1. **Transformacja payload-u do tablicy**

Na potrzebę tego zadania przyjmijmy, że string wysyłany przez http ma zawsze postać liter, znaków oddzielonych “,” np.

Aa,bb,cc

dasd,fdsfds,fvf,vgf,dsds

Dodaj komponent Sat Payload za kostką ObjectToString i wykonajmy na nim operację split()

#[payload.split(",")]

1. **Dodaj komponent Collection Splitter**

Komponent ten przyjmuje kolekcję na wejście i dokonuje podzielenia komunikatu na niezależne elementy - atomowe komunikaty. Mechanizm działa w oparciu o 3 nagłówki:

MULE\_CORRELATION\_GROUP\_SIZE=4

MULE\_CORRELATION\_ID=5bc2c710-8699-11e8-9a07-080027c3c6a2

MULE\_CORRELATION\_SEQUENCE=4

Każdy niezależny komunikat ma swój SEQUENCE number, ten sam correlationID oraz informację o ilości wszystkich komunikatów w ramach grupy.

Jest to niezbędne do tego by móc dokonać agregacji komunikatów po ich przetworzeniu.

1. **Zmiana komunikatu wysyłanego**

Wykorzystajmy komponent Set Paylod z wyrażeniem #[payload]#[payload]. W tym miejscu operujemy na poziomie pojedynczego elementu z tablicy, nasz payload z tablicy stringów stał się pojedynczym elementem z tablicy

1. **Dodaj AsynchScope**

Krok CollectionSplitter, Set Paylod oraz JMS przenosimy do AsynchScope który wstawiamy za krokiem JMS

1. **Dodaj komponent Collection Aggregator**

Do przepływu konsumującego dodajemy komponent Collection Aggregator oraz ObjectToString. Komponent Agregacyjny wstawiamy za Filtrem, ponieważ chcemy agregować tylko prawidłowo przefiltrowane komunikaty.