

**JET  
BRAINS**



# Real World Java 9



**Trisha Gee (@trisha\_gee)**

**Developer & Technical Advocate, JetBrains**

# Real World Java 9

- Jigsaw in Action
- Reactive Programming in Java 9
- The Other Good Stuff

# Why Java 9?



# Free Lunch!

- Better Use of Memory
- Better Performance of:
  - Locking
  - Secure Apps
  - Graphics
- Better Use of Hardware
- Better Documentation
- Prettier Graphics (Windows and Linux)
- Faster Compilation

**Jigsaw**

**REPL**

**And... other things?**

[274: Enhanced Method Handles](#)

[275: Modular Java Application Packaging](#)

[276: Dynamic Linking of Language-Defined Object Models](#)

[277: Enhanced Deprecation](#)

[278: Additional Tests for Humongous Objects in G1](#)

[279: Improve Test-Failure Troubleshooting](#)

[280: Indify String Concatenation](#)

[281: HotSpot C++ Unit-Test Framework](#)

[282: jlink: The Java Linker](#)

[283: Enable GTK 3 on Linux](#)

[284: New HotSpot Build System](#)

[285: Spin-Wait Hints](#)

[287: SHA-3 Hash Algorithms](#)

[288: Disable SHA-1 Certificates](#)

[289: Deprecate the Applet API](#)

[290: Filter Incoming Serialization Data](#)

[292: Implement Selected ECMAScript 6 Features in Nashorn](#)

[294: Linux/s390x Port](#)

[295: Ahead-of-Time Compilation](#)

[297: Unified arm32/arm64 Port](#)

[298: Remove Demos and Samples](#)



