EDA IN THE CLOUD: CONTAINERIZATION, MIGRATION, AND SYSTEM TELEMETRY

Dr. Rosemary Francis – Chief Scientist, Altair | Derek Magill – Executive Director, HPC Pros / November 17, 2020

New and exciting IT infrastructure technologies are widely gaining adoption outside EDA. This paper focuses on various core capabilities needed by EDA companies in order to begin to embrace these new technologies. It was originally developed by Ellexus with the presenters and organizers of the Storage Profiling, Containerization and Cloud for EDA workshop at the 2019 Design Automation Conference in Las Vegas, organized by HPC Pros.

Cloud computing is becoming an increasingly good choice for EDA, but a data-led transformation is needed to take advantage of the flexibility that public compute resources can offer and to maximize performance and cost. Cloud allows an organization to benefit from a clear return on investment that supports innovation and rapid prototyping, provided those advantages are fully exploited. The wins achievable by a well-planned hybrid cloud strategy should see reduced costs both on-premises and in the cloud. The ability to dynamically tune compute and storage resources based on business and application needs is only available in the cloud, and only if the right telemetry and data pipelines are in place to inform infrastructure decisions.

In this paper we discuss how to put that plan in place and ensure that key business objectives are met as workflows are adapted and migrated to a new compute environment.

Why Move to the Cloud?

Today, many organizations are looking to establish a hybrid cloud system. The main reasons are flexibility and scalability. On-premises resources do not scale infinitely, particularly with large, single-namespace file systems.

Hybrid cloud requires a change in working practices. For example, rather than increasing hardware to solve problems, an organization can use on-premises storage for known workloads while sending unexpected, experimental, or problem workloads to the cloud. Workloads with known bad I/O patterns or other resource-hogging execution patterns can be isolated in dedicated ephemeral clusters while predicable workloads are run with maximum efficiency on-premises.

Cost efficiency – In the past the efficiency of high-throughput on-premise resources has made the cloud prohibitively expensive. However, the efficiency of on-premises resources relies on a one-size-fits-all approach. As scaling problems and increased complexity harm the cost effectiveness of homogeneous on-premises systems, the cloud becomes more effective. The higher costs of a managed service can be traded off against rightsizing for cloud-based jobs. If on-premises resources are only used for predicable workloads, they too can be scaled back as there will no longer be a need to buy in resources for unpredictable compute and storage needs.

Emerging technology – Cloud vendors have access to the latest technologies. “Try before you buy” is a great way to stay ahead of the curve while making sure that purchasing decisions for hardware are well informed.

Agility – Moving to hybrid cloud can help a team move from reactive waterfall development to agile methodologies. The reactive effort of firefighting a shared cluster with unpredictable loads and problems can instead be put into a more proactive experiment-test-deploy cycle. While most teams are currently deploying workloads on-premises and moving them to the cloud when ready, ultimately new workloads should be developed in the cloud and moved on-premises when the I/O and compute patterns are known and stable.

This move to an agile way of working is possibly the biggest single change that a hybrid cloud environment can enable. Arm employs a methodology called Will it make the boat go faster?™, as explained by Vicki Mitchell, vice president of the Technology Services group at Arm, at the 2019 IEEE Women in Engineering International Leadership Conference. This technique looks at measuring efforts against measurable business goals such as cost and application throughput. Hybrid cloud makes this possible and transparent by clearly linking engineering effort, cost, and throughput.

Getting Started With Good Data Hygiene

Data hygiene is about understanding the way applications use data:

- Knowing application dependencies
- Understanding data provenance
- Managing the data lifecycle and tiering

Understanding application dependencies is important for application portability, efficiency, reproducibility, and correctness. EDA tools are often run inside complex scripted workflows that introduce many external dependencies and potentially bad I/O patterns. Taking control of what a workflow accesses is the first step towards taking control of the workflow.

Understanding File, Application, and License Dependencies With Breeze

Breeze is used throughout the EDA industry by EDA vendors and semiconductor manufacturers to discover application dependencies and profile I/O patterns. Breeze traces each script, file, and license server that is accessed as part of each workflow, while collecting arguments and the environment for verifying the correctness of a deployment or for triaging any runtime issues.

