Irion®
creating data advantage

Giovanni Scavino

How to reduce Data Management projects duration and costs

Irion EDM In-Depth

About the author

Giovanni Scavino

As a co-founder, Giovanni had the idea of creating the Irion EDM platform.

Upon completing his degree in electronic engineering specialized in Computer Science, aged just 22, Giovanni together with his brother Alberto founded their first software company, Dianos. He already had a successful career in financial software development (front and middle office systems, Risk & Performance Management, Master Data Management) when he decided, in 2004, to found, again together with his brother and with Mauro Sturaro,a new initiative. He took advantage of the significant experience in Enterprise Data Management, Data Quality, process organization, product view, and enterprise strategic positioning.

Giovanni focused on implementing the Irion EDM platform. The aim is to always be able to respond in advance to the market needs and the most urgent priorities and act accordingly.

His experience and expertise have guaranteed Irion's rapid growth. The company has successfully established itself at both local and international levels.

Giovanni Scavino
Irion Chairman of the Board

> With data collection,
> 'The sooner the better'
> is always the best answer.

Marissa Mayer

How to reduce Data Management projects duration and costs

Introduction

In the complex and varied world of performance measurement of the processing of a software system, it is essential, first and foremost, to clarify and agree upon the performance indicators to refer to.

The two main KPIs are **Response Time**, the time it takes a specific process to deliver the outputs after the inputs are made available, and **System Throughput**, the amount of work that a processing system (hw+sw) can perform in a unit of time.

It is evident that both indicators are interesting to monitor and are related.

However, in Enterprise Data Management, the operations are almost always **data-intensive** and very often of **massive and analytical** (OLAP) rather than "atomic and transactional" (OLTP) type. Therefore, it is necessary to optimize the amount of the system's work depending on the hardware (typically CPU, RAM, I/O) provided.

Consider a concrete example. In a data quality control system, whether the data is fed or already present in a Data Lake or a data warehouse, the key point is the number of controls the system can execute in a unit of time and not the time needed to execute just a single control. Or, even more pragmatically, what is important is the total time it takes to run all the necessary controls to guarantee data quality after every update cycle.

Thus, it is clear that the most relevant reference parameter to monitor is the overall System Throughput. It is also worth noting that modern EDM systems are configured with a complex architecture, often multi-tier and multilayer, scalable both horizontally and vertically.

They contain services and processes that implement both automatic and interconnected with "human tasks" operational workflows. Besides, they deal with significant volumes of data, often in heterogeneous formats.

In such scenarios, many variables and choices affect the performance, and often significantly.

Irion's Enterprise Data Management technology is not born in a laboratory. It is not an abstract product of some academic or laboratory reasoning. We have learned a lot from years of experience in mission-critical and data-intensive contexts. Our platform is developed based on **hundreds of projects**: those with lots of data, with time constraints, with specifications that change in the course of work, with complex problems in contexts highly articulated in terms of architecture and organization.

Below is the analysis of some of Irion EDM's particular features, the ones that make it stand out among other offers on the market and explain the **exceptional performances** achieved in various real-life scenarios. We shall pay particular attention to the detailed and specific description of the implemented processing model. Other focus points will be the scaling possibilities and the massive parallel use of resources that the architecture makes possible.

Logical and conceptual setup of executions

The main features contributing to the system performance are related to how the executions are set up logically and conceptually. In essence, as you can easily guess, the low-level logical-applicational and technical design of how the processing is performed is the one that most significantly affects the efficiency of the system. From this point of view, **our approach is disruptive**. Irion EDM is not yet another procedural ETL system with some extra function, some specific connector, a better editor, and magic engines for fast execution.

ETL processes are a variation of Data Management, which covers many other areas (Rule-Based Systems, Data Quality, Data Integration and Reporting, Data Governance, etc). We have redefined the concept of Data Management, inspired by the declarative concept that was historically applied to a category of programming languages. We transformed this concept into an original paradigm for the entire sphere of Data Management practices, making it universal for our technology and methods.

