Paper 3111-2019

Best Practices for Business-Critical SAS® Applications

Ian Munro, Kiwibank Limited

ABSTRACT

SAS® is increasingly being used in many organizations for business-critical processes. The company I work for is a prime example. Kiwibank, New Zealand's fifth-largest bank, uses SAS for a wide range of risk-related processes, many of which are business-critical. These processes include the following: regulatory capital calculations on lending; account behavior scorecards; scorecard monitoring; IFRS 9 credit loss calculations; and loan-to-value ratio (LVR) lending restrictions monitoring.

Ensuring that all these applications keep going every day, producing reliable and accurate results, is no mean feat. It starts by architecting and implementing a resilient and robust SAS environment. This environment then needs to be carefully aligned and integrated with the company's business and IT infrastructure.

Next, disciplined operational practices are required, including essential housekeeping, to keep all systems running smoothly. And finally, there needs to be diligent management and checking of results to ensure accuracy. The consequences of getting them wrong could result in serious reputational damage and regulatory sanctions.

INTRODUCTION

When I first joined Kiwibank I was faced with a very rare challenge: a brand-new SAS server site and no legacy SAS applications to worry about. So that led to an opportunity to build best-practice SAS solutions from the ground up, leveraging off my knowledge and experience working at three other New Zealand banks.

The challenge was to construct a SAS application that would deliver regulatory capital calculations on all of the bank's lending. No pressure! This despite the fact it had never been attempted at Kiwibank before. The end result was an application called the Credit Risk Engine (CRE) that went on to do a lot more than originally intended.

It was during this journey that I came to realize that using SAS for business-critical purposes is a whole lot different to using it for other purposes such as internal management reporting or ad-hoc analysis. I needed to build something much more resilient, reliable and accurate. By taking a strategic and holistic approach to the design, implementation and operation of such systems I believed there would be a much higher chance of success.

This paper describes the practices which in my experience are essential to having successful business-critical SAS applications.

THE BANKING INDUSTRY IS DIFFERENT

For commercial organizations, the primary risk of doing business is running out of money and going bust. If you are a bank there is an extra risk. In many countries to be a bank you

need a banking license to operate. That license comes with strings attached. These are a set of rules a bank must abide by - without exception. A serious rule breach could result in the instant cancellation of a bank's license to operate. The banking regulator – in New Zealand this is the Reserve Bank – could shut us down tomorrow.

Kiwibank's business-critical SAS applications are continually calculating and monitoring many of metrics that underpin these banking license rules so there is a huge incentive to ensure that they don't contain significant errors. Many of these results must be sent to our regulator to confirm continuing compliance with our conditions of registration, or publicly disclosed to our stock exchange.

BUILDING RESILIENT SAS ENVIRONMENTS

Kiwibank's SAS servers have been designed to align with the banks disaster recovery (DR) standards. The primary SAS installation is mirrored at a physically remote secondary data center site. Data and applications are replicated from the primary to the secondary site every night. If there is a need to switch data centers, then that can be done in under an hour. Server alias names are used throughout all SAS software configuration and applications. These stay the same regardless of which data center is being used. To switch data centers only the server network addresses need to change.

REAL-LIFE DISASTER RECOVERY EXPERIENCE

November 14, 2016 is a date permanently etched into my memory. Earthquakes are frequent where I live in Wellington, New Zealand, being on the Pacific Ring of Fire. However, despite being nearly 300km from its epicenter, a 7.8 magnitude earthquake caused a lot of damage in our city.

Our main office building was declared off-limits while damage was assessed. That meant working from home until further notice. Would our SAS infrastructure continue to work reliably and our scheduled jobs continue to run while we had no access to the bank's network to check them? Obviously customer-facing computer systems took priority over our back office ones.

Gaining remote access from home via a secure Citrix internet connection after three days, I was relieved to find our SAS systems had continued to operate without problems in our absence. Our SAS server infrastructure was performing as expected. Our primary data center is located 600km north of Wellington and so was not affected by this disaster. There was no need to switch data centers.

However we were learning that while our SAS server infrastructure was up to the challenge, what happens when you have no office to work from and all your desktop PCs are stuck in an office building that is off-limits? After a week, my team was able to move into office space lent by one of our suppliers where we were up and running using borrowed computers. We continued in this mode for several months until we moved into new premises with replacement computers.

WHAT LESSONS WERE LEARNT?

You need to consider not only your data centers when building resilient SAS systems but also how you will access those systems and from where. Every link in the IT chain from data centers, company networks, network access, client computers and work accommodation need to be included.

Over two years after the earthquake, all bank staff have switched to using laptops which we take home every day. Now, as long as we have internet access, we can work from

anywhere. This means if there is a disaster, either major or minor, then we will be better prepared to cope. It also aligns with the bank's policy of encouraging working flexibly either at home or at the office. If we are away from the office we use secure VPN internet connections to access bank systems. The bank has also spread IT hardware and staff around more buildings and more widely spread locations to significantly improve business resilience.

