irpf-bridge: Controlled Writeback into Official IRPF Desktop Save-State Artifacts

David Barros Hulak, Arthur Freitas Ramos

https://doi.org/10.1590/SciELOPreprints.15845

Submitted on: 2026-04-15
Posted on: 2026-05-04 (version 1)
(YYYY-MM-DD)

irpf-bridge: Controlled Writeback into Official IRPF Desktop Save-State Artifacts

irpf-bridge: Writeback Controlado em Artefatos Oficiais de Save-State do IRPF Desktop

Arthur Freitas Ramos   David Barros Hulak

2026

Arthur Freitas Ramos[1]
David Barros Hulak[2]
[1] Microsoft Corporation, Redmond, USA. arfreita@microsoft.com. https://orcid.org/0009-0003-3568-0325
[2] Independent Researcher. dbhulak@gmail.com. https://orcid.org/0009-0002-8056-1774

Abstract

The Brazilian individual income-tax return (IRPF) is prepared in a desktop application distributed by the federal tax authority, the Programa Gerador da Declaração (PGD) [1], which persists declaration state in structured local artifacts. Public automation efforts usually bypass these artifacts through GUI scripting, parallel declaration representations, or read-only inspection. This paper investigates a different objective: whether the official save-state family can be treated as the primary read-write object while keeping humans in control of every proposed change. We present irpf-bridge [6], a local Python research prototype that ingests baseline XML/BKP pairs, maps supporting-document evidence into candidate deltas, validates them, and writes back narrow 2026 declaration slices under explicit scope boundaries. Results from the public repository combine deterministic rerun-convergence checks, exercised preservation evidence, and bounded manual reopening and acceptance observations in the official PGD. The contribution is not broad tax automation; it is a design-science feasibility result showing that, for the tested scope, the official artifact boundary can be crossed programmatically in a safe, reviewable, and verifiable manner without making GUI automation the primary integration surface.

Keywords: IRPF; official artifacts; writeback; digital government; design science; information systems; human review

Resumo

A declaração anual de Imposto de Renda da Pessoa Física (IRPF) no Brasil é preparada em um programa desktop distribuído pela Receita Federal, o Programa Gerador da Declaração (PGD) [1], que persiste o estado da declaração em artefatos locais estruturados. Esforços públicos de automação costumam contornar esses artefatos por meio de scripting de interface gráfica, representações paralelas da declaração ou inspeção sem escrita controlada. Este trabalho investiga um objetivo diferente: verificar se a família oficial de save-state pode ser tratada como objeto primário de leitura e escrita sem retirar do humano o controle sobre cada mudança proposta. Apresentamos o irpf-bridge [6], um protótipo local em Python que ingere pares baseline XML/BKP, mapeia evidências documentais em deltas candidatos, valida esses deltas e escreve fatias estreitas da declaração 2026 sob limites explícitos de escopo. Os resultados públicos combinam verificações determinísticas de convergência por reexecução, evidência exercitada de preservação e observações manuais delimitadas de reabertura e aceitação no PGD oficial. A contribuição não é automação fiscal ampla; é um resultado de viabilidade em Design Science mostrando que, para o escopo testado, a fronteira do artefato oficial pode ser cruzada programaticamente de forma segura, revisável e verificável, sem tornar a automação de GUI a superfície primária de integração.

Palavras-chave: IRPF; artefatos oficiais; writeback; governo digital; design science; sistemas de informação; revisão humana

1. Introduction

Every year, millions of Brazilian taxpayers gather income statements, bank extracts, and other supporting documents, and manually transcribe the relevant values into the Programa Gerador da Declaração (PGD) for the individual income tax (Imposto de Renda da Pessoa Física), distributed by the Receita Federal do Brasil [1]. The PGD is a desktop application that saves the declaration state in structured local artifacts — pairs of XML and BKP files that the taxpayer can reopen, review, and eventually transmit.

The gap between source documents and a finalized declaration is still largely manual. The representative public projects reviewed for this work are fragmented: some reverse-engineer official layouts and artifact families, while others generate narrow declaration outputs outside the PGD working state [7-9]. In either case, the user must still verify that the external processing has not corrupted or misrepresented the official declaration state.

