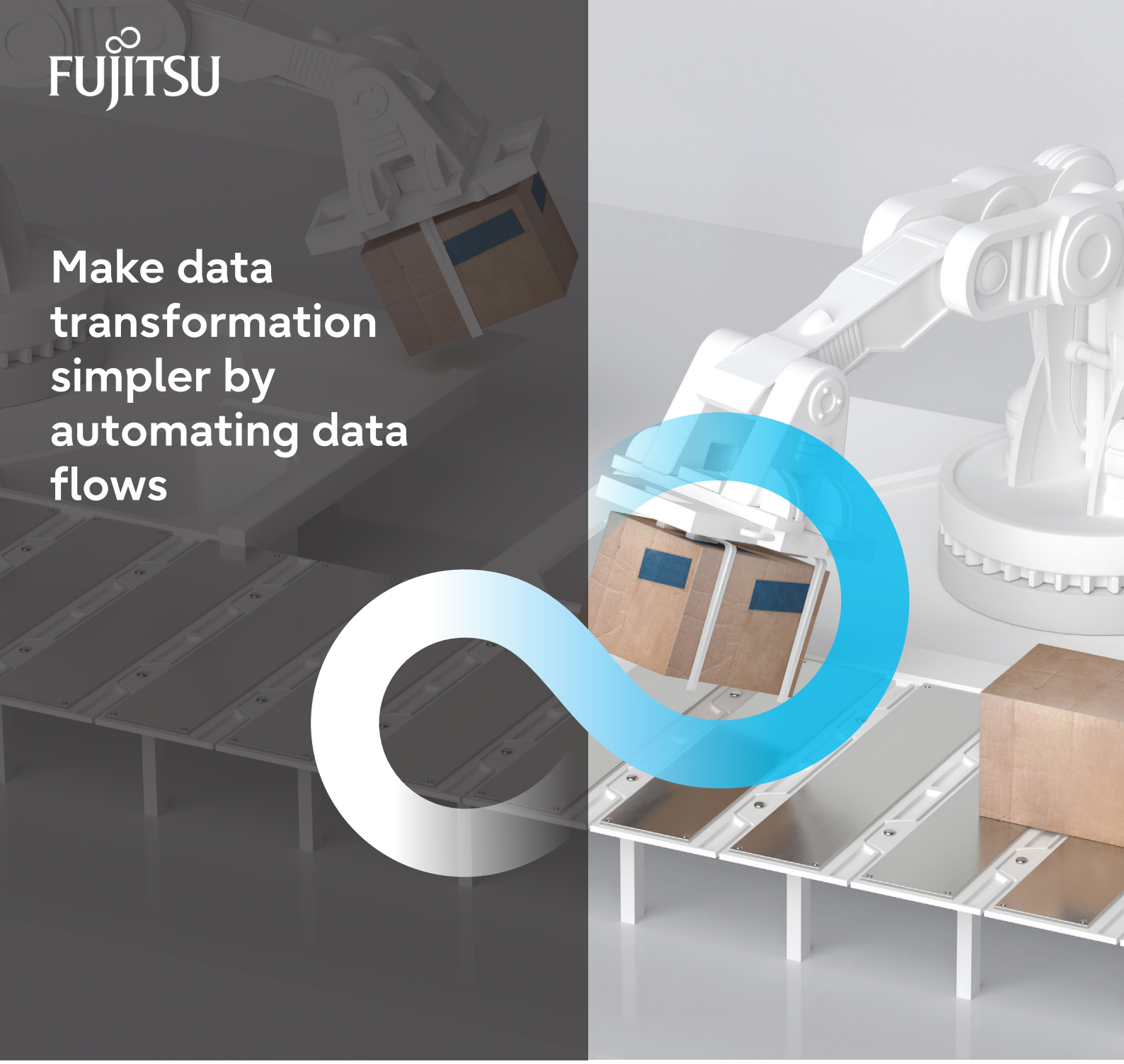
FUJITSU

Make data transformation simpler by automating data flows



If you're an IT Project Manager, you'll know that data transformation can be a long and gruelling affair. Extract, Transform, and Load (ETL) operations that convert data from one format to another are usually difficult to create and require expert knowledge and bespoke code, often running over time and budget. However, there's a much easier solution to many of the ETL headaches. Apache NiFi is a no- to low-code open-source option that allows ETL data flows to be created quickly and simply, radically speeding up the transformation timeline.

What Is Apache NiFi?

NiFi is a free and open-source product from the Apache Software Foundation. Written in Java, it is cross-platform allowing it to be run on any system that has a Java Virtual Machine installed.

Nifi uses a flow-based programming paradigm to simplify and streamline ETL operations. A flow is a collection of data processors, each of which performs a small part of the overall operation. A processor might do something as simple as writing something to a log file, or something more complex like calling a HTTP API endpoint, or writing data to a database. Processors are linked together via queues, which also provide a means of throttling throughput.