One of our core technologies is DELT™. This acronym stands for Declarative Extraction Loading & Transformation and means exactly that. In addition to inverting the phases of the Loading and Transformation (ELT compared to ETL), already widely discussed in the literature, we make sure that the whole process is Declarative, that it is described and performed according to this concept.

As a brief reminder, by the **declarative approach** we understand that it is not necessary to specify how to do the task, but it is enough to declare what should be done!

For an in-depth analysis and comparison with the procedural model typical of the traditional ETL approach, refer to the article Declarative Thinking.

Our architecture and our declarative approach to processing base on many aspects. The most relevant ones, especially in terms of the performance, are the following:

- Each dataset used in the processing is **re-displayed virtually as if it were a table** (or a set of tables) in the relational model by the technique we call **EasT**® (Everything as a Table). The platform runs all the necessary transformations implicitly to appropriately map the data available in any format (CSV, Excel, XML, Cobol, DB, Web Services, API, SAP, etc.). The temporary virtual EasT® tables defined during the processing are **immutable objects**. They are initialized only once, and their contents cannot be modified once they are created. This characteristic guarantees the system is consistent and controllable. Besides, normalizing everything according to a relational model allows adopting and leveraging Data Management engines and techniques matured and optimized over the years.

- The processing (transformation, control, aggregation, analysis, etc.) is **broken down into a set of Data Engines**. That is, the engines that receive the input of one or n virtual tables, perform the appropriate transformations/verifications, and produce the output of one or m virtual tables, according to the EasT® model.

- **Data Engines can be of various types** (Query, Script, Rule, R, Python, Masking, Profiling, etc.). They run simple or extremely complex processing as configured by whoever sets up the solution. Depending on the task to perform, they encapsulate various models and kinds of logic and use the best and most efficient languages and technologies. If necessary, each engine (e.g., R, Python, etc.) can use external computational libraries and/or engage remote processing.

- **The configuration of Data Engines**, access to other EasT® objects, and the production of virtual tables in output **results in implicitly defining an oriented graph** of execution dependencies. Therefore, the system can autonomously organize the steps necessary to produce each output. The designer does not have to describe the algorithm.

- The various virtual tables produced as an intermediate processing step **can optionally be materialized**, providing intermediate data for further (re)use. The process then breaks down into partial, more efficient steps, and the system gets correct estimations on the cardinality and statistical distribution of data, optimizing the subsequent processing. If necessary, it is possible to activate advanced indexing techniques, both row-based and columnar, as well as data compression algorithms to optimize subsequent access.

- **The DELT™ processing follows a Goal-Driven and Backward Chaining declarative scheme**. It specifies the datasets expected in the output. Then it is the task of the system to automatically detect the DELT™ Graph, verifying that the graph is acyclic, and initiate the optimized and parallel execution of all the steps necessary to produce the required result. The DELT™ Graph resolution and crossing algorithm uses several heuristics and optimization rules. It can thus define the optimal order and level of concurrency to initiate the nodes of the graph.

- In very complex cases, the engine **can also actively modify the graph** during execution according to the Dynamic Properties that determine it (they can also be influenced by previous steps). It can even generate and execute at runtime the Data Engines necessary for specific computations configured and optimized according to the parameters and data involved in the execution (VEG Virtual Engine Generator).

- If necessary, a DELT™ Execution **can engage other DELT™ Executions** for encapsulation, optimization, and reusability of the business logic.

- The proprietary technology for making the data involved (EasT® objects and all the necessary temporary data) available to DELT™ Executions in an isolated, managed space is called **IsolData**™. It is a kind of a **sandbox**: a virtually unlimited workspace, dynamically allocated and freed by the system, segregated in terms of access permissions and namespace, volatile or persistent.

