BY JAYADEVAN VIJAYAN

Automation of financial reporting

Financial reporting in insurance is the process of documenting and communicating the financial position and profitability of the company over specific periods, typically on a quarterly or yearly basis. The required disclosures vary by accounting standard (e.g. IFRS 17, US GAAP or local) and may be prescribed by prudential regulators. For example, insurers based in Europe are subject to the Solvency II regime, which requires extensive regular reporting.

Despite the high volumes and complexity involved, many insurers rely on manual processes involving a combination of spreadsheets and text editors such as MS Excel and Word when preparing their financial reports. These processes demand significant effort in transferring data, creating and updating links, tables and charts, and drafting commentary. While these tools are flexible and intuitive, enabling collaboration within and across departments, using them as part of a reporting process is time-consuming and prone to errors.

Process automation offers insurers an opportunity to reimagine financial reporting. By adopting automation, insurers can significantly reduce the time spent preparing financial reports and free up time of skilled experts for more value-adding analysis.

This article first examines how financial reporting is performed manually and how it can be automated using an open-source Python toolkit. It then compares the pros and cons of both approaches before presenting conclusions.

THE GROWING COMPLEXITY OF THE REPORTING LANDSCAPE COMPOUNDS THESE CHALLENGES

Mr. J. Vijayan is a senior consultant in Finalyse's Insurance Practice, Fellow of the Institute and Faculty of Actuaries- UK (FIA) and an affiliate member of Koninklijk Actuarieel Genootschap.

TRADITIONAL TOOLS IN REPORTING

Manual processes in financial reporting present several challenges, including:

- Handling data from multiple sources
- High risk of human error
- Process inefficiencies and multiple iterations
- Extensive runtime

Gathering data from multiple sources and maintaining links between calculation files can be time-consuming and error-prone, compromising accuracy.

Report preparation using spreadsheet tools and manually copying content to text editors can become inefficient, diverting time away from strategic and value-adding analysis. Multiple manual steps and iterations to finalise reports further exacerbate inefficiencies, while the inability to perform real-time updates can delay decision-making.

EFFECT ON RESOURCES AND TIMELINE

The growing complexity of the reporting landscape compounds these challenges. Insurers must navigate regulatory requirements across various reporting bases while managing internal metrics such as economic capital and profitability. The increasing volume of reporting tasks strains resources and timelines, making manual processes unsustainable.

Simultaneous regulatory and internal reporting under tight deadlines adds pressure, increasing the risk of inconsistent results, missing deadlines or strained relationships with the regulator.

Automating reporting processes is essential to reduce costs, enhance efficiency, and maintain the integrity of financial reporting.

MODERN OPEN-SOURCE TOOLKITS

Popular open-source tools for automating financial reporting include Python and R. Both support end-to-end process automation from data imports and transformations to performing complex calculations and generating reports.

For instance, Python offers several libraries to automate all components of financial reporting including:

- Workflow optimisation with automated process mapping
- Data cleaning using pipelines
- Performing model calculations and sensitivity analysis with parallel computing
- Embedding tables and charts into reports
- Generating reports in HTML, PDF, or Word formats
- Using GenAI to produce report commentary

The remainder of this article focusses on the last three items in the list above.

Python is not only a powerful tool for data analysis but also a versatile platform for generating reports. With its rich ecosystem of libraries and packages, Python enables the creation of comprehensive reporting templates, seamlessly embedding tables, charts, visualisations and text.

Furthermore, Python can be used to generate reports in HTML, PDF and MS Word formats:

1. **HTML Reports**
   Python supports the creation of HTML reports via program code. HTML ensures accessibility of reports in the browser. Users have control over the design of their reports via program code and can embed tables, charts and text into their reports. The Jinja2 library enables the creation of reusable HTML reporting templates to offer greater adaptability and efficiency compared to using base Python code.

2. **PDF Reports**
   The FPDF library allows direct PDF creation, generating tables, charts and text and seamlessly convert these into PDF reports using templates. The PDF templates enable quick real-time adjustments, streamlining the reporting process.

3. **MS Word reports**
   The python-docx library can be used to create and modify Word reports programmatically. Users can add sections, tables, charts, headers or footers to their reports and apply formatting.

Irrespective of the output format, GenAI libraries can be used to automate commentary. For example, the OpenAI library allows embedding ChatGPT prompts into reports. However, the risks inherent in this approach must be carefully managed, especially in light of the requirements of the EU AI Act implemented on 1 August 2024.

COMPARISON OF PYTHON WITH TRADITIONAL TOOLS

As shown earlier, Python has powerful capabilities for the automation of financial reporting, but not all organisations are prepared to adopt such an open-source toolkit as they prefer the stick with their spreadsheets and text editors due to their wide availability, transparency and flexibility.

The pros and cons of adopting an automation toolkit for financial reporting using Python are summarised below.

It is clear that the benefits of using a Python toolkit are significant, and the associated challenges can be addressed through the following measures:

- A carefully designed training and implementation programme;
- Outsourcing the setup, maintenance and error handling to specialist providers;
- Low/no-code solutions, such as interactive apps (for example Dash) remove the need for programming knowledge.

CONCLUSIONS

Open-source tools such as Python offers a wide range of benefits for the automation of financial reporting:

- Significantly reduced runtime
- Lower risk of error
- Efficient handling of large datasets with scalability
- Reproducibility
- Automated creation of reports in various output formats (e.g. HTML, PDF, MS Word)
- Advanced features, such as professionally designed and reusable templates
- Flexible, open-source codebase that can be tailored to the evolving needs of the organisation and regulatory landscape

Still, the adoption of such open-source automation toolkits has been slow in the insurance and wider financial services industry. This is largely due to obstacles such as the steep learning curve associated with coding and the perceived lack of user-friendliness, flexibility and transparency.

These obstacles can be largely mitigated through training, outsourcing and implementing low/no-code solutions.
Embracing technological change calls for a shift in mindset, and buy-in from reporting teams is essential for the success of any transformation project. As the insurance industry continues to evolve, automation will no longer be optional but essential. With advancements in Python and emerging technologies like GenAI, financial reporting can be faster, smarter, and more accurate. Now is the time to lead this transformation. ■

PROS AND CONS OF ADOPTING A PYTHON AUTOMATION TOOLKIT

PROS	CONS
Automation: Python excels at automating repetitive tasks resulting in reduced runtime.	**Steep Learning Curve**: Python requires programming knowledge, which can be a barrier for those without a coding background.
Scalability: Python handles large datasets efficiently. Cloud solutions and parallel computing can add further scale and computing power.	**Setup and Maintenance**: Setting up the Python environment and managing dependencies can be complex.
Advanced Analytics: Python facilitates complex data analysis and insightful data visualisations.	**Error Handling**: Python scripts can be less forgiving with errors, requiring debugging and thorough testing to ensure accuracy.
Integration: Python can easily integrate with databases, web services, and other programming languages, providing flexibility in accessing and manipulating data.	**User Interface**: Python scripts may be perceived as less user-friendly and intuitive for non-technical users.
Reproducibility: Python scripts can be version-controlled using tools like Git, ensuring that workflows are reproducible and changes are trackable.	
Cost: As an open-source tool it is free to use, making it a cost-effective option.	