ANSAR NETWORKING

Pushing back on complexity

THE CURRENT PROBLEM

The current default messaging technology (i.e. HTTP and all its different forms) is not fit for purpose – it is not a complete, modern solution to the full set of networking requirements. At best it takes care of a small set of requirements making it is quite useful in that space from browser to backend.

For a long time HTTP has been the technology underlying the RPC interactions between browser and server. More recently there have been initiatives towards asynchronous, bi-directional messaging, i.e. what developers actually need to deliver advanced features. This need is evidenced by the existence of WebSockets and the async options in FastAPI and gRPC. Together these items are encouraging but they bring their own problems like “calllback hell” and the fact that none of them go far enough. Asynchronous, bi-directional messaging is somewhat redundant when used to connect two non-asynchronous apps.

Aside from this fundamental flaw, HTTP-based messaging does nothing about service discovery, session management, zero-configuration networking or WAN level networking, e.g. IoT. Look inside any backend process and you will likely see a combination of messaging-related technologies used to meet the various requirements in those areas. This situation is exacerbated by the fact that some network messaging is performed by vendor-specific client libraries, e.g. for interfacing to PostgreSQL. On one hand these libraries can help to hide specific third-party details. On the other hand they introduce yet another proprietary approach to network messaging with its own type system and execution model, i.e. is it synchronous or is it asynchronous. Source code for backend processes can become ugly places.

Development of modern, distributed solutions is difficult due to the lack of a high quality messaging system that knows about asynchronous apps, session management, service discovery, zero-configuration and the full scope of networking (e.g. WAN). Symptoms that emerge in software projects will include;

- bugs associated with code complexity,
- poor responsiveness and processing delays,
- heavy resource footprint,
- poor scalability,
- reduced availability,
- low development velocity,
- inability to implement some advanced features and behaviours,
- or low quality implementations of features and behaviours.

BENEFITS OF ANSAR

Ansar offers an operational framework for the full set of networking requirements. It provides a basis for out-of-the-box, off-the-shelf, plug-and-play network software.

It pays attention to the needs of the smallest groups of processes (i.e. multi-processing on a single host), through to the largest groups with many processes spread around the Internet.

Immediate technical benefits include;

- platform suitable for advanced features, e.g. propagation of server-side events,
- built-in host, LAN and WAN messaging - optional encryption,
- drop-in compatibility, i.e. software components can be reused across multiple projects,
- simplified deployment, i.e. zero-configuration networking,
- side-by-side deployment – development, QA, training and sales instances on one network.

Longer term benefits;

- shorter time-to-market,
- composition of new solutions using existing components,
- responsive, efficient and scalable software
- fewer bugs,
- quicker maintanence cycles, e.g. bug fixes, new features and updates
- happier development teams

PRODUCT SUMMARY

There is no other messaging technology that addresses the full scope of operations, as described in previous paragraphs. As long as software projects continue to use custom combinations of disparate technologies, they will continue to experience the listed difficulties.

While there is unlikely to ever be a single technology that delivers on all the requirements of all software systems, Ansar is particularly well positioned to assist in the development of high quality backend software. Its asynchronous roots are a much better starting point for development of responsive, efficient and reliable software components.

Ansar is currently tested for operation across a wide range of Linux platforms including Ubuntu and Debian. There is a good chance that Ansar will run on any modern Linux platform that includes an up-to-date installation of Python and Internet connectivity. Testing has also covered the macOS platform and the Raspberry Pi.

Support for Windows operation is currently delayed by a single technical item – teardown of asynchronous sockets under Windows requires special handling. A few weeks of focused effort should add Windows to the list of supported platforms.

Ansar is a Python-only technology. Earlier versions of Ansar used C++ but the effort required to maintain build chains for all the combinations of compilers and target platforms became unsupportable.

COMMERCIAL PERSPECTIVE

Ansar is comprised of three integrated libraries and a supporting WAN service currently running under AWS. The three libraries are in the public domain (available on PyPI.org) under the MIT license. Access to the WAN service is currently free, allowing anyone to try out the built-in WAN capability with a few CLI commands. Git repos for the libraries and the WAN service are all on Github and are private.

There is substantial documentation for the three libraries;

- ansar-encode (serialization)
- ansar-create (asynchronous framework)
- ansar-connect (networking)

Commercial opportunities exist around the WAN service. This is similar to the situations around SQS and MQTT. These include;

- subscription-based access to the public WAN service,
- private deployments of the WAN service for performance or security reasons
- support and consulting