BENNETT INSTITUTE FOR APPLIED DATA SCIENCE

A National Data Library: the Modular Approach

Ben Goldacre, Seb Bacon, Pete Stokes, and team
Bennett Institute for Applied Data Science,
Nuffield Dept of Primary Care Health Sciences,
University of Oxford

This 6,000 word paper is an entry to the Wellcome / ESRC
"UK National Data Library Technical White Paper Challenge"

December 2024

Who we are

We are a large team of public servants delivering high throughput data infrastructure. BG is a clinical professor and Director of the Bennett Institute. SB is CTO with 30 years experience leading major software and data projects. PS is Director of Platform Development; he previously established the highly productive Secure Research Service at ONS. Our team comprises 60 software developers and researchers, pooling skills and knowledge: we develop new working methods for data; then implement those ideas in working data services at scale. We built OpenSAFELY, the national networked Trusted Research Environment for whole population NHS GP data. Our platform overcame privacy challenges to earn the trust of all stakeholders, and therefore provisions an unprecedented scale of data: the full detailed primary care records of all citizens in England. Our platform overcame data management complexities to become highly productive, with a huge volume of outputs on 181 projects from users at 31 different organisations.

The Vision: a modular approach

Previous attempts at delivering national data infrastructure have relied - unwisely - on moving all the data to a single box, and giving all delivery work to huge single teams managed within large single organisations. That model is bad for privacy, scalability, accountability, and delivery. Those projects have gone badly.

Here we set out a new model: a network of single function *components*, stitched together into a working data service by the National Data Library.

This model is better because it:

- Follows best practice in delivery of large data and digital projects;
- Ensures clarity, transparency, and accountability for each component and team;
- Drives best use of existing tools and teams;
- Leaves the best legacy for future work;
- Ensures scalability;
- Facilitates resilience;
- Lets you pick the best team for each component.

This paper summarises: the context; the reasons why recent government data platforms have failed; and the best technical design of an NDL, including the components and the overarching network. It then sets out a series of organisational approaches to help government define the cultures and market incentives to drive success. It ends with a concrete proposal for a small project to illustrate this way of working.

The context: opportunities, challenges, and failures

The Opportunity
Data can supercharge research, government and the economy. Better, broader use of the data *we already have* represents the best return on investment of anything government can do. We should all be optimistic, because the return on investment is phenomenal.

The Challenge: data is like nuclear material, not oil
The challenge is implementation: how to give large numbers of users more access to more data, whilst protecting citizens' privacy, and avoiding another failed government IT project. Data is not like oil: it's more like nuclear material. Personal data has huge potential to do good; but also intrinsic risk. Small amounts of data achieve little, and pose few risks. Data becomes powerful only when brought together, and refined: then it also becomes more dangerous. Aggregating and refining data requires teams with deep, practical, creative, technical skills. Data needs secure environments, because if data leaks, it can't be unleaked. Today - sending data around to multiple locations, chaotically, for one-off projects that are often trivial - we sometimes treat national data like we treated nuclear material in the early days of the atomic age: enthusiastic amateurs, in unsafe conditions, painting glowing material on their teeth, in a clock factory.

The Failures
There is a crisis of delivery on data platforms. **Many large, recent and ongoing government / NHS data platform projects have failed**, despite substantial public investment. This must be accepted, and discussed, but with politeness and empathy. It is not appropriate to list specific failures here.

There is one critical action now: **government should execute a rapid, empathic and simple no-fault audit** of current data platform spend. For speed this should collect only: a list of the data platforms funded; who paid; how much; who was delivering; and a list of completed outputs delivered by their users. This was attempted by an independent organisation here, but - concerningly - many NHS / govt organisations responded incompletely. That survey includes >£100m national data platforms that report minimal outputs to date. Scale doesn't justify slow delivery: effective teams engage in Continuous Delivery, with outputs from the beginning. From our informal knowledge, there is nothing re-usable in any current unproductive platform to justify delay in *new* data platform delivery.

Why did these projects fail?
All had superficially credible plans, pleasing leaders and ministers. All were well-reviewed at inception. All had some technically skilled people. All achieved positive comms and reviews throughout their lifecycle; all while failing. There are recurring systemic themes driving non-delivery that can be acted on if they are understood, accepted, discussed, and mitigated.

