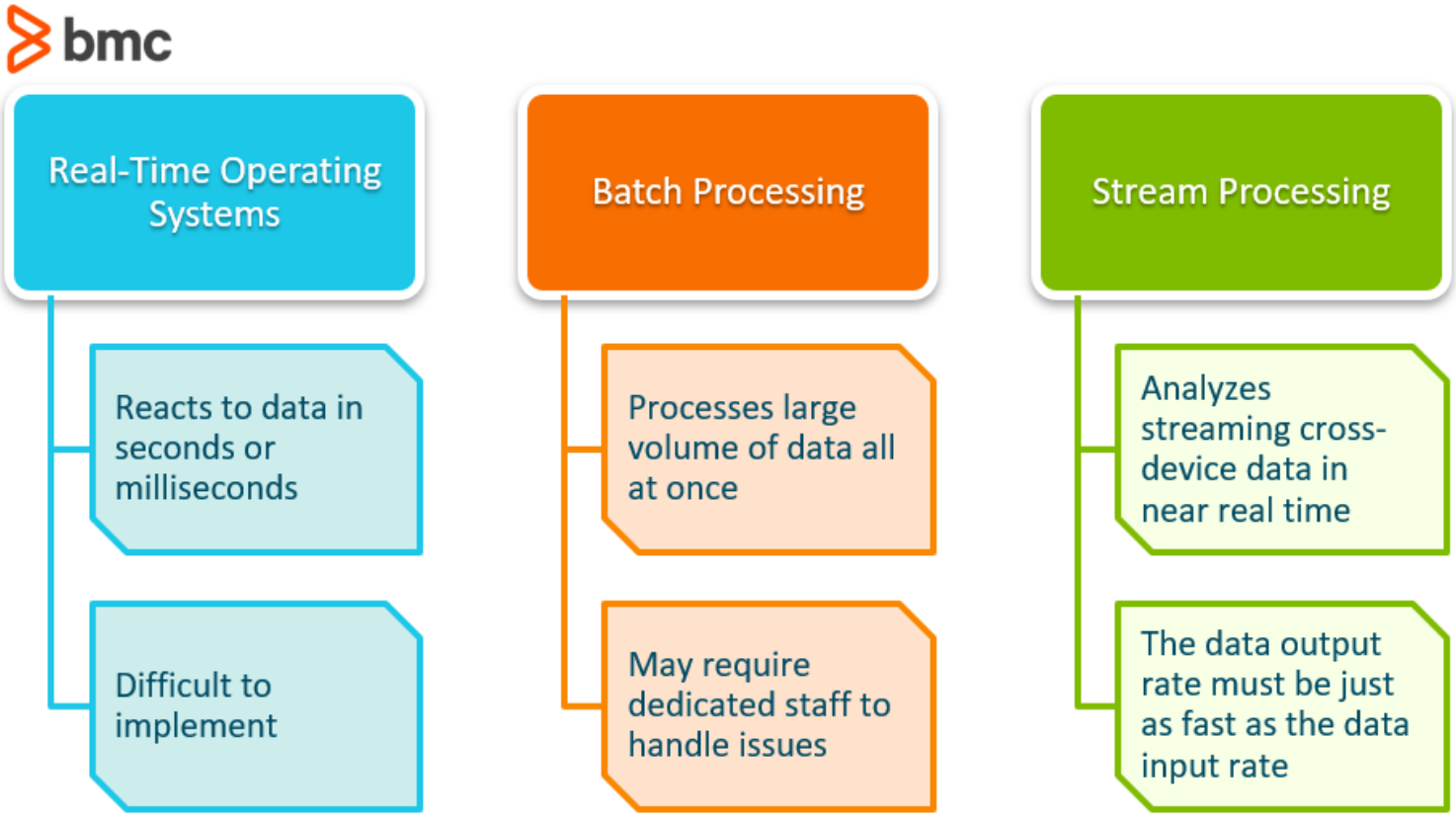
REAL TIME VS. BATCH PROCESSING VS. STREAM PROCESSING

With the constant rate of innovation, developers can expect to analyze terabytes and even petabytes of data in any given period of time. (Data, after all, attracts more data.)

This allows numerous advantages, of course. But what to do with all this data? It can be difficult to know the best way to accelerate and speed up these technologies, especially when reactions must occur quickly.

For digital-first companies, a growing question has become how best to use real-time processing, batch processing, and stream processing. This post will explain the basic differences between these data processing types.