This work investigates a different strategy: **treating the official save-state artifacts as the single source of truth, reading from and writing to them directly, and keeping every proposed change reviewable before application**.

In Information Systems terms, this is a digital-government interoperability problem [13, 18, 19]: the challenge is to design an artifact that can interoperate with an authoritative public-sector system without sacrificing auditability, user review, or continuity with the incumbent workflow. In Brazil specifically, interoperability with official systems is shaped by legal and institutional constraints [14, 22] as well as technical format concerns, and prior work in the Brazilian IS community has shown that public-sector integration consistently exceeds purely technical plumbing [15, 21].

The result is a research software prototype, irpf-bridge [6], that implements a local Python pipeline following a strict **read → diff → validate → review → write** progression. The prototype is intentionally narrow in scope and explicit about its limitations.

The central research question is whether a narrowly scoped bridge can interoperate with the official IRPF save-state while keeping provenance, human review, and the official desktop workflow as first-class constraints. Framed as a design-science improvement contribution [10-12], the paper makes three bounded contributions: (1) an artifact taxonomy of the official IRPF save-state boundary observed in PGD 2026; (2) a local bridge pipeline that keeps reviewable deltas and provenance explicit rather than replacing the official workflow; and (3) bounded empirical evidence — formative and technical-risk-focused in the FEDS evaluation sense [17] — from deterministic convergence checks and manual reopening in the official PGD for the tested slice.

The public repository, including source code, synthetic fixtures, golden outputs, validation documentation, and limitations, is available at https://github.com/d0d1/irpf-bridge under the GPL-3.0-only license [6].

2. Related Work

2.1 IRPF Ecosystem and Prior Tooling

The IRPF ecosystem lacks documented public APIs for third-party integration. The representative public projects reviewed for this work fall into three overlapping categories [7-9]: artifact reverse engineering and format tooling; parallel representations and narrow generators; and read/transform workflows that inspect official artifacts but do not keep the official XML/BKP save-state as the primary reviewed object through a complete read → review → write cycle.

The present work differs by treating the save-state artifact as the primary object for both reading *and* writing, keeping the official ecosystem as the boundary rather than bypassing it.

Direct inspection of the public official 2026 PGD package, recorded in the project's research notes [9], found XML descriptors that point to declaration and mapping layouts. Earlier public projects had already pointed to descriptor files inside official program bundles [7], but the present work uses them only as structural clues, not as stand-alone proof of a complete writeback surface.

2.2 Design Science Research and Artifact Evaluation

The paper follows the Design Science Research (DSR) paradigm for Information Systems, in which the primary output is a purposeful artifact addressing an identified organizational or societal problem [10]. The methodology follows the DSRM process model — problem identification, objectives definition, design and development, demonstration, evaluation, and communication [11] — and the contribution is positioned as an *improvement* artifact in the sense of Gregor and Hevner [12]: a new solution (artifact-first writeback with human-review preservation) to a known class of problem (interoperability with an authoritative public-sector desktop system).

Because the artifact is at an early maturity stage, we follow the FEDS framework for DSR evaluation [17], which distinguishes formative and summative evaluation episodes matched to artifact risk and maturity. The current evaluation episode is primarily *technical-risk-focused and formative*: deterministic convergence checks demonstrate internal validity, and bounded manual reopening observations in the official PGD provide initial real-world fidelity evidence. A broader summative evaluation — covering larger field studies, independent replication, and additional declaration scopes — is explicitly deferred and documented in the limitations.

2.3 Digital Government Interoperability

The bridge is situated within the broader research agenda on e-government integration and interoperability, which Scholl and Klischewski [13] frame as a socio-technical challenge involving technical, organizational, legal, and political dimensions. Dawes [19] argues that digital governance must address privacy, security, and institutional stewardship alongside technical integration, while Janowski [18] emphasizes that digital-government arrangements must be context-sensitive and institutionally embedded rather than generic.