Common failure modes include:

- **Single monolithic projects** where all components of the job are given to one huge team, so that sight of non-delivery in any individual component is obstructed.
- Monolithic projects becoming politically powerful and **too big to fail**.
- **Lack of accountability** and oversight, leading to **bad incentives**.
- **Lack of technical skills** and knowledge **in senior leadership**.
- Giving the task to teams with **technical skills but in adjacent domains**, like statistics or research, rather than platform delivery.
- Giving the task to **large unfocused public sector organisations with multiple other tasks**, who are then: distracted by other tasks; able to divert attention on failing

platforms by delivering on other tasks; tempted to **divert funds to other work** in their organisation; prioritise their own requirements in the new platform over other users' needs.
- **Wishing away the core problem of "data management"**, the preparation of raw data into analysis-ready datasets.

We discuss how best to address the culture of technical delivery in the second half of this document. First, we address the technical design principles.

How To Do It: the technical data architecture

Our two core recommendations are simple. Firstly: build separate, connected data centres, not one giant database. Secondly: build connected single purpose services, stitched together into a national data platform, not one monolithic organisation delivering all the work. We assume only modest initial investment, with potential to scale after success.

Don't build one single huge database

The default design principle from all previous government data projects has been to try to put *all* the data about all citizens in one big box, then let analysts log in to use it there, in whatever way they wish. This makes superficial sense: "my team needs tax + health + schools data in one analysis, so we need all the data in one machine". In reality this aggregation is unnecessary: it also creates huge problems for privacy, and obstructs delivery.

Firstly, **data lakes of this kind are terrible for privacy**. Putting all data on all citizens in one single database, then giving access to many users, will create unprecedented privacy risks. This will catastrophically undermine public and professionals' trust in government use of data. It will - intermittently - create huge, avoidable privacy catastrophes, harming politicians and leaders. Cautious access rules cannot compensate for this bad technical design. Related to this, **data lakes are bad for transparency and audit**: because users simply log in and work directly with huge volumes of disorganised raw data, it is hard to track - or report - what they are doing with that data.

Secondly, **data lakes are bad for data management**. The *preparation* of raw data represents most of the labour in an analysis: a big messy black box containing all the data deteriorates to anarchy, as different teams will always do similar work in different ways, unless huge effort is exerted.

Thirdly, **data lakes create conflict between institutions**. To take an imaginary example (informed by real live disputes): the XYZ team have worked for years to create a complex national database on every citizen's tax/school/pension/etc. They don't want to hand all "their" data to a national data lake; they worry about losing control or sight of the uses; that users will misunderstand the XYZ data they love, or do misleading analyses; they worry that bad analyses will affect the XYZ team's reputation; that others will take credit for XYZ's work; or get privileged access to do analyses first. These human reasons drive failure: some should be fought back; some warrant serious thought.

Do build a network of connected data centres

A better model is a federated model. The complete, raw data in each data centre or department *stays put* in that source data centre (either physically, or "logically partitioned"). Each data centre has a data preparation service: tools, a team, and good user-oriented data documentation. For each project, users follow a "**take only what you need**" approach. Instead of extracting *all* the data, they take from each data centre only the pre-prepared information they need, about each citizen, for each analysis. This is sent to a third location for linked analysis; then archived; then destroyed. This federated model has huge advantages over data lakes.

- **It mitigates the privacy challenges** of data lakes, because the detail in the data is minimised at source, while still letting the user take precisely what they need.
- **It makes single teams clearly responsible for single tasks**, with clear goals and responsibilities.
- **It allows for good audit, with pragmatic granularity**: it's easy to be transparent about what data goes where, and for what; data centres can log the data taken, and the project identifier, without needing code-level logs of *all* actions on all data.
- **It lets departments exert some granular control**: they can have project-level approval (with bulk data exports, they risk losing control forever).
- **It reduces single points of failure**.
- **It keeps the data close to the people who know it well**, sharing expertise in formal structured ways, more than ad hoc conversations.
- **It preserves normal working models**: raw data always requires pre-preparation into analysis-ready forms; the only difference here is that preparation is done in situ.

