Blesséd Code: How Christian ministry might be expressed through software development – James Handley

In the world of the SSM and MSE, it seems all too easy to compartmentalise our activities. I am an ordained minister in the Church of England, and I am also a professional software developer, and these two spheres do not have a lot of overlap. This article outlines a possible aspect of how the diaconal and priestly ministry to which I was ordained might be expressed through my work as a computer programmer (and by extension to other professions.)

By way of background, my job involves writing human readable text ("source code") which a computer translates ("compiles") into machine code, which it can run ("execute"). So for example the code:

```
a = a + 1
```

would be compiled into a set of instructions which will take the number stored in `a` (e.g. 5), add one to it (i.e. make it 6), and store the result back in `a`. Exactly the same result could be achieved by the following, much harder to understand, code:

```
a = (100 / 10) + a - (√1) - 2^3
```

The point is that — as with any language — there are multiple ways to express an intention or notion, and hence opportunities for simplicity or obfuscation. This turns out to be crucial, as we will see later.

Ministry at work

There are several options for addressing the question of being an ordained minister in a secular workplace. Formal chaplaincy is one, although in my case ministry is neither my job nor my function. To use my time at work for ministerial activities when I am being paid to develop software would not only be inappropriate, but dishonest [1]. I am no more paid to proselytize or offer pastoral care than I am to browse Instagram or play Candy Crush.[1]

At the other extreme, I could principally express Christian ministry through activities at church, while generally witnessing at work and perhaps seeking evangelistic opportunities. Somewhere in the middle, there is the path of the MSE – see Lees [2] and Vaughan [3], amongst others. In that model, the language of presence is key — we "are" in the workplace, and that in itself is a ministry.

Alternatively, one could seek "sacred" work; jobs which could be considered to align closely with the gospel agenda — like running a foodbank, or being a doctor or a teacher. One can also fairly easily

[1] There is some nuance here. Providing pastoral support for a colleague may improve their performance, and hence still generate a return for the business.

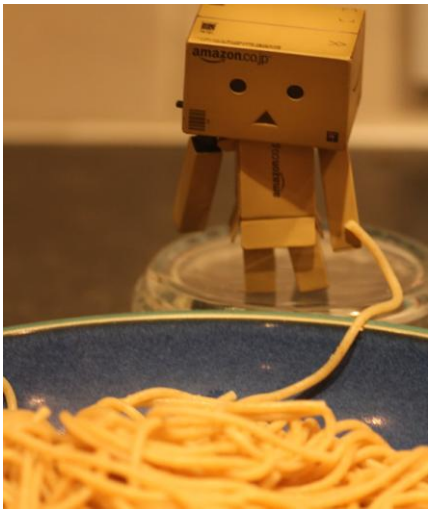
align “servant” roles like judge, policeman, politician,[2] waste disposal, gardener. Even when the work is not inherently “sacred”, the product/result might be. In my field, work-as-ministry could be writing worship software or a church management suite, or even producing something like pornography filters or online grooming detection.

But what of the Formula 1 mechanic, or the investment banker, or the shop assistant? With a bit of theological footwork you can talk about the value it adds to society, such as generating wealth,[3] or providing entertainment, or indeed simply our God given mandate to work. In my specific case, the software I develop can help reduce carbon footprint through operational efficiency gains, which aligns with our stewardship of creation.

However I find myself not fully satisfied with this “generic” priesthood, where the work I specifically do doesn’t really seem to matter too much.

Blessing and curse

At my work, we have a positive organisational culture — high levels of transparency, trust, encouragement, cooperation, and low levels of passive aggression, sabotage, negativity, secrecy, politicking, and so on. However, a fellow software developer at work recently mused that ‘If you consider the culture of the code, then that’s a different story — it can be negative, opaque, it’s passive-aggressive, …’



This notion was something of a revelation to me. He is absolutely right though — source code does have a “culture”, and I don’t just mean the resulting piece of software. As we saw in the introduction there are choices around how code is written, and hence what culture is embedded within it. Source code can be helpful and co-operative; easy to understand and maintain, so you can have confidence that changing one bit here isn’t going to break something over there. Or it can be obstructive and difficult. A spaghetti tangle, where it’s almost impossible to understand the logic, and you can be pretty sure that changing one bit is going to have a completely unexpected side effect in an apparently unrelated part of the system.