In the Brazilian context, Xisto et al. [14] propose a reference architecture for digital-government interoperability that emphasizes reuse and coordinated service management across public-sector systems. Nascimento Silva and Augusto [22] map the legal and normative framework for interoperability and data sharing in Brazilian digital government, showing that integration is constrained by policy-governed institutional practice as well as technical compatibility. Paiva et al. [15], writing in iSys, demonstrate that independent Brazilian public-sector information systems create integration challenges that exceed purely technical concerns, and Monteiro et al. [21], also in iSys, address interoperability maturity modeling for software systems. The present artifact inhabits this space: it bridges a legacy desktop save-state format within the constraints of the official IRPF ecosystem, rather than proposing a new centralized architecture.

2.4 Tax-Administration Digitalization

A recent systematic review by Hesami et al. [16] shows that digital transformation of tax administration — particularly e-invoicing and prefilled returns — reshapes compliance workflows and reduces administrative costs, but also raises concerns about automation opacity. Doxey et al. [20] provide experimental evidence that prefilled tax returns affect taxpayer reporting behavior depending on what the system includes or omits, underscoring the importance of human review and explicit uncertainty in any system that proposes declaration content. The present work sits at the intersection of these concerns: it operates downstream of prefilled data, proposing changes into a declaration that the taxpayer has already partially prepared, and insists on review visibility precisely because silent automation in a tax-submission context carries high compliance risk.

3. Artifact Taxonomy

Analysis of real PGD 2026 save-state sets revealed an artifact landscape richer than official public documentation alone suggests. The project distinguishes four artifact families:

3.1 Primary Save-State Family: XML / BKP

Pairs of `.xml` and `.BKP` files produced by the PGD were observed to carry the same structural declaration content, modulo save-time normalization and timestamp differences. This pair carries the complete serialization of the declaration object graph — header metadata, all declaration sections, computed fields, and internal program markers.

This family is the **primary bridge target**: the artifact the taxpayer saves, that the PGD reopens, and that this project reads from and writes back to.

3.2 Secondary Bridge/Export Family: DBK

Alongside the XML/BKP pair, the PGD writes a `.dbk` file that follows a fixed-width, record-oriented encoding. Its role appears to be export or inter-system bridging rather than serving as the primary working save-state. The prototype ingests `.dbk` for structural comparison only and explicitly excludes it from the writeback surface.

3.3 Regenerated Configuration: CONF

The PGD stores a companion `.conf` file whose content is derived from the final XML payload through a deterministic hash formula. In the current state of evidence, the observed keyed-SHA-1 formula has been confirmed both for the declaration index (`iddeclaracoes.conf`) and for inspected live per-declaration `.conf` samples from the current 2026 build.

The project regenerates a `.conf` from the rewritten XML bytes as part of write-back operations in the current narrow scope. This is sufficient for the manually reopened slice so far, but the broader control/hash/open-flow surface remains under investigation.

3.4 Official-Layout-Oriented Families: DEC / REC

Official public documentation describes annual declaration (`.DEC`) and receipt (`.REC`) layouts in fixed-width format [2]. These remain important for validation and historical reference, but the actual baseline for the user's save-state in 2026 is the XML/BKP family.

4. Architecture and Pipeline

irpf-bridge is implemented as a single-host Python CLI application targeting Linux and WSL. Its execution model enforces a fixed five-phase sequence — **read → diff → validate → review → write** — so that no proposed change reaches the output artifact without passing through each preceding gate.

4.1 Baseline Ingestion

Execution starts by loading a declaration snapshot through a year-specific adapter (currently 2026). For the exercised writeback scope, the adapter consumes PGD-produced XML/BKP pairs as the primary input; the broader reading surface also accepts earlier fixed-width `.DEC` artifacts. The loaded state is frozen into an **immutable canonical snapshot** whose every field preserves a traceable link to the raw artifact. Fields that fall outside the adapter's mapping are carried forward opaquely rather than dropped.

4.2 Document Ingestion and Evidence Extraction

Each supporting document is fingerprinted, classified against a registry of recognized document types, and decomposed into evidence units — the smallest reviewable fragments that carry factual assertions. At present, the ingestion surface is limited to pre-structured JSON payloads; native PDF parsing, scanning, and OCR remain deferred.