Critically, **exceptions to this working model can be managed**. It is common to hear from people that an efficient, secure, federated model of this kind will not work for the occasional specific analysis, or some kinds of AI approach. This is fine. A tiny number of projects may, indeed, legitimately require more raw data in one place. Those can be treated as exceptions: their higher risks managed thoughtfully; their higher costs understood and passed on appropriately; their delays accepted, for those users only. These occasional exceptional projects cannot justify lower standards on privacy, efficiency, speed, standardisation, and transparency for all work by all users across all datasets.

Don't give all the tasks to one huge monolithic delivery organisation

The historic norm for major data projects is: give all money and all tasks to one closed group, inside or outside government; that organisation then delivers the overarching design, and all components in the service. This can feel "simple" "clean" or "decisive", especially for senior decision-makers who want one accountable individual.

In reality, the "monolith" approach undermines delivery and accountability, because that monolith is empowered, incentivised and resourced to create a black box around all design choices, and progress. Only brief comms material is shared externally, because nobody else "needs" detailed information on individual teams and services within the black box.

This has huge negative consequences.

- Nobody outside the monolith can see the work.
- Problems stay hidden for longer.
- Outside teams - with skills and knowledge - can't offer help or feedback.
- Teams cannot be held accountable to delivery by anyone outside the monolith.

Do build a network of standalone services, stitched together into a platform

A better model is to have a thoughtfully designed network of single standalone component services, each responsible for one task. This model is flexible, and extendible. Each service interacts with others via formal computational interfaces (APIs, containers, or similar) rather than meetings and personal relationships alone; or, for more "human" services (like bespoke dataset creation) via SLAs, forms, etc.

This model brings huge benefits:

- Single standalone component services can fail without bringing all work to a halt for everyone.
- Standalone services are accountable: a service either works, when called by other services, or it does not.
- It widens the pool of talent and ideas beyond one single closed budget-holding organisation.
- Services can scale: bottlenecks are addressed with modifications, or duplication.
- Services can compete, driving improvement.
- Component services work naturally in the open (where the experts are): their expectations and outputs must be clearly visible to all services that interact with them… and therefore to everyone.
- Single services are easier for teams to build, because the requirements are clearly specified (rather than "create a machine that does everything for all users of all government data").
- Teams can be internally agile, independent and creative about how they meet their concrete external requirements.
- Teams can upgrade at their own pace, and add new capabilities where needed, without forcing system-wide changes.

Examples of data platforms made from re-usable services

To bring this model alive we provide two brief descriptions of end-to-end data platforms, made by stitching together single standalone re-usable services. These are high-level, indicative illustrations, not complete technical blueprints, due to word limits. Under Continuous Delivery this would change over time. Note: we make no comment on detailed information governance issues, and presume nothing here; this is a speculative white paper on the best technical models to preserve privacy and usability, for discussion and later implementation.

Example 1: a data platform delivering linked health and education data

Previous attempts to link data from these two domains have been ad hoc one-off linkages, for a single analysis; or monolith projects, where all the data is sent to one machine in one organisation, creating huge challenges around privacy and non-delivery.

Here we set out a component services approach. We assume that the aim is: provide a scalable platform to provision National Pupil Database and GP data (the highest value datasets from each domain); in a usable form; in a usable analysis environment; for many researchers and government analysts; and provide generalisable approaches that can work for other education and health datasets later. We are happy to provide more detail.

Data preparation services for NPD

NPD data is complex. There is a DfE team that knows it well: collecting it, and analysing it internally. This team, building on a mature service, with outside help as needed, would produce three services:

Bespoke Data Preparation Service
This team can meet with prospective NPD users, discuss their aims, and produce a derivative of NPD on their behalf. For example, a user might want - from the hugely detailed NPD dataset - a derived variable created for each pupil that says “More Than 5 GCSEs, yes/no”. The NPD team know this task well. They assist users with low NPD knowledge; and produce unusual bespoke data cuts not met by the service below.

DIY Data Preparation Service
This team maintains self-service tools that receive incoming requests - as code - for NPD-derived datasets to be prepared. They can meet some but not all dataset requests, focusing initially on the commonest NPD preparation tasks. They *only* prepare the data, to send to an analysis Service (below).

