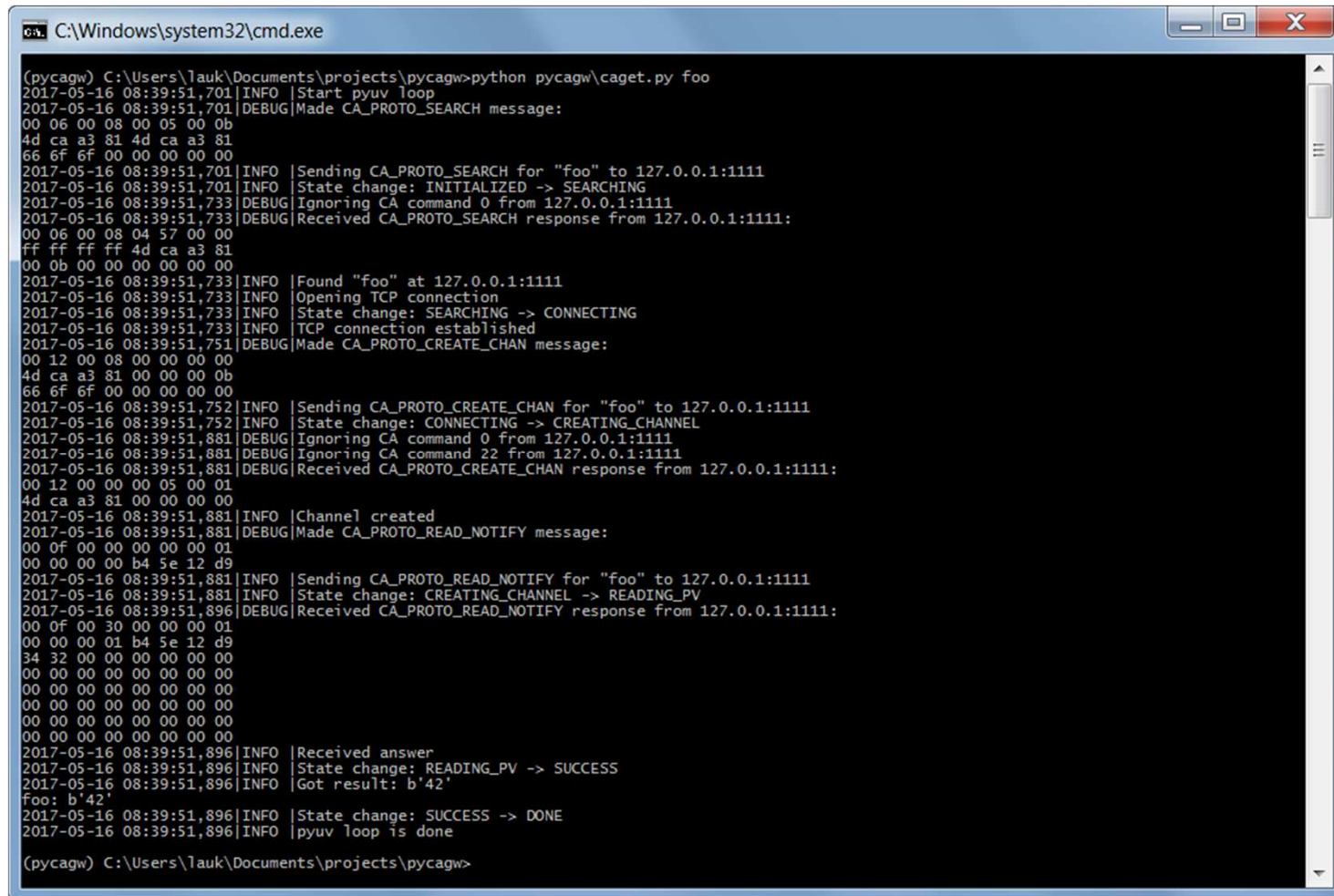
A lesson in humbleness

- 
- I know rather well...
 - How Channel Access works
 - Edge cases the gateway needs to cover
 - Reusing virtual circuits
 - Reusing channels
 - Another client request shows up, while gateway is processing one request
 - I tried to get the gateway to work right away.
 - I bumped my head for quite a while... and had to step back a bit:
 - Start with «only» a caget
 - Only 1 PV
 - Only 1 IOC
 - Only 1 virtual circuit
 - Only 1 channel

My own «caget»