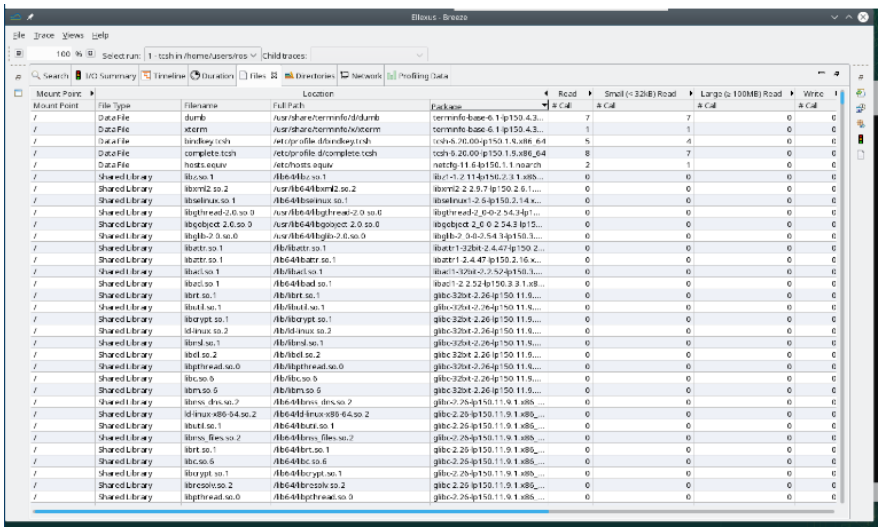
> *"Hard-coded paths caused a massive grid slowdown where jobs were taking two or three times as long to run. Using Breeze, we were able to not only quickly confirm the suspicion of a hard-coded path, but also later check to confirm that the problem was actually handled. The initial fix reduced the grid load but did not fully fix the hard-coded path. We were then able to use Breeze to continue to audit and check until the hard-coded path was fully removed."*
>
> —Jenson Ho, developer support engineer at Mentor Graphics

Application dependencies should include the following as a minimum to ensure application correctness and portability with particular care given to versioning:

- File dependencies
- Binaries and tool libraries
- System libraries
- System packages
- Workflow scripts and makefiles
- Environment and argument settings
- EDA licenses

Case Study: Application Dependencies as a Service at Qualcomm

At Qualcomm, Breeze has been deployed as a service so that users can opt in to collecting dependency data about their workflows. This is called the Bill-of-Materials (BoM) workflow and was originally designed to help users migrate workflows between on-premises clusters. It has been used extensively in the effort to prepare and migrate workflows to a hybrid cloud environment.



File and mount point dependencies from a Synopsys workflow at Qualcomm (obfuscated)

Users submit thousands of workflows at a time to the BoM framework via a workload management queue. The BoM framework is implemented as a set of Python wrappers built around the Breeze command-line APIs and is run as a job is started to capture and wrap every job that is submitted to the Breeze BoM queue. This is a neat and easy way to run Breeze as a service and remove the need for user training.

Breeze traces each of the workflows and produces both a list of every file that has been accessed and a summary of each mountpoint that was used. The process of extracting the list of dependencies and mountpoints is a post-run step that is performed on a set of job runs so that results can be aggregated. It is common for multiple runs of a single application to have slightly different dependencies, so more than one run should be used to ensure that all possible dependencies have been captured.

The Python wrapper prepares a directory in which to store the trace results and then runs the job with the Breeze wrapper on the assigned compute node. Once the top-level job and all child jobs have completed, the job trace is post-processed and the dependencies extracted.

While it is not necessary to wrap up Breeze in this way, it simplifies dependency collection within Qualcomm's complex job environment and it means that system users do not have to be familiar with Breeze in order to access dependencies.

> *"We have a variety of users who will run a variety of tools under Breeze BoMgen, especially moving stuff to the cloud. Without Breeze it would be really hard to identify which of our thousands of mounts points a tool is accessing."*

By taking control of the files, programs, and mountpoints needed to run each workflow it becomes easy to move workflows between on-premises clusters and to migrate applications to the cloud. In an industry where compliance is important, capturing the exact inputs to and components of an EDA workflow is critical to ensuring that legacy tools can be maintained correctly.

Containerization

Once workflow dependencies are known, the workflow can be containerized. Containerization offers the opportunity to encapsulate a workflow with obvious benefits for portability and reproducibility. Containers offer many of the benefits of virtualization, but with lower overhead, making it easier to compose the data-driven workflows that take advantage of heterogeneity in the cloud.