Data Documentation Service
This team maintains detailed technical documentation on the raw data including: how it’s collected; the data types; existing data quality reports; common usages; commonly used derivative datasets; the code used to produce them.

All three preparation services will likely have internal utility for DfE’s own analysts.

Data preparation services for GP data

Data Documentation Service
As above, but for GP data.

Bespoke Data Preparation Service
As above, for GP data.

DIY Data Preparation Service
As above, for GP data. (Note this already exists in OpenSAFELY ehrQL tools).

Data linkage services
The next challenge is to link information about each citizen from NPD onto the data for the same citizen from GP. For security and delivery reasons, this works best as a standalone function, receiving the minimum data needed for the linkage task, rather than the full record from each source data centre.

Probabilistic record linkage service
This service uses names, addresses, etc to match people in different data sources; other data from source records can be used cautiously to validate matching quality, and produce matching-quality reports.

Deterministic linkage service
Where unique identifiers exist in both settings, and mappings exist between identifiers, this standalone service provides a match and returns a new pseudo-identifer. These services already exist in government / NHS, but buried inside monoliths, with unclear performance: they should be unleashed.

Data analysis settings
Once the data is prepared and minimised, and linkage keys provided, the minimum data required from each data centre is brought together, linked, and provisioned to analysts. Different analysts have different skills and needs.

A "remote desktop" service
Pre-prepared and minimised datasets from multiple departments' data centres are called in, then provisioned to users in a remote desktop, where they can login and work on it. This kind of environment is easier for less experienced analysts, but provides weaker security safeguards than some other approaches: therefore data provisioned here should have additional privacy mitigations, e.g. random sub-sampling of the population, and more stringent minimisation during data preparation.

A "remote code execution" service
Pre-prepared and minimised datasets from multiple departments' data centres are called in, then provisioned to users in a remote / blind / automated code execution service. This kind of environment provides high security safeguards, therefore data provisioned here can be more comprehensive, and support more users.

A data analysis setting might be co-located with a data preparation service, to maximise privacy and minimise data flows.

Administrative services

An applications service
This service acts on application requests for the platform within constraints for priorities, resources etc. It does nothing else.

A shared "output checking service"
Analysts can view outputs (tables, graphs, logs) inside a Trusted Research Environment. When they want to publish those, to a wider readership, universal best practice states that those outputs should be manually reviewed, by trained staff, to ensure nothing disclosive leaves. This should be a general shared service, not tied to one analysis environment (except where domain knowledge is critical): it benefits from economies of scale; is sometimes done poorly, in platforms motivated more by throughput than privacy; and to needlessly varying standards.

A design, orchestration, and resourcing service
This team is responsible for designing a network of re-usable component services that comprise a data platform: understanding users' needs; identifying the best existing services to comprise a platform; identifying any new services required; resourcing these services; ensuring the platform runs end-to-end. In their design they are expected to maximise: efficiency; automation; re-usability; auditability; privacy; reproducibility. This is a *critical* function, and must be carried out by people with *outstanding* proven delivery records. The components must be selected as truly useful, truly separable, and orchestrated effectively.

This model in practice

"Describe a user's journey"
An applicant receives permission for a project. They read the documentation. They write one repository of code that can execute on the NPD and GP DIY Data Preparation Services. The minimised prepared data is linked on pseudo-ID from the chosen linkage service, then flows to a remote desktop service. The user logs in, does some more data preparation, then their analysis. They prepare their outputs for publication, submit them for output checking, and leave. Their code is archived; their data is archived, and later auto-deleted.

"Tell me about re-use and why this is good"
All these components can be re-used elsewhere, as set out in Example 2 below.

"What about competition and scale?"
If there was a shortage of output checking resource, a new team could offer that service in parallel, across all settings; or explore semi-automation.

"What about delivery?"
A remote desktop environment for data analysis is a common, simple service that *many* government / NHS projects have struggled to deliver, partly (in our experienced view) because those teams were all also trying to deliver *everything else in the pipeline*. Focus enables delivery.