- **The IsolData™ allows running n DELT™ in parallel**, on the same or completely different data, and/or with different parameters, rules, and logic. There is no need to predefine and manage any particular structure (e.g., there is no need to define and remove temporary tables, spaces, scripts, etc.). The framework coordinates and manages all the support operations for the infrastructural management in an optimized and transparent to the user way. Thanks to IsolData™, the system minimizes the contention between various parallel processes to make the most of the hardware resources available. The temporary tables contained in an IsolData™ are immutable objects. They are initialized only once and read several times with no possibility to modify them. Hence these datasets are accessed in the NoLock mode, minimizing the Log. The system "housekeeping" activities are performed centrally and asynchronously so as not to impact the response times of the processes.

- During a DELT™ execution, **all data processing operations are performed at a low level on the server** and in the set-oriented mode, i.e., considering an arbitrarily complex set of datasets simultaneously and not one line at a time. In short, it is much more efficient to move the rules rather than the data, to engineer the process and minimize the operations the system performs.

- The declarative model allows the DELT™ engine to **choose the optimal algorithm and level of parallelism for data processing automatically and at runtime** based on the up-to-date data statistics.

- Conversely, a procedural model restricts to anticipate all the steps and possible parallelism design-time, starting from the distribution of data that appear during various executions.

- The system's control and transformation rule engine **can dynamically generate the instructions** necessary for parallel low-level execution of all the rules relating to one dataset. This way, it avoids reading and analyzing the same dataset, again and again.

- The server **minimizes the onerous data movement tasks** and the possible complex processing that produces intermediate data for reusing multiple times (partial aggregates, lookups, etc.). They are executed only once, with no need for explicitly saving the intermediate results in files or supporting tables.

Most data processing occurs server-to-server. There is no need to transfer the data to a client to process and then rewrite it on the server itself.

However, in case of a necessity to run the processing client-side, pipelining techniques with cycle buffering in memory and bulk-load in streaming minimize the processing and waiting times.

If necessary, part of the processing can be done directly at the data source (push-down techniques) to minimize data transfer (e.g., filters, pre-aggregations, etc.).

In certain situations, the system may also transparently use the local cache of data to minimize the access cost and decouple the data sources. This way, it cancels the impact on external systems and overcomes possible time restrictions on data availability.

- Knowing a set of statistics on the data allows the engine to **perform specific optimizations while running the engines**. For example, in the case of single-row tables, the system can use the current record values to make better use of the information available on data distribution and choose the data-reading strategies to adopt (i.e., Scan vs. Seek). Conversely, in the case of tables with a significant number of rows, the optimization engine tries to use query resolution operators in batch-mode. This way, each operator, even very low-level, works simultaneously on the dataset thanks to special vector operations the CPU provides.

In certain cases, adopting these techniques helps to achieve performances of a different order of magnitude.

Conclusion

The declarative Enterprise Data Management makes the system's overall Throughput beyond comparison with the traditional ETL-based approaches. This is thanks to the fine-tuned algorithms and techniques, horizontal and vertical scalability levels, efficient use of resources, and cost-based orchestration and optimization of parallelism levels.

Irion

Irion is the Italian company that has developed Irion EDM, an all-in-one platform for Enterprise Data Management.

The platform is at the core of our offer to support innovation and our customers' businesses. It is complemented by RTG (ready-to-go add-ons). These are accelerators designed for faster implementation of data-intensive business solutions. A rich set of professional services completes the value proposition.

Irion EDM is based on the declarative paradigm and metadata-driven approach. This is a generational leap from traditional data management systems.

Thanks to the declarative paradigm and metadata-driven approach, one can specify what needs to be done without specifying how to do it. They help to benefit from data and manage the organization's digital evolution in an agile and flexible way.

We have been on the market for over 16 years and have in-depth knowledge of business needs. That's why we have equipped the all-in-one Irion EDM platform with unique functions. In addition to traditional data processing, it supports more innovative practices and methodologies for data-intensive solutions (Agile, DataOps). It also ensures compliance with the norms and regulations that are more and more oriented towards risk assessment and data assets governance.

Irion EDM

The all-in-one declarative and Data Fabric ready platform to extract value from your data.

LEARN MORE

Irion®
creating data advantage

GO BEYOND, BE DECLARATIVE.

www.irion-edm.com