4.3 Fact Interpretation and Candidate Mapping

Evidence-derived facts undergo a normalization step that produces interpreted tax facts, each of which is then projected onto a specific declaration target within the supported slices. Every candidate carries:

- a delta classification (`add`, `update`, `remove`, `no-change`, or `unknown`);
- a full provenance chain linking the candidate back through its tax fact, evidence unit, and originating document;
- a named confidence level;
- completeness and conflict flags.

4.4 Validation

Two validation layers run before any proposal reaches the review surface. Structural validation confirms that proposed values conform to the declaration schema, while tax-reasoning validation checks logical consistency against IRPF rules and surrounding declaration context. Any proposal that triggers a blocking error is quarantined — still visible for inspection, but excluded from downstream stages.

4.5 Review Bundle

The normative output of each run is a structured review bundle that records the baseline snapshot, all proposed deltas, per-proposal validation verdicts, complete provenance chains, and any unresolved items. A companion human-readable summary is derived from this bundle for manual inspection.

4.6 Writeback

When the user accepts a set of deltas, the pipeline applies them to a fresh copy of the baseline XML and produces four output artifacts:

- the modified `.xml` file;
- a byte-identical `.BKP` sibling;
- a regenerated `.conf` computed from the updated XML under the observed hash formula for the current scope;
- a machine-readable writeback report.

Immediately after writing, a convergence check re-ingests the modified artifact and verifies that the same inputs now produce zero remaining deltas for the supported slices.

5. Supported Scope

The current implementation supports a narrow 2026 scope, with different evidence strength per slice:

Slice	Record	Public synthetic evidence	Automated writeback convergence	Manual PGD reopen evidence
PJ taxable income	`_21` (Rendimentos tributáveis de PJ)	Yes	Yes	Yes

Slice	Record	Public synthetic evidence	Automated writeback convergence	Manual PGD reopen evidence
Dependent anchor creation (bounded core)	_25 (Dependentes)	Yes	Yes	Yes (bounded; see the structured _25 PGD proof)
Dependent PJ taxable income	_32 (Rendimentos tributáveis de PJ de dependente)	Yes	Yes	Yes (bounded; see the structured _32 PGD proof; the exercised run requires a compatible canonical dependent anchor)
Payment deductions	_26 (Pagamentos efetuados / doações)	Yes	Yes	Yes (codes `01/26` additions)
Debt updates (domestic, update-only)	_55 (Dívidas / ônus reais)	Yes	Yes	Yes (update-only)
Deposit-like assets	_27 (Bens e Direitos, deposit-like items)	Yes	Yes	Yes
Structured asset positions (non-crypto)	_27 (Bens e Direitos, structured path)	Yes	Yes	Yes (list-view)
Self-custody crypto	_27 (grupo 08 structured path)	Yes	Yes	Yes (list-view)
Exchange-custody crypto	_27 (structured path)	Yes	Yes	Yes (list-view)

Broader _25 dependentes semantics, crypto/Coinbase treatment, brokerage positions, foreign income, exterior-liability semantics, and other declaration sections remain explicitly out of scope.

Document ingestion currently requires prepared structured JSON; arbitrary PDFs and scans are not yet first-class inputs.

6. Security Model and Human Review Boundary

Because the bridge modifies artifacts that carry legal and financial consequences, irpf-bridge enforces a layered set of safety invariants:

1. **Copy-on-write baseline.** The pipeline never overwrites the original save-state file; every write operation targets a distinct output copy, preserving the pre-modification artifact for comparison.
2. **Differential mutation only.** Rather than regenerating the entire declaration, the pipeline expresses changes as deltas against the frozen baseline. Sections outside the supported scope are intended to survive unchanged, though the strength of that guarantee is path-dependent and not yet universal.
3. **End-to-end provenance.** Each reviewable proposal must trace an unbroken chain from the originating source document through its evidence unit, the interpreted tax fact, and the resulting candidate mapping. Proposals that lack any link in this chain are demoted to blocked status.
4. **Pre-write validation gate.** Proposals that trigger blocking validation errors are quarantined in the review bundle for inspection but are not forwarded to the writeback stage.
5. **Post-write convergence verification.** After writeback, the pipeline re-ingests the modified artifact and confirms that repeating the same supported inputs yields zero residual deltas — a deterministic self-consistency check.
6. **Human-review primacy.** The structured review bundle, not the written artifact, is the primary deliverable. Writeback is an available action, but the review surface is the default proof of value.
7. **Official-desktop acceptance as the ultimate gate.** Reopening the rewritten artifact in the official PGD remains the authoritative acceptance test — a manual, tranche-specific check that the project does not bypass.

7. Validation Evidence

7.1 Synthetic Fixtures

The repository now includes ten fixture scenarios with deterministic manifests, synthetic inputs, and golden outputs. Automated checks (`make fixture-check`) exercise baseline reading, canonical extraction, mapping, and deterministic review output across the current `_21`, `_25`, `_26`, `_27`, `_32`, and `_55` public synthetic surface. Separate automated convergence tests cover the writeback paths. All fixtures are deterministic and depend only on synthetic material committed to the repository.

7.2 Rerun Convergence

After writeback, a verification pass confirms that the same supported inputs produce zero deltas — demonstrating that the targeted changes were absorbed into the artifact. This is verified programmatically in automated test suites.

7.3 Local Testing with Real Artifacts

A local-only harness supports staging of real PGD save-state artifacts for writeback testing. Current local results show supported updates applied, `changed=true`, and rerun convergence verification with zero remaining updates and zero validation errors.

7.4 Manual PGD Reopening Observation

A human observer reopened rewritten artifacts in the official 2026 PGD for the following slices: `_21` PJ income, bounded `_25` dependent-anchor registration, bounded `_32` dependent PJ-income list-view, `_26` payment additions (codes `01`/`26`), deposit-like `_27`, structured `_27` list-view rendering, and update-only domestic `_55` debt entries. Four caveats apply to this evidence:

- it is manual rather than automated;
- it covers only the list-view and validation slices that were exercised;
- it is anchored to the 2026 PGD build available at the time of testing;
- a single observer performed it — no independent reproduction has occurred.

The `_25` proof is particularly noteworthy because it shows the writeback path creating a dependent anchor that the official Dependentes section renders with the correct code, CPF, name, and birth date, and from which the PGD itself auto-computes the dependent deduction amount. A complementary GUI-originated proof shows that the PGD can create and persist such an anchor independently, so the dependent evidence family is reinforced from both directions.

The open `_32` gap is therefore no longer the anchor's provenance but the absence of a single end-to-end public run that begins from a GUI-created or pre-filled anchor and then walks through the bounded `_32` acceptance slice.

None of this evidence extends to detail-view forms, electronic transmission, or the complete control/hash/open-flow surface for general-purpose save-state mutation.

7.5 Remaining Gaps

- No automated PGD acceptance verification exists.
- Writeback in `.dbk` format is not implemented.
- The broader control/hash/open-flow surface is not proven for general-purpose mutation, even though the observed keyed-SHA-1 formula now matches the inspected live per-declaration `.conf` sample for the current 2026 build.
- Document ingestion requires structured JSON, not arbitrary PDFs.

- Manual PGD acceptance remains single-observer and has not yet been independently reproduced by third parties.

8. Reproducibility

All source code, synthetic fixtures, golden outputs, and validation documentation are committed to the public repository. The following commands reproduce all public evidence from a clean checkout with Python ≥ 3.11 on Linux:

```
git clone https://github.com/d0d1/irpf-bridge.git
cd irpf-bridge
make setup
make test
make fixture-check \
  SCENARIO=fixtures/scenarios/minimal-noop/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/informe-rendimentos-pj-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/dependent-anchor-creation-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/dependent-pj-income-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/payment-deductions-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/debts-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/noncrypto-deposit-like-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/noncrypto-asset-positions-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/crypto-self-custody-supported/manifest.json
make fixture-check \
  SCENARIO=fixtures/scenarios/crypto-exchange-custody-supported/manifest.json
```

These commands use only committed code and fixtures. No real taxpayer artifacts, external services, or private material is required.