Example 2: a data platform delivering linked health and tax data

Many people would like to evaluate the impact of poor health (or health interventions) on economic activity. This is currently hard: tax and health data are each regarded, rightly, by those who hold them, as highly sensitive.

"What are the components for this platform?"
The components are all the same as Example 1, but the NPD data preparation service is replaced by an HMRC data preparation service. This shows the value of a modular approach.

"Why is this better for privacy, or control?"
As an example, the analyst might only want to know, for each citizen, "are they working at all, yes/no", and nothing more. This is sparse, minimised information. HMRC (or, plausibly, DWP, whose data can also inform this question) decide (in this *imaginary example*) to support the project because of its value, and the privacy safeguards in the minimised data flow and remote code execution environment. HMRC is - exceptionally - permitted an output review, prior to release of final outputs, as goodwill. The data flows. The analysis is done.

"What about commercial data?"
The same approaches set out above can also be applied to commercially held data about individuals, and more: minimise data at source; and maximise modularity in the system.

In summary
Standalone component services are the key to success. By breaking down the NDL into smaller pieces we deliver value to users, quicker; reduce development and operational risks; and learn / adapt based on actual usage.

How To Do It: delivery approaches

Here we set out approaches and strategic advice that will help drive successful delivery.

DELIVERY APPROACH

Get users, using things, early
Failing IT projects deliver elaborate systems before checking they're needed. Get real users fast: give them early versions; watch users working; throw away ideas they don't need. In OpenSAFELY, we went from initial concept to first published research in 42 days. Some services were still manual, behind the scenes; but we proved our core concept by delivering real scientific insights. Early delivery of real completed user stories reveals real-world challenges, and validates core assumptions, through delivery rather than plans. It draws in collaborators, allies, and helpful critics.

"Wizard of Oz it"
Don't automate everything immediately. Like the Wizard of Oz, behind the curtain: have people, doing things manually, at first, for some services. The *final* version of an NPD Data Preparation Service might give users a self-service data preparation tool; initially, a small team might do that work manually; but perhaps (if informative) responding to API calls, via the API, as if they were that API. Then you deliver value quickly, while learning what needs automating, how, and how urgently; and test your assumptions about how the network of services will work together. You might find some disaggregated microservices actually work better rolled up into a larger service, or vice-versa.

Build a walking skeleton
Build the simplest version of the system that connects all its essential parts, to test all your core assumptions, early, cheaply, without drowning in complexity.

Scrapheap Challenge: re-use what exists today
Develop and adapt your plans to re-use existing tools, teams and datasets, while still delivering value. The sweet spot between ambition and pragmatism is full of easy wins.

STRATEGIC DATA CHOICES

Start with the "Top 3 Datasets" in each domain; build omnipotent systems later
Teams often want to provision *lots* of datasets, quickly, producing slide decks for ministers with "big numbers". But the outputs from commonly-used datasets can be phenomenal, and help you learn about delivery. Start with the "Top 3 Datasets" that researchers actually want. In healthcare: GP records; HES (every hospital visit); and death certificate data. In education: the NPD. Build good services and processes for those core datasets. Build capability to handle more complexity later. Where feasible, make design choices that generalise to other datasets later: but don't get paralysed.

Beware "dataset access" as a procrastination technique
Researchers should use the "Scrapheap Challenge" approach too. Users often say: "I can't do anything brilliant with data today… but if I had the following new datasets, tools, methods, and standards perfectly implemented… then I could do something great". Be skeptical. Productive users will create an avalanche of great work with the Top 3 Datasets in any domain.

Invest in data management, data curation: it's the missing link!
Most public sector data projects undervalue the work to turn raw data into something useful. Good data preparation services take messy source data and create analysis-ready datasets; document how variables are derived and validated; make code and documentation openly available; have dedicated staff who understand the data and its meaning. Don't try to organise all data at once: start with the Top 3 Datasets, and common variables. Build capabilities through real analyses. Share code and documentation as you go. These are skilled librarians, who understand the collection, and maintain quality standards. The data platform is the building; true value comes from organising knowledge inside it.