My own «caget»



```

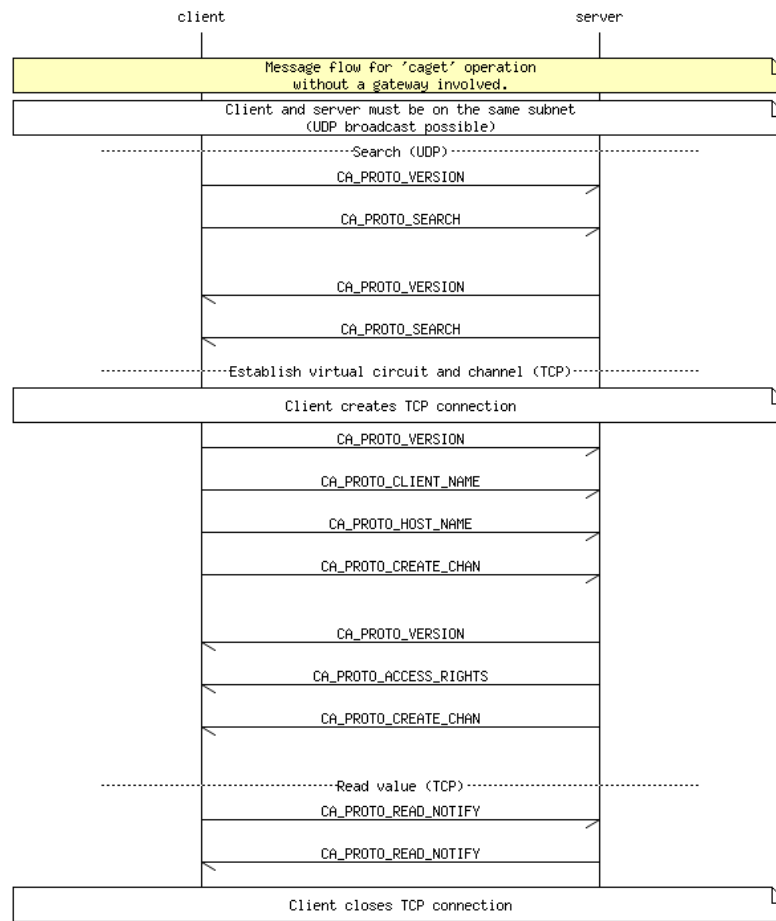
C:\Windows\system32\cmd.exe

(pycagw) C:\Users\lauk\Documents\projects\pycagw>python pycagw\caget.py foo
2017-05-16 08:39:51,701|INFO |Start pyuv loop
2017-05-16 08:39:51,701|DEBUG|Made CA_PROTO_SEARCH message:
00 06 00 08 00 05 00 0b
4d ca a3 81 4d ca a3 81
66 6f 6f 00 00 00 00 00
2017-05-16 08:39:51,701|INFO |Sending CA_PROTO_SEARCH for "foo" to 127.0.0.1:1111
2017-05-16 08:39:51,701|INFO |State change: INITIALIZED -> SEARCHING
2017-05-16 08:39:51,733|DEBUG|Ignoring CA command 0 from 127.0.0.1:1111
2017-05-16 08:39:51,733|DEBUG|Received CA_PROTO_SEARCH response from 127.0.0.1:1111:
00 06 00 08 04 57 00 00
ff ff ff ff 4d ca a3 81
00 0b 00 00 00 00 00 00
2017-05-16 08:39:51,733|INFO |Found "foo" at 127.0.0.1:1111
2017-05-16 08:39:51,733|INFO |Opening TCP connection
2017-05-16 08:39:51,733|INFO |State change: SEARCHING -> CONNECTING
2017-05-16 08:39:51,733|INFO |TCP connection established
2017-05-16 08:39:51,751|DEBUG|Made CA_PROTO_CREATE_CHAN message:
00 12 00 08 00 00 00 00
4d ca a3 81 00 00 00 0b
66 6f 6f 00 00 00 00 00
2017-05-16 08:39:51,752|INFO |Sending CA_PROTO_CREATE_CHAN for "foo" to 127.0.0.1:1111