9. Limitations and Non-Goals

Bounded declaration coverage. The prototype addresses only a fixed set of 2026 declaration slices (_21, _25, _26, _27, _32, _55). Every claim in this paper is confined to that set; extending coverage is planned work, not a current capability.

Pre-structured document intake. The ingestion surface accepts only prepared JSON payloads. Support for PDF, OCR, and unstructured text inputs remains deferred.

One layout year exercised. Although the adapter boundary is designed for annual rotation, only the 2026 adapter has been implemented and tested.

Stops before transmission. The project scope ends at save-state mutation. Filing, receipt generation, and electronic transmission are excluded.

No graphical-interface scripting. GUI automation is rejected as an architectural strategy, not merely deferred.

No tax or accounting guidance. The tool assists with mechanical aspects of declaration preparation. Professional review remains necessary.

Manual desktop acceptance. Whether the official PGD accepts a rewritten artifact can only be verified by manually reopening it — an inherently tranche-specific, single-observer check at this stage.

Incomplete control-surface characterization. The `.conf` regeneration relies on the keyed-SHA-1 formula observed for the current narrow scope. The full set of desktop-program control surfaces for arbitrary save-state mutation is not yet mapped.

AI-assisted development. Language-model assistance was used throughout development. All code, fixtures, and documentation were human-reviewed before repository inclusion.

10. Discussion

10.1 The Importance of the Artifact Advance

The tax-mapping logic itself is deliberately narrow and contributes little novelty. What the project does contribute is observational evidence that **the PGD save-state boundary is not opaque to external tooling**: for the tested scope, an outside process can parse the desktop program's native files, introduce validated mutations, recompute the required companion artifacts, and yield a result that the desktop program re-opens without error.

This finding reframes the design space for IRPF-related tools. Where previous approaches either replicated the declaration outside the official format or drove the GUI programmatically, the artifact-bridge pattern keeps the official save-state as the single authoritative object — proposing reviewable deltas rather than replacement representations.

From an Information Systems standpoint [10, 13, 18], the pattern is significant because it positions interoperability as a governance and control challenge within a high-risk public-sector system, where institutional norms constrain what can be delegated to automation and what must stay under human oversight [19, 22]. The bridge neither supplants the official desktop application nor conceals uncertainty behind a shadow declaration; it surfaces deltas, preserves provenance, and leaves the taxpayer in charge of the final declaration trajectory.

10.2 Openness and Public-Interest Posture

The repository is open source under GPL-3.0 [6] precisely because work that proposes modifications to official tax artifacts should be public and verifiable. The code, fixtures, evidence, and limitations are open for inspection by any interested party.

10.3 Narrow Scope as Contribution

The restricted declaration coverage is intentional at this maturity stage rather than a sign of architectural ceiling. Every new slice that clears writeback convergence and official-desktop reopening enlarges the validated surface while reusing the same pipeline architecture.

10.4 Evaluation Positioning

The evaluation evidence in this paper is deliberately bounded. Following the FEDS framework [17], we classify the current evaluation episode as formative and technical-risk-focused: the primary goal is to establish that the artifact can cross the official save-state boundary without corruption for the tested scope, rather than to measure broad organizational impact or adoption. The deterministic rerun-convergence checks address *internal validity* — the artifact does what it claims for the tested paths. The manual PGD reopening observations address *external fidelity* — the official program accepts the result for the tested slice. Neither of these claims extends beyond the documented scope.

A summative evaluation — involving independent replication, broader declaration coverage, user studies, or field deployment — is appropriate for a later maturity stage. This boundary is consistent with DSR practice: Gregor and Hevner [12] note that the evaluation strategy should match the contribution type and maturity, and Venable et al. [17] argue that premature summative evaluation of an immature artifact risks measuring the wrong thing. The present paper's contribution is the feasibility evidence and the artifact-first integration pattern, not a claim of field-validated efficacy.