It is nothing new to suggest that code can be good or bad — but if you see it as embodying a culture, then you could say it becomes a potential means of blessing or a means of curse. Easy to work with, or difficult and obstructive. The language of blessing opens up a door to start to draw in priestly

[2] Perhaps slightly more challenging in the current climate!
[3] Which, so the theory goes, helps everyone in society?

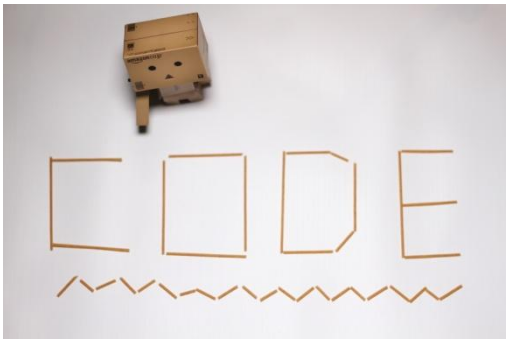
ministry. Jim Francis helpfully draws out of the Ordinal three distinct strands for priestly ministry: “bless”, “reconcile”, and “nurture” [4]. Might blessing, reconciling, and nurturing also apply to software development?

Whose code is it anyway?

Another colleague pointed out that the code I write is not “my” code. I don’t own it — it is the company’s Intellectual Property, and it will probably have a lifespan beyond my employment there. It is likely that it will not even be me who is the next person to work with it. From this, I realised that a principal way that I interact with my colleagues (and indeed our customers) is through the code I write. If I write code badly, this causes pain for my immediate colleagues, as well as those to come. On the other hand, if I write code well, this can be a source of blessing to my colleagues. They can pick up what I have written; easily understand what it is doing, and why I wrote it that way, and make whatever modifications they may need to.

It is at this point all these thoughts collide — that part of my calling is to be a blessing; that computer code has a culture; and that one of the main ways I interact with my colleagues is through code…

Blesséd code



Might it be that I can exercise a priestly ministry at work by writing blesséd code? I don’t mean consecrated, I mean code that it is a means of blessing. I mean my work bringing pleasure to others; beautiful, well crafted, and elegant code that is a blessing to work with. Conversely, code could be “cursed” — again not hexed, but rather a curse to work with, bringing pain and torment. I have certainly seen enough of that code in my career. But even “cursed” code can usually be “saved”. It can be re-written or refactored into blesséd code — and might this in turn be a picture of reconciliation and redemption? If we are working to create a blesséd environment within which to be joyful and fruitful in our labour, that is something of the Kingdom of God, surely? It’s like turning a scrapyard full of stinging nettles and old tyres into a garden or allotment; bringing order out of chaos and life out of death. I’m not sure that it’s any different for code.[4]

Closing Thoughts

[4] In conversation with the Bishop of Ripon, she highlighted the re-wilding movement, and that our neat and ordered gardens do not necessarily fully reflect God’s glory in creation. This does beg the question of what “re-wilded” software might look like?

Let us first be clear that every Christian — whether employed or not, whether ordained or not — should be asking themselves these sorts of questions. Another way of expressing it might be ‘In what ways am I participating in the Missio Dei?’ However, I do think that being ordained and in secular work brings it into sharper focus; similar to the way the ministerial priesthood brings into focus the royal priesthood of the whole church.

Secondly, there is a danger of being over spiritual, and that what I’m describing is nothing more than doing a good job. However if our output at work forms part of our relationship with those with whom we work, then I think it is potentially fruitful to at least reflect on it in terms of ministry.

Thirdly, while we have been concentrating on source code, the end result (i.e. the piece of software itself) can also be blesséd — a joy and blessing to use — or the opposite. Blesséd software in this sense is also potentially a ministry to both our customers, and our technical support team.

Finally, this doesn’t only apply to software. If you write protocols or instructions, which have to be understood and followed by others in the organisation, the same principle applies. If you run the IT network, or if you’re in charge of the laundry — all of these things, and potentially many more, can be the basis of a ministry to colleagues, a means of blessing and of advancing the Kingdom of God.

References

[1] Antony Hurst. Rendering unto Caesar. An exploration of the place of paid employment within the framework of Christian belief. Churchman Publishing Limited, Worthing, 1986.

[2] John Lees. Self Supporting Ministry. A Practical Guide. SPCK, London, 2018.

[3] Patrick Vaughan. Speaking for themselves. In Peter Baelz and William Jacob, editors, Ministers of the Kingdom. Exploration in Non-Stipendiary Ministry. CIO Publishing, London, 1985.

[4] James Francis. A reflection on MSE. Ministers-at-Work, (152):26–32, January 2020.