39th Meeting of Voorburg Group on Service Statistics

Modernizing Official Statistics: Transition to Open-Source Systems for Production of Producer Price Indexes

Session: Cross Cutting Topics

Xin Ha (Statistics Canada) Steve Martin (Statistics Canada)

Abstract

In response to growing demands for transparency, agility, and cost-efficiency in the production of official statistics, Statistics Canada has modernized the infrastructure supporting its Producer Price Indexes. The transition from legacy corporate systems to reproducible analytical pipelines based on open-source tools has led to significant improvements in reproducibility, methodological flexibility, and operational resilience. This paper outlines the rationale, technologies, implementation strategies, and challenges associated with the modernization initiative, and offers lessons for other national statistical offices undergoing similar transformations.

1 Introduction

The Producer Price Indexes (PPIs) are a collection of macroeconomic statistics that measure the evolution of prices faced by domestic producers over time, and are a core component of Canada's system of official economic statistics. Their primary role is to deflate nominal GDP to produce accurate measures of real economic output, but they also find applications supporting monetary policy by providing timely indicators of inflationary pressures in the economy, indexing industrial contracts, and informing business planning and economic research. The central role of the PPIs in the system of economic statistics, and broader economic analysis, confers a certain importance on the computer systems used to make these statistics. Historically this meant building large, monolithic in-house systems designed to encode a small collection of well-defined workflows for making price indexes. Although there are merits to this approach and it can work well for certain kinds of workflows, we found that these legacy systems were increasingly becoming a barrier to making new PPIs and improving existing ones.

The most salient issue with the incumbent systems was their lack of flexibility, both in terms of the kinds of data and methods they could use, and the types of workflows they supported. These systems were built as stand-alone systems to work with a handful of index-number formulas that were traditionally used for indexes made with survey data, precluding the use of newer or more complex methods that are needed for indexes made with non-survey data (e.g., multilateral index-number formulas for retail services and housing). The tightly-coupled architecture of these systems made it complicated to change the types of methods and data that could be used to make the PPIs. This lack of methodological flexibility spawned numerous "satellite" programs to fill the gaps in the corporate systems, leading to fragmented analytical workflows in which certain, usually quite

important, computations were done outside the corporate systems. The results from these satellite programs would then need to be loaded into the corporate systems in such a way as to give the desired result.

These systems were similarly computationally slow. Computing some PPIs took several hours, limiting the ability to analyze the indexes or incorporate new data late in the production cycle. Dozen of PPIs all needing to use one system created a scheduling problem and constrained the kinds of workflows analysts could use to make their indexes.

Although lack of flexibility was the most obvious drawback of the corporate-systems model for making PPIs, it also created an environment in which the production of the PPIs was treated as an opaque process. This reduced the transparency for how the PPIs were constructed and engagement from analysts. The production of PPIs became largely procedural, reducing analysts' understanding of the underlying processes and, in turn, limiting their ability to enhance the indexes over time.

Over the last several years, we have been moving towards a more modern and flexible approach to building the PPIs at Statistics Canada centered around open-source software and workflows. The transition reflects broader trends in public sector innovation and digital transformation. Although there have been some challenges, the move has improved our ability to construct official statistics.

2 The shift to open-source software

The shift away from large corporate systems to make the PPIs was driven by the adoption of open-source software and workflows that are widely used for data science and statistics. The guiding principle was to cast the construction of the PPIs in the framework a reproducible analytical pipeline (RAP) (NHS RAP Community of Practice 2024).[1] This transition had the obvious goals of moving to a more flexible and transparent architecture for making the PPIs, but also had the aim of moving the process to make the PPIs onto more domain-aligned teams. Giving the team responsible for constructing a PPI ownership of the pipeline to make the index was a key target to advancing how the PPIs are made over time and empowering analysts to move beyond routine tasks. This enhances analysts' comprehension of the underlying processes and methodologies, while simultaneously ensuring systematic and comprehensive documentation that supports transparency, auditability, and long-term maintainability.

2.1 Enabling technologies

The abundance of open-source tools for working with data and statistics means there are several ways to implement a RAP (see, e.g., Rodrigues 2023; The Turing Way Community 2025). Different combinations of tools imply different tradeoffs, and we settled on a set of tools that balanced easy-of-use with good properties around reproducibility. As is usually the case for a RAP, R and Python are the main tools for manipulating data and doing statistical computations, with `git` being used for version control. An internal instance of GitLab complements `git`. Most PPIs are calculated with a similar workflow, and developing new open-source software such as the `piar` (e.g., Martin

[1] The goal was specifically to target the baseline level of the RAP framework, with a quick move to the silver level. We did not aim for the gold level because price indexes require carefully analysis and are not amenable to being built on event-based triggers or on a schedule.

2024) package has been useful for streamlining index calculations, as it allows users to efficiently implement the required computations while maintaining flexibility in methodology and design.