A "catalogue" is not as compelling as senior leaders think
It has been common - for decades - to find large government and health data projects "start" by making an "index", "catalogue" or "gateway"; and often do little more. Catalogues have huge kerb appeal to decision-makers; the value to users is overstated. Users generally know the data they want: they need better access to it. Catalogues often present their lists as somehow tying the contents of data centres together, in some more profound way… they do not. A catalogue has non-zero value: this simple task should be done cheaply and quickly, if deemed necessary. Focus on evaluating real usage and value: new users, genuinely finding and using new datasets, for the first time.

START SMALL AND FOCUSED

Beware the Everything Machine
Don't bite off more than you can chew. Set small goals that - if delivered well - will produce teams, and working methods, and services, that can help on bigger, grander goals.

Complex problems need lots of small wins
Big technology projects often fail big. Working in small steps can feel slower: it's faster and safer. You discover what's failing, before the crisis. You learn what users need, early. You can change direction when you discover an assumption was wrong. Each small success builds confidence and momentum.

EVOLVE AND IMPROVE

Automate relentlessly
When faced with a difficult data task, power through manually, the first few times. If you need to get *one* car over a river, maybe swim it over, piece-by-piece. But real opportunity lies in relentlessly automating every repeatable element, even small tasks. Each automated file transfer, reusable data quality check, reliable well-tested function, standardised governance process… is another stepping stone in place. Eventually you have a bridge everyone can drive over.

Useful open standards grow from actual working tools
Civil servants and policy people often focus first on rules: "standards" for data harmonisation, "frameworks" for approval, "guidelines" for TREs. This work can feel familiar - "like policy" - to those lacking technical skills. But the best technical standards emerge from technical delivery, not meetings. The Internet Engineering Task Force principle ("rough consensus and running code") shows this well: build something useful; develop standards from what works. This same goes for less technical standards: the Five Safes Framework, now used for all TREs, grew from best practice in the oldest ONS TRE. Think about why you need standards, where, and when. Should you really standardise the APIs in your network, at the

beginning, when they only link three services, so the cost of modifying them later would be trivial? Do you really need to standardise the raw data itself, or the tools used to prepare it into usable derivatives?

Fund services you need; be inclusive about other services
Government often thinks about data infrastructure like a kingdom: you pay for it, you control everything in it. But a network approach means organisations can fund specific services they need, to integrate with others'. A cancer charity might fund a service that makes cancer registry data more accessible, within a linked network of services. They aren't trying to control or deliver all data work, or all cancer data work: they're adding one useful service, to a wider ecosystem, to support work in their domain; and they're ensuring it has a clear single function, useful to them and others. Other organisations can build on this service, using it in their own network of services; or ignore it, if it meets no needs. That's how successful digital infrastructure grows: through independent teams building useful services that work together, not one organisation trying to control everything.

BUILD CAPABILITY FIRST

Build teams and people who understand government and health data
You can't buy developers who understand this data: you need to build them. Train them. Nobody has these staff "on-the-shelf": if you outsource them, you're paying someone else to train them today, then own them (and their knowledge) tomorrow.

Find successes: and grow them
The system has almost nothing, anywhere, that currently works well for data platforms. If you see *any* team delivering anything well, with real outputs already delivered, even if it's not precisely what you want: pack resource around them; buy people out of lower priority work; give them staff to train. Do this to expand on successful teams, and build capability to deliver the things you want more. Success enables more success.

Hire managers with deep technical expertise
Good generalist managers are vital. But you also need people with actual technical skills in senior roles. A generalist manager might fool their superiors with surface-level knowledge, but they won't fool their technical staff.

Hire developers with deep technical expertise
This doesn't mean "pay Google salaries". Great technical people will often accept lower salaries on high-impact public projects, *if* they see evidence you're delivering. They won't work on projects that don't really exist; or under managers who haven't shipped working tools; or who don't understand the technical domain. This is another reason why early proofs of actual delivery are critical: delivery is a recruitment flare, attracting good technical people.

If you can't do it, or understand it: you probably can't outsource it either
Big IT procurement fails when organizations outsource the core technical decisions, the overarching design of an ecosystem of services, that they simply don't understand themselves. Data platforms are not impossibly complex. Outsource components. Never - catastrophically - outsource the design *and* delivery of *all* components to one provider. You need internal technical competence to write good contracts, or verify quality. If you lack skills to deliver anything like the task at hand, at any scale, you lack the skills to procure it: so stop. No "compliance requirements" or "knowledge-transfer" clauses in contracts can fix this.