11. Conclusion

This work presented irpf-bridge [6], a research software prototype that demonstrates controlled writeback into official IRPF save-state artifacts. The bridge pipeline reads the PGD XML/BKP save-state, maps supporting documents into candidate deltas with provenance, validates proposals, and writes back in the same artifact format with rerun convergence verification. Manual reopening observation in the official PGD indicates viability for the narrow tested slice.

Framed as a design-science improvement contribution [10-12], the main result is bounded feasibility evidence for an artifact-first integration pattern in digital government: instead of bypassing the official ecosystem, tools can operate within it, proposing reviewable changes against the real declaration save-state. The evaluation follows a formative, technical-risk-focused strategy appropriate for the current artifact maturity [17], and the paper documents both the demonstrated evidence boundary and the explicit path toward summative evaluation in future work. This result opens space for future tools — within the Brazilian IRPF ecosystem and potentially in analogous public-sector desktop-artifact systems — that extend coverage without changing the architecture.

The public repository is available at https://github.com/d0d1/irpf-bridge [6] with code, fixtures, golden outputs, and validation and limitations documentation, enabling independent reproduction of all public evidence.

Data Availability Statement

Data availability statement: the public repository contains all source code, synthetic fixtures, golden outputs, and documentation: https://github.com/d0d1/irpf-bridge [6].

License: GPL-3.0-only.

The repository does not contain real taxpayer data. Local-only artifacts used for testing are not committed and are explicitly excluded via `.gitignore`.

Funding

This project received no external funding.

Conflict of Interest

The authors declare no conflicts of interest.

Author Contributions (CRediT)

- **Conceptualization:** David Barros Hulak.
- **Methodology:** David Barros Hulak.
- **Software:** David Barros Hulak; Arthur Freitas Ramos.
- **Validation:** David Barros Hulak; Arthur Freitas Ramos.
- **Investigation:** David Barros Hulak.
- **Writing – Original Draft:** David Barros Hulak.
- **Writing – Review & Editing:** David Barros Hulak; Arthur Freitas Ramos.

These contributor-role labels follow the CRediT taxonomy [5].

Submission-platform note

This preprint package is prepared for submission to SciELO Preprints [3], which SciELO currently operates on the Open Preprint Systems platform [4]. These references are included only to document the submission venue and platform context for this manuscript package; they are not evidentiary sources for the IRPF workflow claims.

References

1. Receita Federal do Brasil. Baixe aqui o Programa Gerador de Declaração (PGD) da Declaração do Imposto sobre a Renda das Pessoas Físicas (DIRPF). Available at: https://www.gov.br/receitafederal/pt-br/centrais-de-conteudo/download/pgd/dirpf. Accessed: 2025.
2. Receita Federal do Brasil. Leiautes do Programa da Declaração do Imposto de Renda da Pessoa Física (DIRPF). Available at: https://www.gov.br/receitafederal/pt-br/centrais-de-conteudo/publicacoes/documentos-tecnicos/dirpf. Accessed: 2025.
3. SciELO. SciELO Preprints — About the Server. Available at: https://preprints.scielo.org/index.php/scielo/about. Accessed: 2025.
4. Public Knowledge Project (PKP). Open Preprint Systems (OPS). Available at: https://pkp.sfu.ca/software/ops/. Accessed: 2025.
5. NISO. CRediT — Contributor Roles Taxonomy. Available at: https://credit.niso.org/. Accessed: 2025.
6. irpf-bridge contributors. irpf-bridge: Research-first IRPF artifact-bridge prototype [software]. Version 0.1.0. Available at: https://github.com/d0d1/irpf-bridge. License: GPL-3.0-only.
7. RafaelEstevamReis. IRPF [software]. Available at: https://github.com/RafaelEstevamReis/IRPF. Accessed: 2025.
8. JulioCesar82. gerar-IR-de-renda-variavel [software]. Available at: GitHub repository. Accessed: 2025.
9. irpf-bridge contributors. Source inventory and prior-art review [project documentation]. Available at: https://github.com/d0d1/irpf-bridge/blob/main/docs/research/source-inventory.md. Accessed: 2025.
10. Hevner, Alan R.; March, Salvatore T.; Park, Jinsoo; Ram, Sudha. Design Science in Information Systems Research. *MIS Quarterly*, v. 28, n. 1, p. 75-106, 2004. Available at: https://doi.org/10.2307/25148625.
11. Peffers, Ken; Tuunanen, Tuure; Rothenberger, Marcus A.; Chatterjee, Samir. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, v. 24, n. 3, p. 45-77, 2007. Available at: https://doi.org/10.2753/MIS0742-1222240302.
12. Gregor, Shirley; Hevner, Alan R. Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly*, v. 37, n. 2, p. 337-355, 2013. Available at: https://doi.org/10.25300/MISQ/2013/37.2.01.