To ensure a consistent computational environment, virtual environments are managed with `conda` (conda contributors, n.d.) and an internal mirror of conda forge. Unlike Python virtual environments or {renv} (Ushey and Wickham 2025), `conda` is a polyglot tool that gives a reasonable level of environment management without being complex to use.[2] Pipeline orchestration and data versioning are both handled by `dvc` (The DVC team and contributors 2020). PPIs are usually made monthly or quarterly, and `dvc` makes it straightforward to version the pipeline used to construct the index at a given point in time. Other pipelining tools like {targets} (Landau 2021) offer more flexibility for defining a pipeline at the expense of data versioning; having these functions coupled makes `dvc` useful for PPIs that have constantly evolving data without computationally heavy workflows. Taken together, these tools give the means to construct a data pipeline for making PPIs with strong properties around reproducibility.

It's worth noting that this is a fairly lightweight set of tools to implement a RAP. Any linux machine with `git` and `conda`, and access to (a mirror of) conda forge, is suitable to implement a RAP with these tools. In practice this is done on cloud infrastructure using a linux-based data science container, but the software imposes few restrictions on the computational environment.

2.2 Modernized workflow

With the tools in place, it becomes straightforward to build the PPIs in the RAP framework. Each PPI is treated as its own project under version control (i.e., its own `git` repo), structured as a research compendium. There is an `environment.yaml` file that defines the `conda` environment, a collection of R/Python scripts that define the steps to get from raw data to final index, and a `dvc.yaml` file that defines how these scripts are executed.[3] Building a PPI is just a case of executing the pipeline with `dvc` in the `conda` environment and, at each point in time, the commit associated with the state of the project used to make the index is tagged with its associated reference period. This makes it easy to checkout the state of the pipeline and computational environment used to build a PPI at that point in time. GitLab is used to collaborate on the construction of the PPIs with a conventional GitOps-style workflow.

As an example, Figure 1 illustrates a simple end-to-end data pipeline used in the modernized production of the PPIs. The pipeline consists of three core steps: data ingestion, cleaning and transformation, and index calculation, each of which is done by one or more R/Python scripts. The scripts to execute these steps are orchestrated with `dvc` and run in a `conda` environment to execute everything in a stable software environment. The results of this pipeline are some analytical reports, a file used by a downstream system to disseminate the PPI, and cloud storage that houses the input and output data from the pipeline. By decoupling operational logic from the tools that support it, this architecture promotes scalability and allows analysts to easily adapt the workflow to make a PPI.

[2] Docker is another popular tool for managing environments. It gives more control over the environment than `conda`, but is consequently harder to use. Similarly, {rix} (Rodrigues and Baumann 2025) is an R-centric alternative to {renv} with more control over the environment.

[3] See work by the UN Task Team for Scanner data for an example.

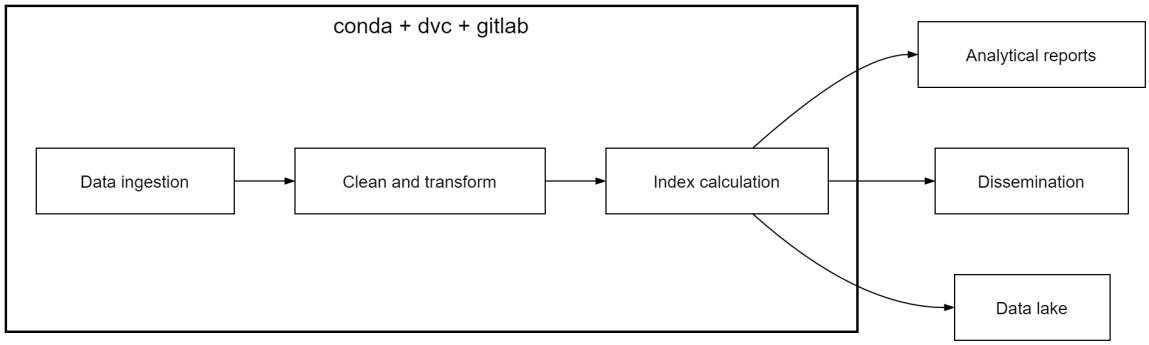

Figure 1: Example of a PPI data pipeline.

The main benefits of this change in architecture were a reduction is the amount of resources needed to make the PPIs (Figure 2) and improved transparency about how the PPIs are calculated. Allowing each PPI to exist as its own pipeline meant that the pipeline could be constructed in a way that made sense for the specific data and methods of the index; consequently, fewer modifications were required to align the workflow for a PPI into the mold of a rigid corporate system, promoting transparency and leaving more time to collect the data needed to make the PPI. A key piece of realizing these benefits was having the pipelines be created and updated over time by domain-aligned teams.

