O que é um processamento batch?

É um processo de dados **FINITOS** que ocorre sem **interação ou interrupção**. Ênfase no **FINITO,** você pode ter dados que são processados infinitamente e dados que tem um certo "range", ou seja, tem delimitado onde termina seus processamentos.

Com relação a **interrupção e interação**, é meio óbvio, isso significa que o processo tem que acontecer sem interferências externas. Por exemplo, numa api web você tem algo que da "trigger" no processo, uma rotina batch tem que acontecer sem esse tipo de estimulo, ela é algo que sabe onde começar e onde terminar.

Em que caso são usados processamentos batch?

ETL *(extract, transform and load)*: Basicamente falando é uma rotina onde extraímos dados de algum lugar ***(database, csv etc)***, transformamos eles ***(adicionando um novo campo, fazendo alguma operação aritmética etc)*** e carregamos para um destino ***(salvamos em outro database, transformamos em csv etc...).***

REPORTING: Aqui são processamentos baseados em alguma regra de negócio, por exemplo, em um banco, durante a madrugada é necessário rodar um processamento imenso de dados para que eles estejam disponíveis já no dia seguinte. É basicamente um processamento que acontece depois do expediente de trabalho e tem uma demanda específica.

Big Data: se processos batch são feitos para trabalhar com uma quantidade grande de dados, obviamente ele está envolvido em BIG DATA, muito na parte de exportar esses dados para um software mais específico.

O que é o Spring Batch?

Um framework específico para processamentos batch na JVM (resposta esperada né). Ele foi desenvolvido com base na **JSR 352** (esse cara aqui é uma especificação do JAVA EE 7 para se trabalhar com processamento em lote/batch), então ele é bem semelhante a ela. Dentre os recursos que o Spring Batch prove, temos:

Job flow State machine: Quando você define um **job***(que é um fluxo composto de "steps"*), todo o controle de infraestrutura em volta da transição entre os **steps** *(tipo, tu vai do step 1 pro 2, do 2 pro 3, e assim em diante)* é feito pelo **Spring Batch**, que te além de te oferecer handlings (maneiras de se lidar com) desse estado, também te da formas fáceis de configura-lo.

Transaction Handling: Tudo que você possa imaginar sobre controle de transações, o Spring Batch lida pra você. Por exemplo, se você tiver um arquivo com 1 milhão de registros provavelmente você não vai querer importar todos eles de uma vez numa única transação.

O Spring batch oferece facilidades para quebrar isso em **chunks** (pedaços) menores e manter o estado, então conforme você for importando e commitando essas mudanças, se em algum ponto falhar, o Spring Batch consegue recomeçar o processo a partir desse ponto.

Declarative I/O: Aqui estamos falando do Spring Batch já nos oferecer diversas maneiras de se ler entradas e lidar com erros durante essa leitura, sem que precisemos fazer essa implementação que é de infra mas sim fazer a implementação de regras de negócio.

Fora isso, tem uma outra série de benefícios, a gente está falando de algo do **ecossistema Spring**, então dá pra utilizar com outros projetos do Spring. Ele já é muito bem testado e está no mercado a muito tempo.