Use technical people for technical work
Your technical people are your scarcest asset. Use them for technical work. Don't waste them on non-technical work.

FAILING WELL

Fail quickly, and informatively
Document what didn't work, and why. Avoid repeating mistakes. Be specific: what was tried, delivered, what didn't work as expected. Focus on failing processes, not people. When teams report problems without fear, they learn and adapt. Share lessons openly. Better solutions emerge because failure becomes informative, not career-limiting.

Keep failure small, and success repeatable
When systems are built from independent services, experiments can fail safely without affecting the whole system.

Failed data platforms don't just waste money: they obstruct better options
Well-funded but poorly-performing programmes generally prioritise relationship-management over delivery. They dominate senior stakeholders' time and attention; fill diaries with update meetings; and create endless documents. Their pseudo-activity creates an illusion of progress. They stop smaller, more effective teams being heard.

Close failing projects well
Be brave, but courteous and empathic. Hand the job of winding a failing service down to people with communications skills, or managerial skills. Consider what services - if any - are truly re-usable elsewhere; be honest about this.

GOVERNANCE, PRIVACY AND TRUST

Work in the open to build trust
With sensitive data, trust comes from transparency, not promises. Working openly means: publishing your code; stating what data you're using, and why; blogging about your work (successes and failures); sharing what you learn. Open working will also attract great technical people.

Privacy through design, not single tools, or rules
Many organisations try to add privacy, as a standalone function, after building systems: through access rules, or buying a "privacy-enhancing technology" product. This misunderstands how privacy works. Privacy needs rules *and* technical architecture: it should shape how you collect, store and share data from the start.

Components are better for security
Discrete services implement precise security controls at each boundary. When something goes wrong (inevitably) the impact is more contained. Standalone services are easier to understand, and therefore audit: you can only secure what you understand.

Publish metrics
Pick metrics that spark meaningful debate; put them online. Examples like "analyses published per million pounds spent", or "percentage of salaries spent on technical staff" drive useful conversations on priorities, and efficiency. Emphasise thoughtful interpretation: sometimes higher is better; sometimes not.

Treat public accountability as a core service
Access to sensitive data requires public trust, which comes from visible accountability. The NDL needs robust oversight to show the public how data is used, and how it helps. This means publishing all projects, and outputs; sharing audit logs, security checks; regular public consultations and professional feedback; with an ethics board providing oversight.

How to Get Our Help

We are happy to give advice: but fundamentally, we build things. We are a large team of specialist public servants with a proven record of shipping highly productive, trusted data services, and open-source tools for data management. OpenSAFELY went from concept to published research in 42 days. We focus on delivering value quickly, and iterating work into scalable platforms, through broad collaboration.

Here we propose two projects that we are keen to deliver - rapidly - and demonstrate the principles in this document.

GP and environmental data
This first walking skeleton would facilitate work on links between local air quality, local weather, and respiratory outcomes during the pandemic. It would deliver rapid value because:

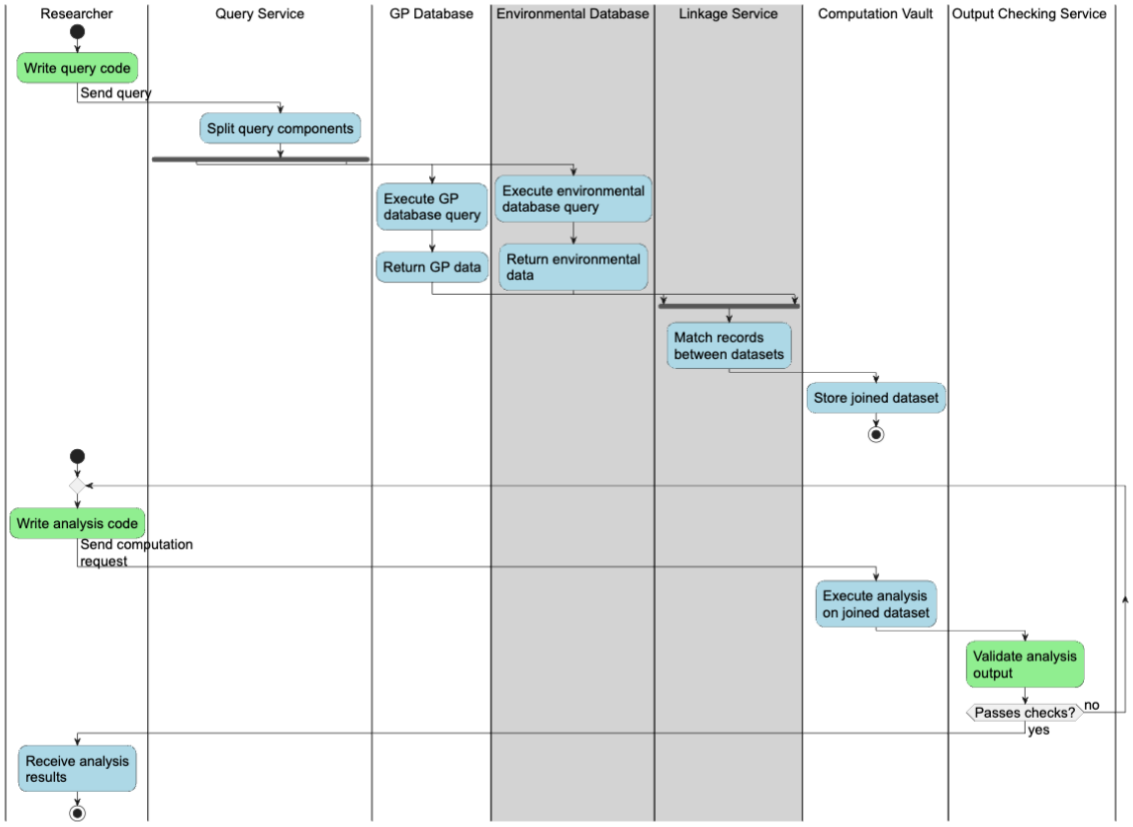
- The environmental data is open, documented and ready to use;
- No complex governance is needed for access;
- OpenSAFELY already handles the GP data pipeline;
- We have collaborators who know the data / field well ;
- The initial research questions are straightforward, but important;
- There's a long queue of high impact research after that.

For researchers, the experience would be straightforward: they write one piece of analysis code that requests the variables they need, then receive their results. Behind the scenes, our services would:

- Split their query to target each dataset appropriately;
- Route GP data requests to OpenSAFELY's existing tools;
- Send air quality data requests to a separate data centre containing that data;
- Link the resulting datasets;
- Run the analysis using OpenSAFELY's proven tools.

As with OpenSAFELY, some processes might start as manual "Wizard of Oz" operations. That's good: the objective is to test assumptions about the shape of the network and the disaggregation of tasks; and identify unforeseen barriers, whether technical, regulatory, organisational, or "other"; while also, from the outset, delivering real data access that draws in collaborators and users. In the following diagram, the grey columns would be new services that could be "Wizard of Oz"; the other columns are existing OpenSAFELY functionality. Blue nodes are automatic operations, green are manual.

The following diagram illustrates the walking skeleton for the proposed GP and environmental data. Blue boxes indicate automated or wizard-of-oz-automated processes. The gray columns indicate new services; all the other columns are existing OpenSAFELY services.



Process flow for GP Data + Environmental Data walking skeleton

GP and *any* department's useful person-level data

Person-level data linkage presents bigger challenges on record linkage, information governance, alongside hearts-and-minds: so it's a good second project. Building on phase one, we can address a more complex integration between GP data and whichever government department's personal data makes for the best pilot, according to feasibility, usefulness of data, NDL priorities, departmental appetites, and generalisability. DWP or HMRC data could be used, for example - minimised and sparse - to explore the impact of health on economic activity, to guide initiatives for economic growth.

However, we make no assumptions. We would pick partners and datasets - as ever - in collaboration; then follow the amber lights - as ever - in the desired direction of travel.

ENDS