2017-05-16 08:39:51,752|INFO |State change: CONNECTING -> CREATING_CHANNEL
2017-05-16 08:39:51,881|DEBUG|Ignoring CA command 0 from 127.0.0.1:1111
2017-05-16 08:39:51,881|DEBUG|Ignoring CA command 22 from 127.0.0.1:1111
2017-05-16 08:39:51,881|DEBUG|Received CA_PROTO_CREATE_CHAN response from 127.0.0.1:1111:
00 12 00 00 00 05 00 01
4d ca a3 81 00 00 00 00
2017-05-16 08:39:51,881|INFO |Channel created
2017-05-16 08:39:51,881|DEBUG|Made CA_PROTO_READ_NOTIFY message:
00 0f 00 00 00 00 00 01
00 00 00 00 b4 5e 12 d9
2017-05-16 08:39:51,881|INFO |Sending CA_PROTO_READ_NOTIFY for "foo" to 127.0.0.1:1111
2017-05-16 08:39:51,881|INFO |State change: CREATING_CHANNEL -> READING_PV
2017-05-16 08:39:51,896|DEBUG|Received CA_PROTO_READ_NOTIFY response from 127.0.0.1:1111:
00 0f 00 30 00 00 00 01
00 00 00 01 b4 5e 12 d9
34 32 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
2017-05-16 08:39:51,896|INFO |Received answer
2017-05-16 08:39:51,896|INFO |State change: READING_PV -> SUCCESS
2017-05-16 08:39:51,896|INFO |Got result: b'42'
foo: b'42'
2017-05-16 08:39:51,896|INFO |State change: SUCCESS -> DONE
2017-05-16 08:39:51,896|INFO |pyuv loop is done

(pycagw) C:\Users\lauk\Documents\projects\pycagw>

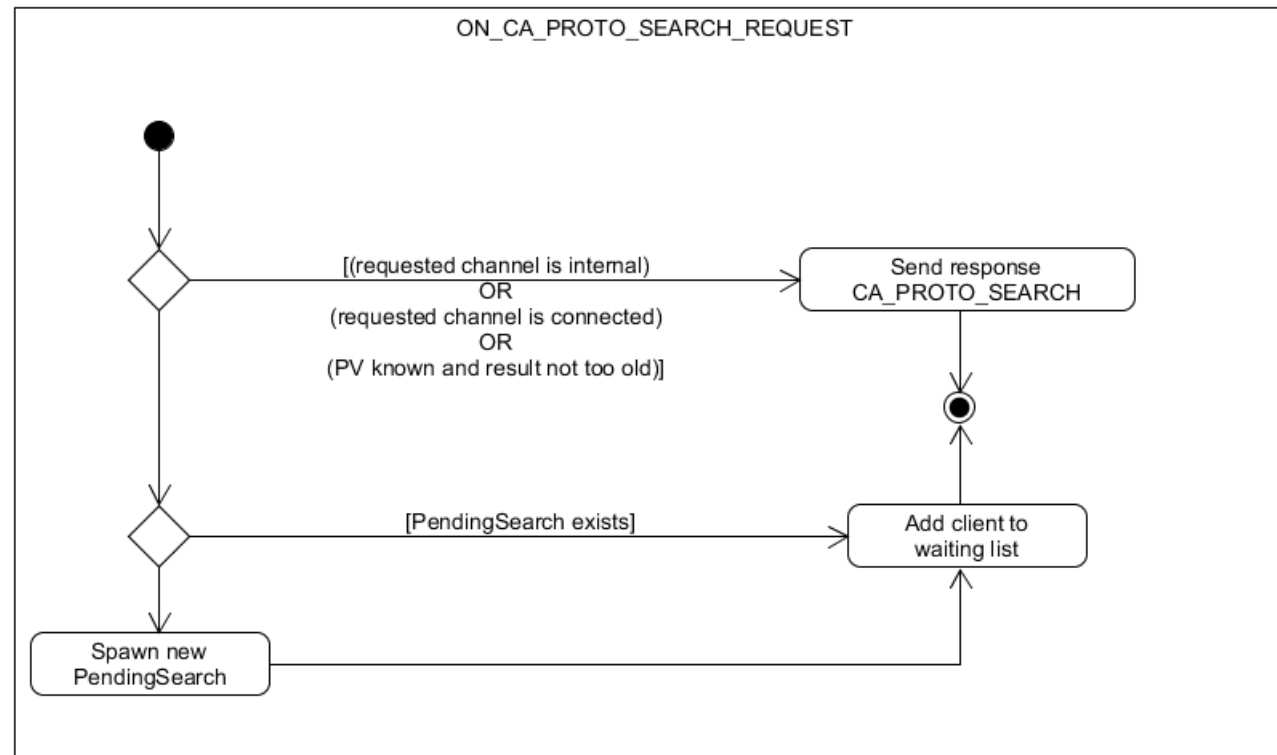
```