We use SAS Enterprise Guide as our primary application development tool. SAS client tools like EG are installed on our laptops. We also use SAS Visual Analytics web-based interfaces for self-service reporting. These work equally well whether working remotely or from bank premises.

For SAS administrators we go one step further. Each administrator also has remote or local access to a hosted PC via secure Citrix internet connections. So that means if we have laptop problems or we don't have it with us we can still remotely access SAS from any internet-connected PC or laptop.

SEPARATE PRODUCTION AND DEVELOPMENT ENVIRONMENTS

Maintaining structurally separate production and development environments is another essential requirement for business-critical applications. In an ideal world this would mean having physically separate servers for production and development. However, given a limited software budget, Kiwibank decided instead to go with just the one set of SAS servers, metadata, application and mid-tier. On each of these servers, we run two separate instances of all SAS server processes. This maintains the required resource and application separation. Similarly, disk space is physically separated and only SAS administrators have the permissions to deploy and run production jobs.

SAS INFRASTRUCTURE SECURITY

Our SAS systems store a lot of confidential account and customer data. Hence there is appropriate security in place to prevent unauthorized access to both the systems and the data. This involves having several security layers:

- Company internally-hosted SAS systems only - all with firewall protection
- System and folder access is controlled by AD security groups and firewall port rules
- SAS data in-flight or at rest is appropriately secured
- SAS metadata folder permissions are also used particularly for SAS Visual Analytics
- Data exported outside the SAS environment needs to be appropriately secured including being anonymized if necessary

In addition, different security rules apply to our production and development areas. For example, write-access to production is limited to SAS administrators only. All other SAS users are limited to read-only access.

MAINTAINING DISCIPLINED OPERATING AND DEVELOPMENT PRACTICES

SERVER AND SOFTWARE MONITORING AND SUPPORT

In smaller organizations like Kiwibank, it isn't common to have a lot of SAS expertise within Information Technology departments. To solve this problem, IT supports just the hardware and operating systems of our SAS servers while the SAS administrators look after the SAS software environments.

If there is a SAS infrastructure problem the SAS administrators work together with IT to solve it. We find the SAS Environment Manager tool invaluable for checking server health. The dashboards provide a quick and easy way to check the SAS server software performance and to spot any problems.

Our servers are required to run continuously except for scheduled replication and maintenance outages. All our servers run on the Windows Server 2012 operating system. This means the servers are easy to maintain and straightforward to operate. Network connectivity and compatibility is also very good in an all-Windows environment. While scalability is not necessarily a Windows strong point, for small to medium SAS server installations like ours, the operating system is entirely suitable for our needs.

BUSINESS CONTINUITY PLANNING

BCP is an essential requirement for business-critical systems. We have already talked about this from a disaster recovery perspective but BCP is important in day-to-day situations as well. This applies not only to computer infrastructure it also applies to the people responsible for keeping the systems running.

Apart from running DR tests to ensure the secondary data center is working correctly, having staff work off-site regularly is another good way of ensuring business continuity when there is an office disruption.

To minimize key person risk, our team members regularly swap doing business-critical tasks so several people always know how to do these. This ensures the tasks will still get done regardless of who is working on any day. As we are a small team that is a continuing management and skill challenge but one we must practice constantly.

VERSION CONTROL

Historically, SAS developers have not been widespread or indeed enthusiastic users of version control tools. In my opinion however, version control is essential for business-critical SAS applications. Without it you cannot have certainty and reliability with your software deployments. Nor will you be able to easily trace the history of your software changes or know what code changes relate to a particular change request.

I cannot stress enough how important this is. Time and again we refer back through our code history where we can identify each and every change that was made. This functionality is essential for excellent software quality control and also to meet company and the regulator’s auditing standards.

Some SAS tools already integrate with versioning tools like SAS Enterprise Guide and Data Integration Studio. If your SAS applications are constructed with SAS programs, then these are text files that are compatible with virtually any version control tool.

My best advice is to use whatever version control tool your organization provides as the required infrastructure will already be available to you. We use Microsoft Visual Studio with Team Foundation Server as this is the bank’s officially-provided tool. It works a treat. Not only does it version control all our application artefacts it also provides project management capabilities making it easier to administer our software project workflows.

Visual Studio works really well with SAS Enterprise Guide. Opening a SAS program in Visual Studio automatically loads it into Enterprise Guide, ready for editing. Saving a program in Enterprise Guide directly saves it back to each user’s Visual Studio workspace. The days of forgetting where you stored a particular program are gone because you know it will be stored somewhere in your workspace.

Version control is not complete without automated software deployment. Visual Studio contains a graphical interface for software builds and we have custom build jobs to deploy our SAS applications automatically into production. Underneath the hood these are simply source code folder copies. That means we can have total confidence that the correct version of our software is always delivered to our production environment.