# Caveats & Disclaimers



# Java 9 in Production?

**Questionable**

**Everything is EA**

**Best Practice?**

# The Case Study





TWITTER  
FIREHOSE

CONNECT  
&  
EXTRACT

MOOD  
SERVICE

OVERALL  
MOOD  
MVC

HAPPINESS  
MVC

USER  
SERVICE

LEADERBOARD  
MVC

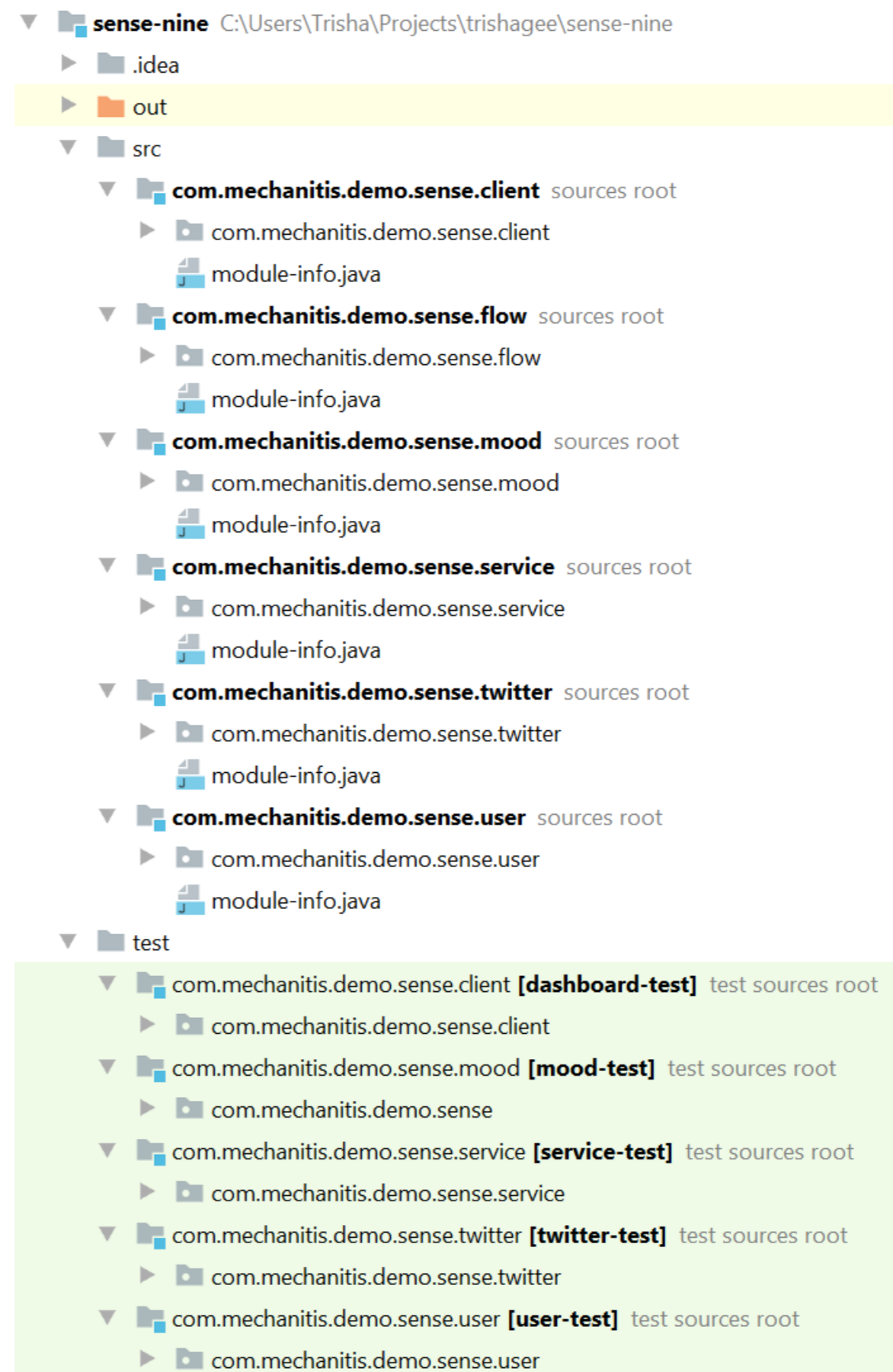
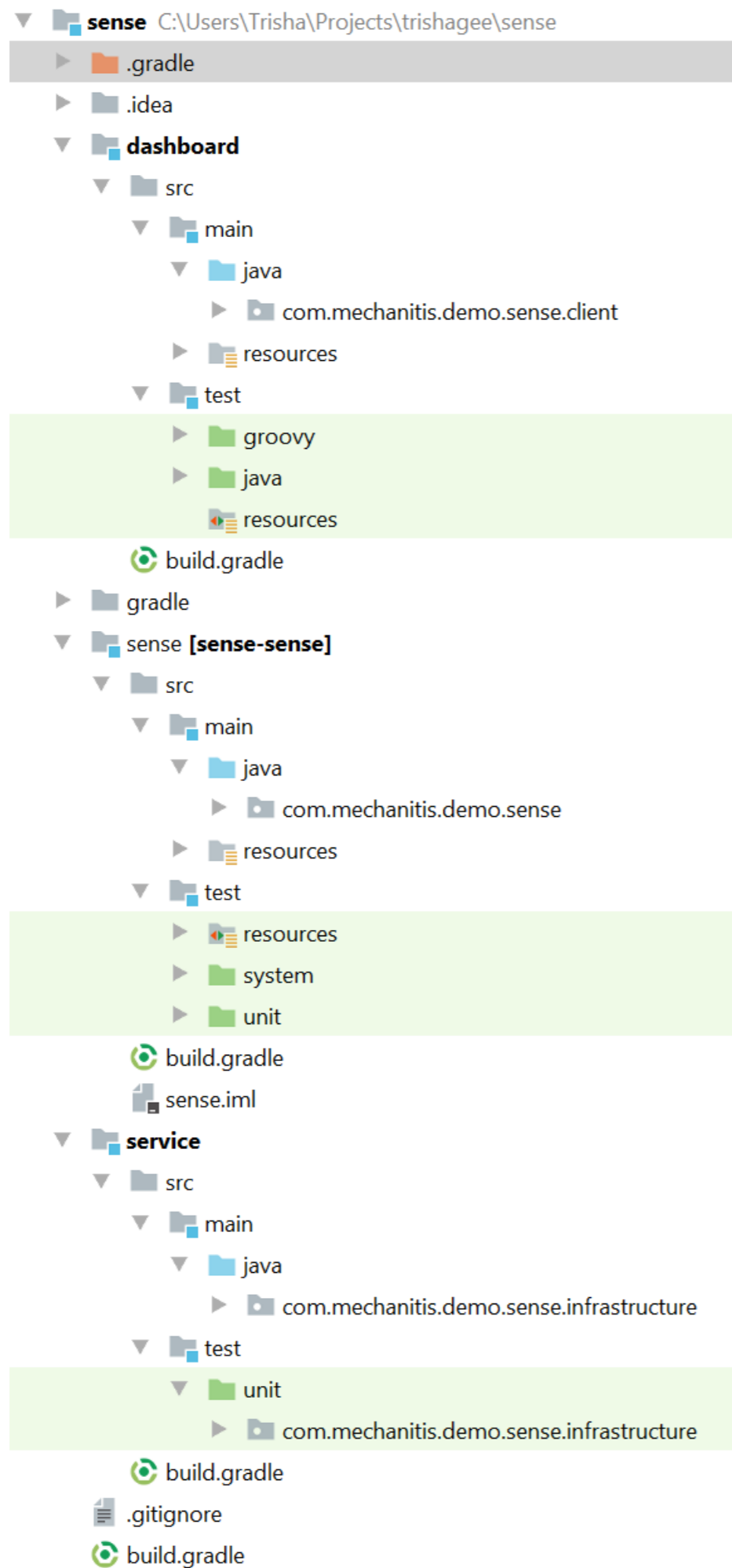