Divide and conquer

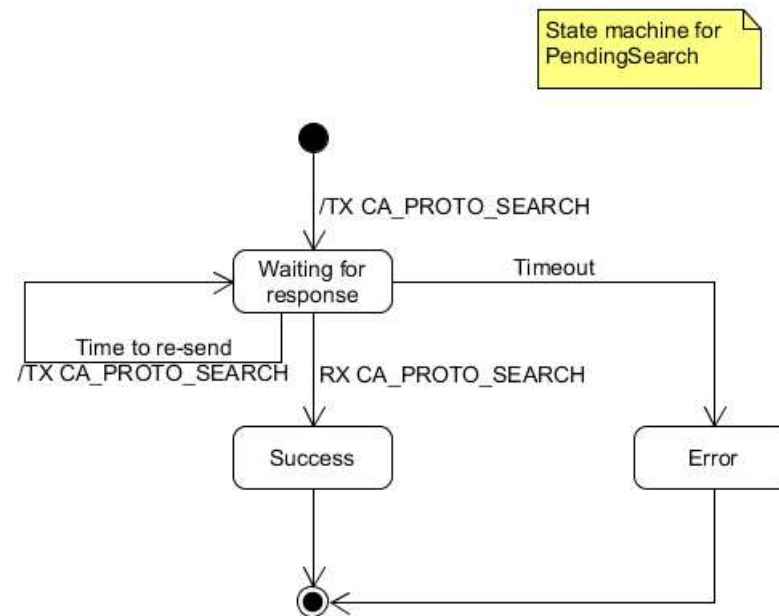


- Three (and a half) stages
 - Locate the PV (aka «search»)
 - (Create virtual circuit)
 - Create channel
 - Read PV