SCHEDULING AND PROCESSING

Another essential requirement for business-critical SAS applications is server-based scheduling of SAS jobs. There are many scheduling tools to choose from and it is worthwhile checking out those your organization uses. We chose the SAS-supplied LSF scheduler primarily because of its integration with SAS and the ability to easily manage most scheduling tasks through SAS Management Console. It is easy and efficient to either schedule or run SAS jobs immediately. One weakness of LSF is the requirement to use the separate Platform Process Manager tool for creating schedule events and monitoring running jobs. It would be great if all scheduling capabilities were available in one interface.

All of our production processing is done with batch jobs controlled by the LSF scheduler. All batch job logs are kept so we have a complete record of all processing. If there is a problem with a batch job an email is sent to our SAS administrator email account. That alerts the SAS administrator to go and check what has happened.

Personally, I'm not a fan of bogging SAS programs down with lots of error-checking code. In my view it is mostly wasted effort. We rely on simple system settings like syntax check mode. When this option is set, if SAS strikes an error it automatically stops processing any data and quickly runs through the rest the job. Being notified of any jobs that fail quickly means they get fixed quickly as well.

We have found it very useful to document processing schedules including having schedule checklists for all production tasks and who is responsible for doing them. Tasks are ticked off as they are completed so any team member can track progress.

SAS APPLICATION DEVELOPMENT

CRE is regarded as a core business-critical system at Kiwibank. Some key design decisions were made at the start which set it on the path to success:

- Use SAS coded programs only as this is best for version control and required for batch processing
- Leverage SAS's metadata-driven architecture to standardize data sourcing and storage
- Use only source system data where possible so there is one unmodified version of the truth
- Create a risk data mart so all risk-based metrics are stored in one place and can be used for multiple purposes
- Make extensive use of SAS macros (mostly stored in AUTOCALL libraries) for common tasks and extending to data-driven automated code creation reducing manual coding by around 20 percent and improving reliability

Making the right design decisions up front doesn't guarantee the delivery of successful business-critical applications but it does greatly reduce the chance of failure.

Coding standards become way more important with business-critical applications. Here are some recommendations:

- Write code that is readable and understandable by other SAS

- Layout code so it is easy to follow
- Avoid obscure or "clever" coding unless there is a clear benefit for using it
- SAS macros can reduce coding effort considerably, but don't use them where there is no benefit
- Data-driven SAS macros can greatly improve programmer productivity and code reliability
- Include liberal comments including a program header that documents changes and links to work requests. We regard these as mandatory to meet our governance and auditing standards.
- Deliver clean code – no log errors, warnings or notes that are warnings

SAS APPLICATION CHANGE MANAGEMENT

Having the right processes for managing application change is another essential requirement for business-critical systems. These include providing:

- Clear and complete software change requirements
- Concise documentation of the changes made
- Detailed evidence of testing the changes, including getting the expected results and not changing anything else unexpectedly. We cannot afford any changes having unintended consequences on regulatory outputs.
- Impact assessments on key metrics particularly regulatory ones
- Peer review and management signoff for critical changes
- The higher the importance of a change the more rigorous the testing and review process must be

There needs to be a high level of discipline around change processes. Change tasks must be defined, tracked and sufficiently tested. This isn't about creating an industry around application development. It is about striking a sensible balance around delivering enough documentation that confirms these software changes have been done correctly, have met the change requirements, and have been accepted as good to deploy to production.

DILIGENT ADMINISTRATION OF APPLICATION PROCESSES

No matter how good your SAS architecture or your operating and development practices are, it could still come unstuck if you don't properly manage the processes for your business-critical applications.

Recommendations include having:

- Checklists for running production processes, ticking them off when completed
- Documentation with instructions for doing non-trivial processes
- System administrators notified of any job failures. Email works well for us.
- Control reports for checking that processes have run correctly
- Exception reports for key inputs that need correcting
- Reconciliation steps where possible to compare results with other systems

CRE is probably typical of many business-critical SAS applications. It contains a mixture of manual and automated steps:

- Automatically-scheduled and manually-run batch programs
- Manual setup and input steps
- Manual checking of control reports
- Manual checking of exception reports and updating manual inputs
- Automated updating of self-service SAS Visual Analytics reports
- Automated updates of other systems
- Reconciling key metrics
- Distributing manual outputs

While complete process automation is the ideal goal, the reality is that manual processes remain a necessary evil. Kiwibank's primary production schedule runs monthly, from the end of each month. The schedule is mostly complete by the fifth business day despite being constrained by some inputs not arriving until the second business day. Our outputs are required by other bank systems and the deadlines are tight. This is where good administrative practices really pay off.

CONCLUSION

Building and supporting business-critical SAS applications requires a holistic approach that should start before SAS software is even installed. Decisions are required during the design, build, and support phases to provide resilient infrastructure and robust administration.

The same applies to the SAS application design, build, test, implementation and administration phases. Decisions that enhance the system reliability and accuracy are essential. Kiwibank's Credit Risk Engine is a good case study of the benefits of this holistic approach.

ACKNOWLEDGMENTS

I would like to acknowledge the help and encouragement of my fellow Kiwibank colleagues Richard Boodee and Dave Munro for their valuable help in preparing this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Ian Munro

Kiwibank Limited

ian.munro@kiwibank.co.nz