Benefits & drawbacks of common data processing types

Real time data processing and operating systems

Real-time operating systems typically refer to the *reactions* to data. A system can be categorized as real-time if it can guarantee that the reaction will be within a tight real-world deadline, usually in a matter of seconds or milliseconds.

One of the best examples of a real-time system are those used in the stock market. If a stock quote should come from the network within 10 milliseconds of being placed, this would be considered a real-time process. Whether this was achieved by using a software architecture that utilized stream processing or just processing in hardware is irrelevant; the guarantee of the tight deadline is what makes it real-time.

Challenges with real-time operating systems

While this type of system sounds like a game changer, the reality is that real-time systems are extremely hard to implement through the use of common software systems. As these systems take control over the program execution, it brings an entirely new level of abstraction.

What this means is that the distinction between the control-flow of your program and the source code is no longer apparent because the real-time system chooses which task to execute at that moment. This is beneficial, as it allows for higher productivity using higher abstraction and can make it easier to design complex systems, but it means less control overall, which can be difficult to debug and validate.

Another common challenge with real-time operating systems is that the tasks are not isolated entities. The system decides which to schedule and sends out higher priority tasks before lower

priority ones, thereby delaying their execution until all the higher priority tasks are completed.

More and more, some software systems are starting to go for a flavor of real-time processing where the deadline is not such an absolute as it is a probability. Known as soft real-time systems, they are able to usually or generally meet their deadline, although performance will begin to degrade if too many deadlines are missed.

Real-time processing use cases

When you are continually inputting and processing data, and handling a steady data output stream, you need real-time processing. Here are some real world situations where real-time processing is necessary.

- **Automated Teller Machines (ATMs):** To improve the customer experience, boost back office efficiency and analytics capabilities, and reduce fraud, banks are adopting real-time processing. Processing ATM transactions, immediately posting transactions to the account, and adjusting the balance in real-time requires secure and efficient user credential validation, account balance checks, and on-the-spot transaction authorization.
- **Air traffic control:** Safely managing and moving aircraft in a crowded space requires using data from multiple sources, such as radar, satellite imagery, sensor networks, and aircraft communications. Advanced technologies like AI require real-time data to provide up-to-the-second situational awareness, advance warning of potential conflicts, and the ability to optimize flight paths and manage air space. The ideal is being able to forecast proactive decisions that avoid collisions and minimize congestion.
- **Anti-lock braking systems (ABS):** The vehicle safety advantage of automated and optimized braking requires real-time data. Information about tire-to-road conditions have to be immediate. The second wheel slip is detected, the system regulates dynamic braking force. Speed on straightaways, curves, and angles, along with sensitivity to road surface and conditions like moisture, ice, and oil are all data points to which the system must instantly detect, analyze, and respond to prevent wheel lock-up, maintain vehicle control, and reduce stopping distances in emergency braking situations.

Batch Processing

Batch processing is the processing of a large volume of data all at once. The data easily consists of millions of records for a day and can be stored in a variety of ways (file, record, etc). The jobs are typically completed simultaneously in non-stop, sequential order.

A go-to example of a batch data processing job is all of the transactions a financial firm might submit over the course of a week. Batch data processing is an extremely efficient way to process large amounts of data that is collected over a period of time. It also helps to reduce the operational costs that businesses might spend on labor as it doesn't require specialized data entry clerks to support its functioning. It can be used offline and gives managers complete control as to when to start the processing, whether it be overnight or at the end of a week or pay period.

Challenges with utilizing batch processing

As with anything, there are a few disadvantages to utilizing batch processing software. One of the biggest issues that businesses see is that debugging these systems can be tricky. If you don't have a

dedicated IT team or professional, trying to fix the system when an error occurs could be detrimental, causing the need for an outside consultant to assist.

Another problem with batch processing is that companies usually implement it to save money, but the software and training requires a decent amount of expenses in the beginning. Managers will need to be trained to understand:

- How to schedule a batch
- What triggers them
- What certain notifications mean

(Learn more about modern batch processing.)

Batch data processing use cases

For less time-sensitive jobs, batch processing can be an efficient option. When it doesn't matter that processing could take hours or even days, you don't need real-time or near-real-time options.