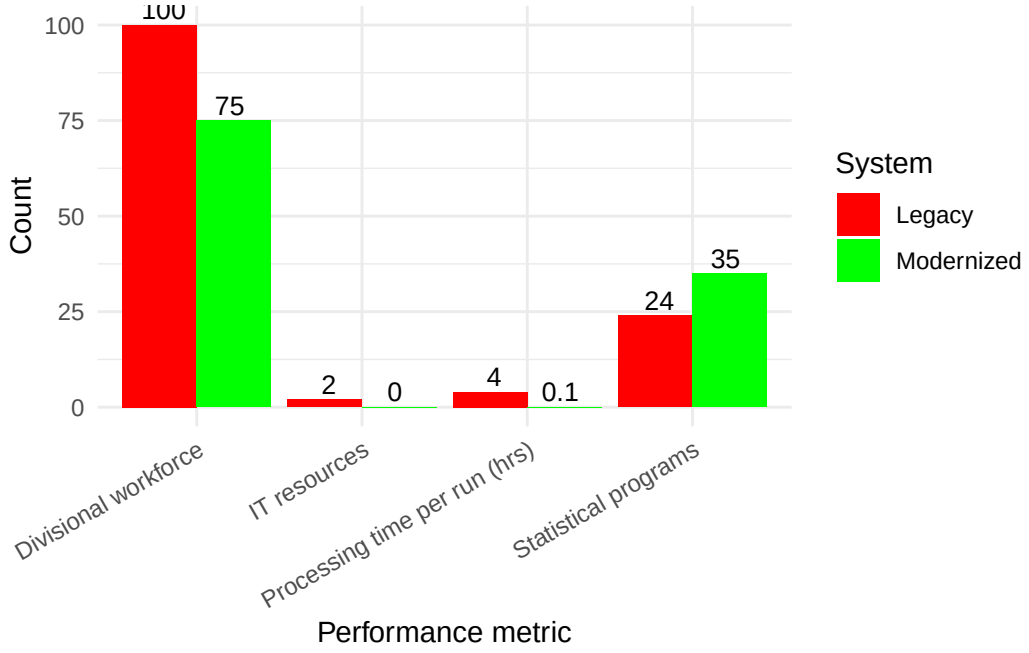

Figure 2: PPI system performance.

3 Challenges

There were two notable challenges that we faced when transitioning towards the model of a RAP for making the PPIs. Adopting new tools and workflows required a significant shift in both technical skills for analysts and team culture. The abundance of high-quality and freely-available material for

R, Python, `git`, and RAPs helped with the mechanics of the new tools. The change in workflow required analysts to better understand the structure and methods of a price index, as these need to be explicitly defined as part of the pipeline. As analysts are now responsible for the entire workflow from data ingestion to index calculation, validation, and dissemination, they are required to understand not only the mechanics of the process but also the underlying economic and statistical concepts. This has transformed the role of analysts to active stewards of the full end-to-end production process.

Integrating open-source tooling and workflows within Statistics Canada's existing IT governance framework was another challenge that required considerable collaboration with corporate IT and infrastructure teams. This has become easier as open-source tools become more ubiquitous at Statistics Canada, but nonetheless represents a change in paradigm that required close collaboration with corporate IT.

4 Conclusion

The adoption of open-source tools and workflows has transformed the way Statistics Canada produces its PPIs. Structuring a PPI as a RAP with a lightweight and open-source software stack provides a foundation for transparency, reproducibility, and scalability. This transition enables more agile statistical production, elevates analysts' skills and prepares Statistics Canada for future innovations in real-time data, machine learning, and big data analytics. Our experience can serve as a model for other statistical programs modernizing their infrastructure and workflows in alignment with international best practices and digital government strategies.

References

conda contributors. n.d. "conda: A system-level, binary package and environment manager running on all major operating systems and platforms." https://github.com/conda/conda.

Landau, William Michael. 2021. "The Targets R Package: A Dynamic Make-Like Function-Oriented Pipeline Toolkit for Reproducibility and High-Performance Computing." *Journal of Open Source Software* 6 (57): 2959. https://doi.org/10.21105/joss.02959.

Martin, Steve. 2024. "piar: Price Index Aggregation R." *Journal of Open Source Software* 9 (101): 6781. https://doi.org/10.21105/joss.06781.

NHS RAP Community of Practice. 2024. "RAP Community of Practice." https://nhsdigital.github.io/rap-community-of-practice/.

Rodrigues, Bruno. 2023. *Building Reproducible Analytical Pipelines with R*. https://raps-with-r.dev/.

Rodrigues, Bruno, and Philipp Baumann. 2025. *rix: Reproducible Data Science Environments with "Nix"*. https://doi.org/10.32614/CRAN.package.rix.

The DVC team and contributors. 2020. "DVC: Data Version Control - Git for Data & Models"." https://doi.org/10.5281/zenodo.012345.

The Turing Way Community. 2025. *The Turing Way handbook for reproducible, ethical and collaborative research* (version 1.2.3). https://doi.org/10.5281/zenodo.15213042.

Ushey, Kevin, and Hadley Wickham. 2025. *renv: Project Environments*. https://doi.org/10.32614/CRAN.package.renv.