13. Scholl, Hans J.; Klischewski, Ralf. E-Government Integration and Interoperability: Framing the Research Agenda. *International Journal of Public Administration*, v. 30, n. 8-9, p. 889-920, 2007. Available at: https://doi.org/10.1080/01900690701402668.

14. Xisto, Adriana; Sommer, Felipe; Costa, Marcus Vinicius; Silva, José Lutiano Costa da; Cappelli, Claudia; Nunes, Vanessa. Interoperability of Digital Government Services: A Brazilian Reference Architecture Model to Promote Communication, Management, and Reuse of Solutions. In: *Enterprise Design, Operations, and Computing*, p. 91-107, 2022. Available at: https://doi.org/10.1007/978-3-031-17604-3_6.

15. Paiva, Eduardo Soares; Revoredo, Kate Cerqueira; Baião, Fernanda Araujo. DW-CGU: Integração dos Dados do Portal da Transparência do Governo Federal Brasileiro. *iSys - Brazilian Journal of Information Systems*, v. 9, n. 1, p. 6-32, 2016. Available at: https://doi.org/10.5753/isys.2016.298.

16. Hesami, Siamand; Jenkins, Hatice; Jenkins, Glenn P. Digital Transformation of Tax Administration and Compliance: A Systematic Literature Review on E-Invoicing and Prefilled Returns. *Digital Government: Research and Practice*, v. 5, n. 3, p. 1-20, 2024. Available at: https://doi.org/10.1145/3643687.

17. Venable, John; Pries-Heje, Jan; Baskerville, Richard. FEDS: a Framework for Evaluation in Design Science Research. *European Journal of Information Systems*, v. 25, n. 1, p. 77-89, 2016. Available at: https://doi.org/10.1057/ejis.2014.36.

18. Janowski, Tomasz. Digital government evolution: From transformation to contextualization. *Government Information Quarterly*, v. 32, n. 3, p. 221-236, 2015. Available at: https://doi.org/10.1016/j.giq.2015.07.001.

19. Dawes, Sharon S. The Evolution and Continuing Challenges of E-Governance. *Public Administration Review*, v. 68, n. s1,

    p. S86-S102, 2008. Available at: https://doi.org/10.1111/j.1540-6210.2008.00981.x.

20. Doxey, Marcus M.; Lawson, James G.; Stinson, Shane R. The Effects of Prefilled Tax Returns on Taxpayer Compliance. *The Journal of the American Taxation Association*, v. 43, n. 2, p. 63-85, 2021. Available at: https://doi.org/10.2308/jata-18-055.

21. Monteiro, Erasmo Leite; Mascarenhas, Ana Patrícia Fontes Magalhães; Maciel, Rita Suzana Pitangueira. A Strategy to Specify Customizable Interoperability Maturity Models for Software Systems. *iSys - Brazilian Journal of Information Systems*, v. 18, n. 1, 2025. Available at: https://doi.org/10.5753/isys.2025.4368.

22. Nascimento Silva, Patrícia; Dos Santos Augusto, Ana Maris Fernandes. Governo digital: arcabouço legal sobre interoperabilidade e compartilhamento de

dados no Brasil. *Palabra Clave (La Plata)*, v. 14, n. 2, e250, 2025. Available at: https://doi.org/10.24215/18539912e250.

This manuscript describes the public state of the irpf-bridge repository. It does not contain taxpayer data, specific institution names, account numbers, or other non-public material.