Singularity from Sylabs

While Docker has become the industry default for microservices, there are other containerization technologies available for large scientific applications such as EDA flows. Singularity is one such container technology that was developed at Lawrence Berkeley National Laboratory and is now owned and maintained by Sylabs.

EDA tools often have very large deployments, but that is not a problem for containers. Large install bases will result in large containers, but the on-disk footprint will not be significantly larger than the bare-metal installation. Like virtual machines, containers are not loaded into memory in full when run so containers shouldn't have higher memory requirements than tools installed on bare metal. EDA datasets and libraries also have very strict licensing and access restrictions, but again this is not a problem for containers.

Singularity containers have been built with security and multi-tenancy in mind. They use the Singularity Image Format (SIF), which is a single file that can be digitally signed to ensure that trusted containers cannot be tampered with. SIF is therefore an ideal format for moving applications and data between on-premises systems and the cloud. With the ecosystem of container management growing daily, it is increasingly attractive to containerize applications for hybrid cloud for easy and reliable deployment.

System Telemetry for Hybrid Cloud

While hybrid cloud offers a lot of potential when it comes to aligning business goals with engineering efforts, it is necessary to have good telemetry in place to take advantage of that. Without knowledge of the IT infrastructure today, it is not possible to plan for tomorrow. Application steering, right sizing, and cost efficiency can only work if there are metrics to tune on and a framework for adjustment.

Mistral is a scalable monitoring solution designed for high-performance and high-throughput workloads such as EDA workflows. It was designed with the IT infrastructure department at Arm to provide always-on, system- and storage-agnostic application performance monitoring (APM) and profiling. In an on-premises environment, the telemetry from Mistral makes it easy to debug and triage operational difficulties such as noisy neighbor jobs that overload shared storage.

> *"The data and system control provided by Mistral allows the infrastructure teams to prevent risky I/O patterns and gives us a lot more information to learn from."*
>
> —Olly Stephens, Engineering Systems Architect at Arm

In a cloud environment, Mistral can offer much more than operational telemetry. By using Mistral as a data source for a machine learning pipeline, it is possible to deploy an intelligent, adaptable compute environment that not only provides the tuning and steering information needed to take advantage of cloud, but also the business intelligence needed to predict and plan for the future.

Data That Pays for Itself

System telemetry always has a trade-off between quantity and value. The Mistral and Breeze tools allow tuning on the volume and granularity of data collected. It is not uncommon to collect large volumes of data for workflows under development and to collect much lighter statistics for the workflows in production. By feeding the data into multiple pipelines it is possible to maximize the business wins:

- Per-job dashboards for workflow development and optimization
- System overview dashboards for operations and triage
- Data pipeline for business intelligence, tuning, and forecasting

By collecting metrics on workflow resource needs (CPU, memory, I/O, etc.), on system performance and on higher level measurements such as costs and time to market, it is possible to affect measurable business wins for relatively low investment up front.

Per-job Metrics and Happy Hosts

The Mistral and Breeze tools specialize in collected per-job metrics. They come with a wrapper that collects performance and I/O data for each job with controllable rates and granularity. The advantage of this approach is that the data is system- and storage-agnostic. The tools run in the same user space as the job so can collect other happy host and system metrics to augment per-job data and add context to the measurements. Typically, the wrappers are installed and managed by the scheduler or container orchestration framework so users do not need to know if or how the metrics are collected.

Telemetry for Workflow Development, Steering, and Operations

Developing dashboards for workflow developers and for operations has a lot of overlap, so there is no reason why two systems can't be unified. System administrators and developers both want access to the data instantly for rapid decision-making, so real-time dashboarding solutions for time-series data are well suited. ELK stack and Grafana are popular and increasingly rich environments for handling this type of data, but there are other dashboards and databases. Time-series databases are popular for managing system telemetry, but for richer data it might be better to choose a relational database that can handle time-series data.

