

# Python in Large Organisations

**Shaun Hills**

*NHS Connecting for Health*

shaun.hills@nhs.net

## **Abstract**

Python provides some flexible and powerful tools at low cost, and encourages clean and maintainable design. This can be a compelling value proposition and is recognised by many organisations. But on the other hand, Python's penetration of the "enterprise" space is not always high.

This document outlines the experience of the NHS Connecting for Health (NHS CFH) web team, deploying, running and supporting Python-based systems. NHS CFH is one part of the UK National Health Service. Collectively, the NHS is one of the largest organisations in the world.

## **1. Introduction**

As in many organisations, large parts of the NHS build their IT infrastructure on proprietary commercial systems such as Microsoft Windows. Within this context the NHS CFH web team have been using and supporting several F/OSS and Python-based systems for nearly five years.

### **1.1. Organisational background**

The team are not responsible for all IT – or even web – projects. Mainly they deliver information-based services (e.g. documents and contacts databases) to the wider NHS. Development resources are limited; a large amount of time is also spent on running existing sites, visual customisations and advising other teams.

### **1.2. Plone**

In late 2004 the team were supporting Microsoft ASP/IIS websites and were involved in a project to implement Microsoft's CMS product. Organisational changes required a new website to be delivered in a short time; Plone was trialled and subsequently implemented after the trial went well.

Plone is currently used for content-related tasks. Zope is also used to host some static non-Plone content.

### **1.3. Django**

The team inherited several legacy ASP/MSSQL applications, and also carry out new development. Some new apps were initially written in PHP, this was due partly to a lack of familiarity with Python outside Plone skinning. It was also due to the (perceived or actual) complexity of carrying out greenfield development in Zope/Plone, especially when working with relational data, and the fragmented nature of Python web frameworks at the time.

Django is currently used for application development, especially where the data is suited to a relational storage model. Legacy applications are gradually being moved to Django.

### **1.4. Related technologies**

Systems are hosted on Linux, hosted on VMWare ESX virtual machines. Plone search is replaced by Google Mini appliances. All systems are proxied via Apache, so a unified Plone/Django front end is presented to the user. As yet there is no compelling reason to move to WSGI; HTTP proxying works and is well understood. The team are investigating Deliverance to further unify the user front end.

## **2. Challenges**

Finding Python developers (and particularly developers familiar with Plone) has been and continues to be difficult.

This is exacerbated by the fact that many Python consultancies are small businesses. There are overheads associated with projects in large organisations, which these businesses aren't always well set up to address. For example procurement and payment timelines can be very long, system changes can take a long time to approve, and project management and reporting requirements can be onerous.

The systems (notably Plone) can be a technical “moving target”. This has implications for support, planning and staff training. Many developers are interested in optimising the technology, but users/integrators may be interested in getting something deployed and not having to re-architect it every 18 months.

Organisations tend to be risk-averse. Partly related to the above points, partly because people will go with what they know, and partly due to precedent (“No one ever got fired for buying IBM”).

## **3. Benefits**

Python has technical merits such as the very large standard library, the lines-of-code productivity of a dynamic language, and its encouragement of clean and consistent

code. These are important to programmers, but they're also important to integrators, project managers and support staff.

The cross-platform nature of CPython is important. Several NHS Plone sites have been developed simply because it can run on Windows. The emergence of Jython and IronPython is also likely to become important because they allow Python to slot directly into enterprise infrastructure.

A large amount of functionality is available in Python-based systems; though not always well promoted or obvious. For example Plone has inbuilt workflow, granular security and desktop integration. Django allows developers to quickly build applications in a business-logic-centric (i.e. model-driven) way.

Python is still a relatively niche technology, but it's less niche than it used to be. "Google uses it" is a powerful precedent. UK recruitment consultants have recently started head hunting for Django developers.

One advantage of niche status is that it tends to attract enthusiasts, interested in the technology for its own sake. This helps with delivering quality. Also technology enthusiasts in other areas (e.g. PHP and Linux) can often be easily cross-trained.

#### **4. Strategies**

Focus on the business requirements and not on the technology. Delivery on requirements is how a successful project is measured. In several NHS CFH projects the commissioning team have not been aware of the technology used, and in some cases have assumed it to be ASP.NET or SharePoint.

Consider Python technologies on their own merits, rather than as a niche "alternative". For some problems Python provides very competitive solutions. And large organisations produce many opportunities. Use Python where it's compelling, don't try to force it where it's not, and in the middle don't waste too much time trying to push it against other technologies that are just as good for that particular problem.

Contrary to common belief, F/OSS systems often don't have a compelling one-off cost advantage. Over the lifetime of a project, the license cost may be an insignificant percentage of total cost, especially if the F/OSS system requires retraining or additional customisation. However the marginal return on investment in the open system is higher. Additional deployments and work will be very low cost.

Make sure the "soft" factors are covered. Communicate frequently and well. Consider budgeting time and resources for project and stakeholder management.