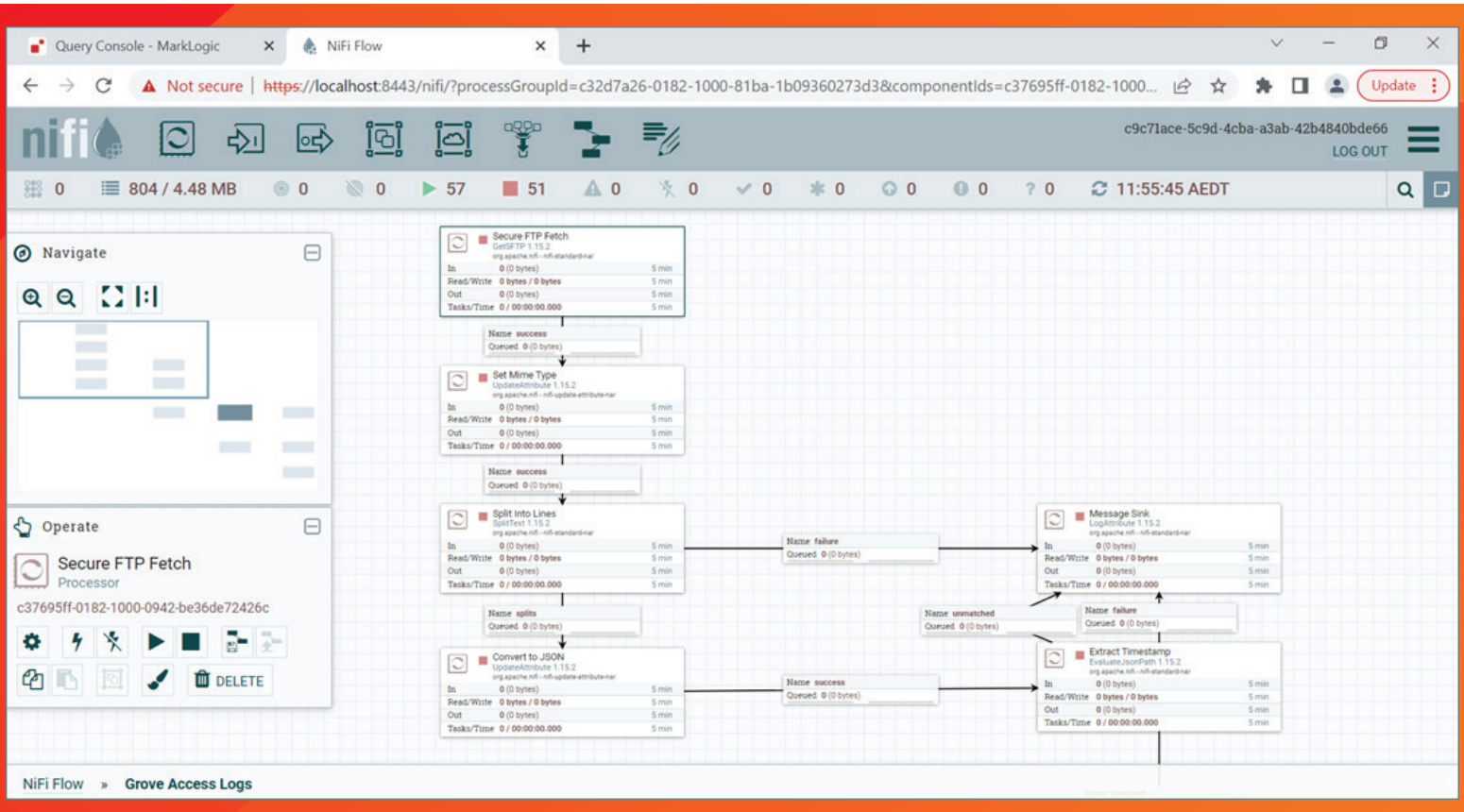
Data is contained in FlowFiles, each of which can have associated metadata attributes, to allow for routing or conditional processing. FlowFiles can contain data in any format, and the contents of a FlowFile can change as it moves through the flow. For example, the flow could read a comma separated value (CSV) file from a filesystem, split that file into individual rows (each in its own flow file), then convert each row into JSON format. All of this can be done without writing any code.

How do you create a flow?

Creating a flow is a simple process. A new processor is added to the flow by dragging the processor icon off the toolbar onto the flow canvas, then selecting which type of processor from a list.

Once added to the canvas, the processor is configured by filling in the required parameters in a configuration dialog. This tailors the operations of that specific processor for the needs of the ETL operation.

Each processor normally has one or more output queue options, for example ‘success’ or ‘failure’. Processors are linked together by dragging one processor and dropping it onto another one, then choosing which queue to connect to the target processor. Linking processors in this way allows for error handling and reprocessing.



Can it be extended?

NiFi comes with a huge range of processor types at installation. However, its out-of-the-box functionality can be extended in several ways:

- The first is third-party provided processors. These are often vendor-specific database interfaces, or JDBC database drivers. Adding new processors like this is as simple as copying the processor Java archive file into a folder in the NiFi installation and restarting NiFi. The new processors will then be immediately available for use in a flow.

- ExecuteScript processors allow for bespoke code to be written in a variety of languages (such as Python, Groovy, or JavaScript). This code is executed against each incoming FlowFile. These scripts are for relatively simple tasks, and error options are limited to success and failure queues only.

- Custom processors can also be developed in Java for situations where more complex logic with finer grained control over outputs is required.

What knowledge level do you require?

Creating a basic NiFi flow is very straight forward and can be learned quickly. A business analyst who understands basic data operation and logic flows could set up a flow. NiFi's expression language is similar in complexity to using a Microsoft Excel formula, so anyone who is familiar with Excel could learn NiFi's expression language syntax quickly.

Writing bespoke code, either for use in an ExecuteScript processor or creation of a custom processor in Java, will require more specialised software development knowledge and experience.