Normal I/O: Mistral data tracking per-job metadata, bandwidth, CPU, and memory

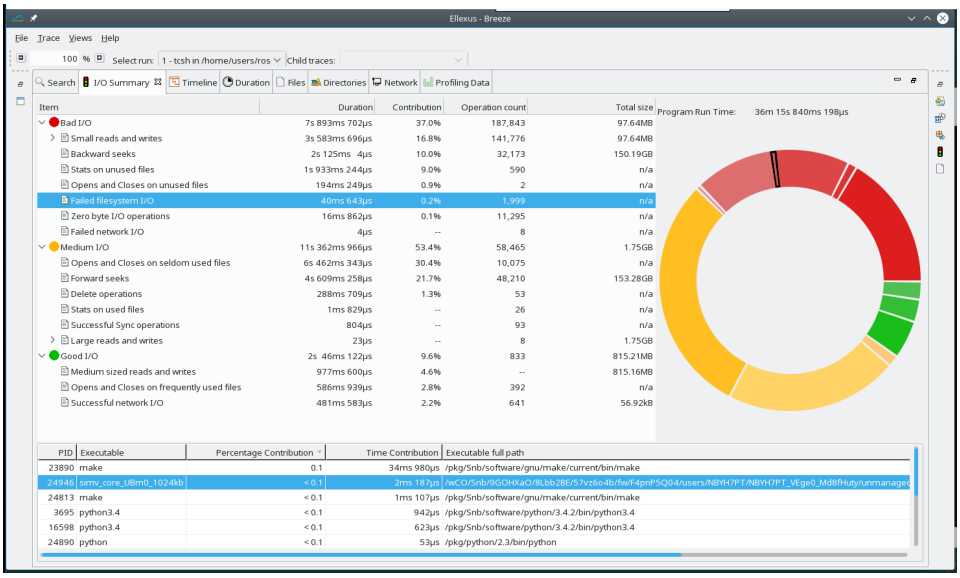
Knowing what is normal is important for improving workflows and operations and can be built into a good dashboard. This is also a good way to communicate to IT users what is good application behavior and what could be improved.

Mistral and Breeze allow monitoring of high-level metrics such as the amount of time spent in good and bad I/O as well as raw data such as metadata operations, bandwidth, and storage performance. By building these metrics into a service such as the Breeze Bill of Materials at Qualcomm, users can more easily understand the impact of good workflow design on on-premises resources and cloud costs. User education with a framework of objective feedback is critical when designing workflows for a hybrid environment because workflows may be run in an environment that differs significantly from the development environment due to the dynamic nature of the cloud.

Job steering is one of many challenges in the cloud: How do you know which workloads to run on-premises and which to run in the cloud? By getting a good handle on the resource needs of each workflow it is possible to start with some back-of-the-envelope calculations for cloud cost forecasting and then improve from there.

For organizations with a large, stable workload it is likely to be more cost-effective to run that on-premises. Noisy neighbors and workloads with unpredictable or bad I/O patterns and resource needs harm the performance of shared clusters so make good candidates for cloud deployment, but there will be other factors to take into account as well such as data access, tape-out deadlines, and burst capacity needs.

It is easy to see how informal reasoning on job needs is a good start but quickly becomes inadequate when juggling the needs of the business against the many options in a hybrid cloud environment. That is why evolution of the data pipeline should have a roadmap.



Good, bad, and medium I/O in EDA workflow

Business Intelligence and Machine Learning

Data-driven transformation needs a powerful decision-making data pipeline to collect the best metrics. This could be a machine learning pipeline, but it doesn't have to go to the lengths of a self-driving car to deliver the benefits of tuning and forecasting for high-level business goals. The potential of using system telemetry for higher-level analytics should be built into the telemetry framework from the beginning, but a machine learning pipeline doesn't need to be the first step when migrating to the cloud.

Once a good corpus of data has been collected it is possible to run thought experiments for tuning and steering. For example, once you know the capacity of on-premises resources and the cost of running each workload in the cloud, it becomes possible to take the workloads run in a day or week and simulate cost and capacity experiments with different combinations of on-premises and cloud divisions. It may be more efficient to move some workloads back on-premises and displace others to the cloud.

EDA Cloud Migration Overview

Moving to a hybrid environment without interrupting services on-premises is complex, but not impossible. By following these best practices, it is possible to use data-led system design to improve on-premises infrastructure at the same time as informing hybrid cloud deployments.

1. Implement good data hygiene through dependency tracking and containerization to prepare for migration.
2. Put in place resource tracking and system telemetry to identify good candidates for migration and to ensure good resource sizing with early cloud deployments.
3. Build system telemetry into your cloud orchestration framework from the beginning for resource tuning and basic forecasting of costs.
4. Start with operational dashboards combined with hand-tuning and steering given a limited granularity of job types and cloud architectures.
5. Evolve the analytics pipelines to automate cloud steering and right-sizing of resources for more finely tuned and diverse cloud architectures incorporating high-level business metrics.