- **End-of-day reporting:** Financial institutions typically run end-of-day reports that include bank ledger data, such as starting balance, deposits, withdrawals, and transfers, culminating in an ending balance for the business day. The report supports accuracy by flagging errors and ensures system integrity. It helps managers improve and maintain operational efficiency and provides an audit trail required by law. Activities occur all day, but the processing of the report and analysis is done at a single time after the bank day ends.
- **Data warehousing:** Data in data warehouses is typically managed in scheduled and regular extract, transform, and load (ETL) batch processes. Periodic updates are an efficient way to handle large volumes of data, while ensuring that the data warehouse is kept up-to-date with the latest information, for analytical purposes.
- **Payroll processing:** Company payrolls are usually done in a regular cadence, typically bi-weekly or monthly. Using batch processing streamlines and speeds up the work of collecting timekeeping data, calculating salaries, taxes, and other deductions, and then generating paychecks.

Stream Processing

Stream processing is the process of being able to almost instantaneously analyze data that is streaming from one device to another.

This method of continuous computation happens as data flows through the system with no compulsory time limitations on the output. With the almost instant flow, systems do not require large amounts of data to be stored.

Stream data processing is highly beneficial if the events you wish to track are happening frequently and close together in time. It is also best to utilize if the event needs to be detected right away and responded to quickly. Stream processing, then, is useful for tasks like fraud detection and cybersecurity. If transaction data is stream-processed, fraudulent transactions can be identified and stopped before they are even complete.

Data streaming process

Here are details to illustrate how the data streaming process works.

- **Event-driven:** When something relevant happens, the event triggers a function. It could be clicking a link on a website or, in the case of IoT devices, a smart lighting system could sense sunset and the darkening of a room.
- **Data flow:** Data is processed as it comes, to support workflows without interruption. Examples include rideshare apps, stock trading platforms, and multiplayer games.
- **Timestamped:** This process is helpful when dealing with data that happens in a time series, such as logs, transaction flows, and task flows.
- **Continuous and heterogeneous:** Helpful when dealing with diverse kinds of data in different formats from multiple sources that are continuous, meaning that they change within a range, such as time. Outside temperature and wind speed are examples.

Data streaming characteristics

- **Low latency:** While not zero, processing time is measurable in seconds, even milliseconds.
- **Scalability:** The process supports unexpected and rapid growth in the amount of data handled.
- **Fault tolerance:** The approach requires high uptimes with minimal, even zero, risk of data loss.
- **State management:** Session tracking, windowing, and counting events in a time window when tracking, storing, and processing data.
- **Distributed processing:** This approach ensures workloads are handled by multiple nodes to provide fault tolerance and scalability.
- **Integration with other systems:** The ability to connect and exchange data with other platforms, including other databases, message queues, and analytic tools.

Challenges with stream processing

One of the biggest challenges that organizations face with stream processing is that the system's long-term data output rate must be just as fast, or faster, than the long-term data input rate otherwise the system will begin to have issues with storage and memory.

Another challenge is trying to figure out the best way to cope with the huge amount of data that is being generated and moved. In order to keep the flow of data through the system operating at the highest optimal level, it is necessary for organizations to create a plan for how to reduce the number of copies, how to target compute kernels, and how to utilize the cache hierarchy in the best way possible.

Stream processing use cases

Stream processing is helpful in several core functions.

- **Fraud detection:** Track and monitor transactions in real-time, flagging activities and events that are suspicious. By quickly identifying potential fraud, you can take steps to either validate a transaction or possibly identify the fraudster or their broader cybercrime exploit.
- **Network monitoring:** It is vital to detect anomalies in network traffic that could indicate either malfunctions or malfeasance. Constant monitoring can quickly detect and address issues.

- **Predictive maintenance:** Monitoring equipment and systems in real-time catches issues early instead of waiting for a failure that could lead to costly downtime. This makes maintenance more efficient and operations more cost-effective.
- **Intrusion detection:** Monitoring for cybersecurity makes it possible to identify unauthorized access in real-time. Responding quickly can limit the harm done and having data helps your organization comply with regulations. You will also better understand the tactics, techniques, and procedures criminals use, to develop better protections moving forward.

Conclusion

While all of these systems have advantages, at the end of the day organizations should consider the potential benefits of each to decide which method is best suited for the use-case.

Additional Resources

- BMC Workload Automation Blog
- BMC Big Data Blog
- Beginner's Guide To Workplace Automation
- What Is a Batch Job?
- What Is a Data Pipeline?

Manage sl as for your batch services joe goldberg from **BMC Software**