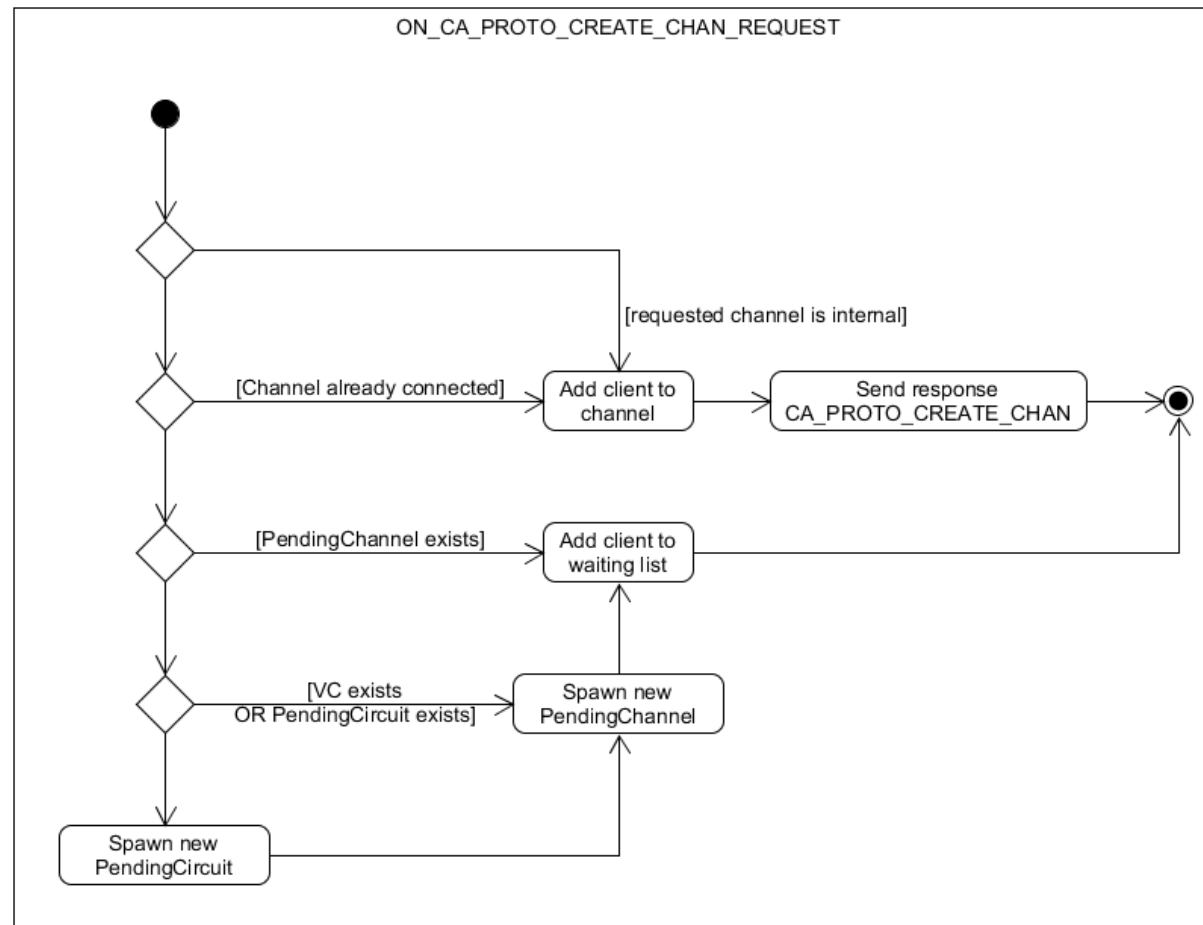
Activity Diagram: Handle search



State machine: Pending Search

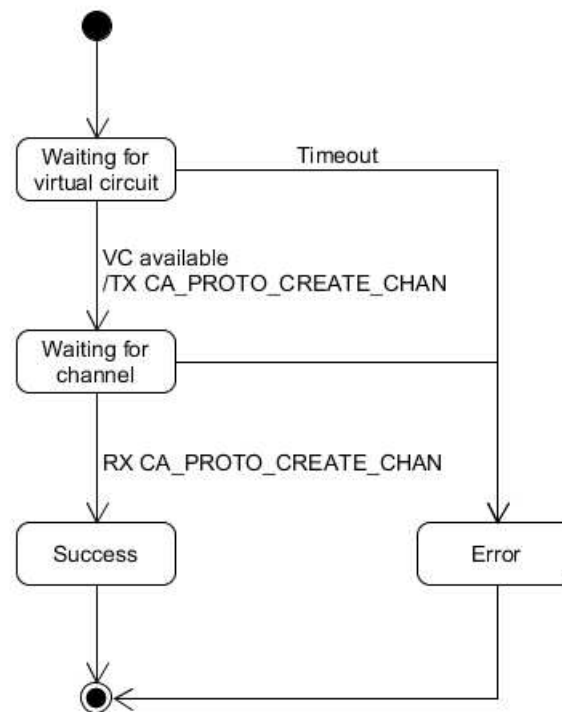


Activity Diagram: Create Channel

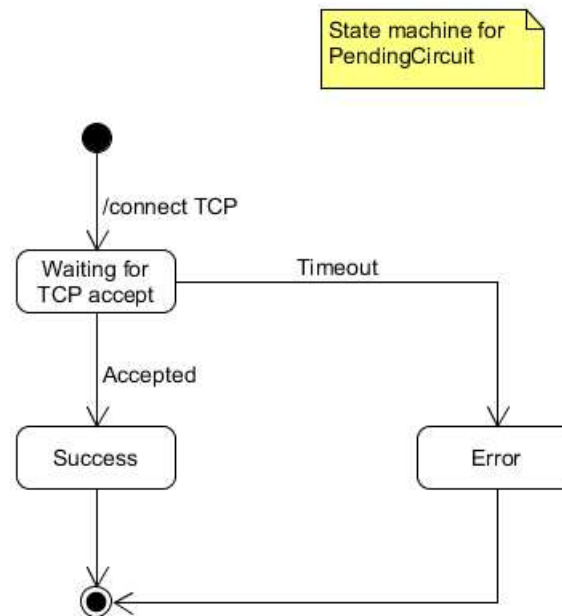


State machine: Pending Channel

State machine for
PendingChannel



State machine: Pending Circuit




Another lesson in humbleness


- My local development setup:
 - Local CA server (in C#) listening on 127.0.0.1:5064
 - Local gateway (in Python) listening on 127.0.0.1:1111 and 127.0.0.1:2222
- It all was working just fine the day before, but the next day...
 - caget.exe PCTOTO02:INT
 - Nothing
 - python pycagw\caget.py PCTOTO02:INT
 - Search time out
- Change EPICS_CA_ADDR_LIST to contact CA server directly
 - Still nothing
- Put breakpoint in Visual Studio in C# CA server library
 - Search is being received and processed

Hunh?!



https://media2.fdncms.com/thecoast/imager/bundy-al-bundy/u/zoom/1088729/al_gif-magnum.jpg

- 
- A solid grey square is positioned to the left of the first three bullet points.
- I was trying to read PCTOTO**0**2:INT
 - caget.exe PCTOTO2:INT works
 - python pycagw\caget.py PCTOTO2:INT works
-
- I should have copy&paste from the server code to the shell prompt
 - I changed the name of my server's PV to **foo**

- 
- RFC 1925 was right.
 - Pair programming and/or code review helps.
 - You need a good test setup for development
 - Setting breakpoints in all 3 components (server, client, gateway) helps
 - At least be able to adjust debug output levels (on server and client)
 - Using asynchronous I/O (pyuv)
 - Good: Single threaded → No locks / mutex / semaphores!
 - Bad: Callback hell!
 - Using Python
 - Good: The usual (readable, quick to write, dynamic)
 - Bad: Many of my mistakes were based on type mismatches from copy&paste or refactoring (strong typing and compile step would have helped)

**Thank you for your
attention!**