**<http://bit.ly/Java8InAnger>**



**Jigsaw**





▼ **jdk9** sources root, C:\Users\Trisha\Projects\sa

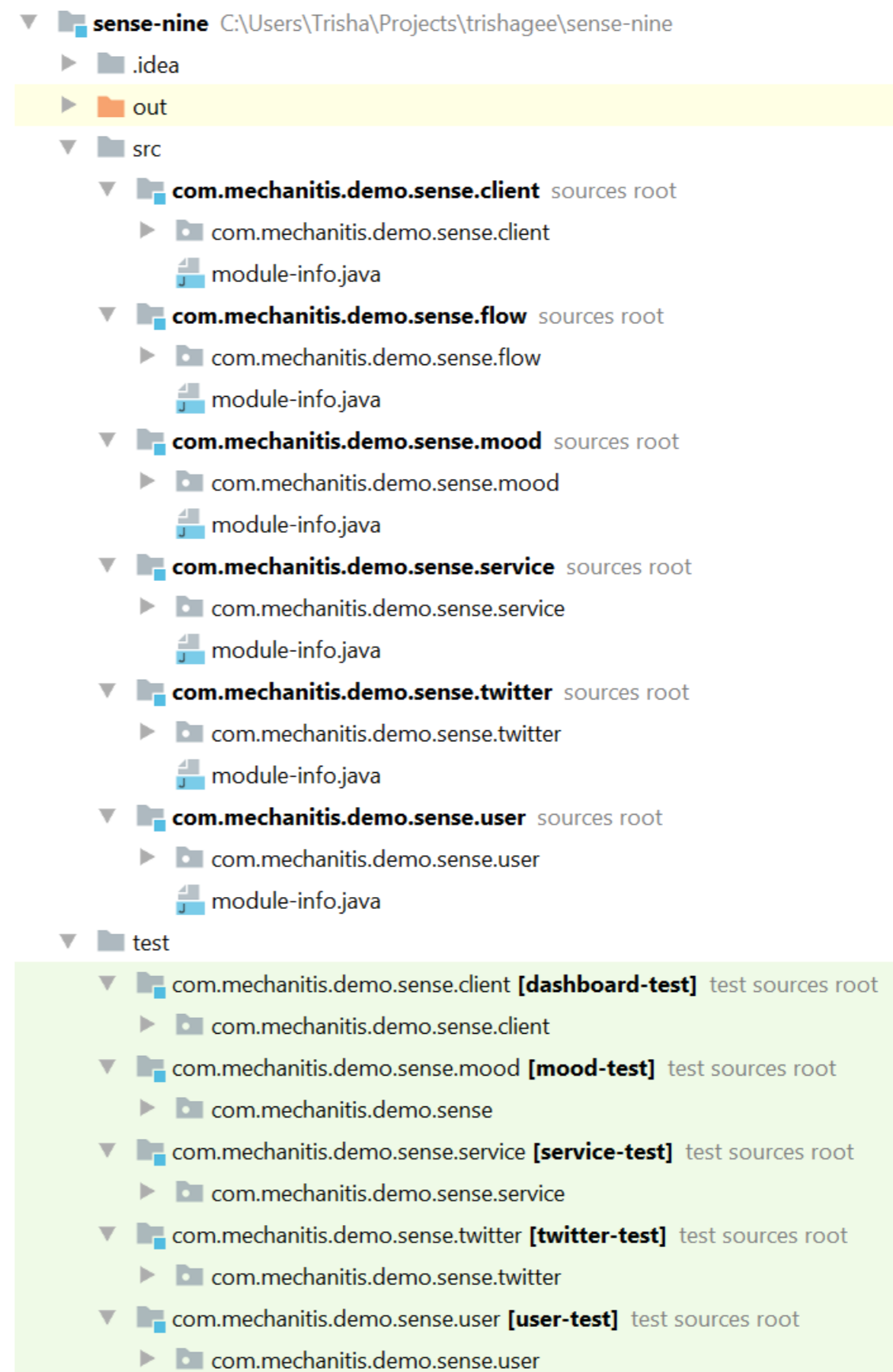
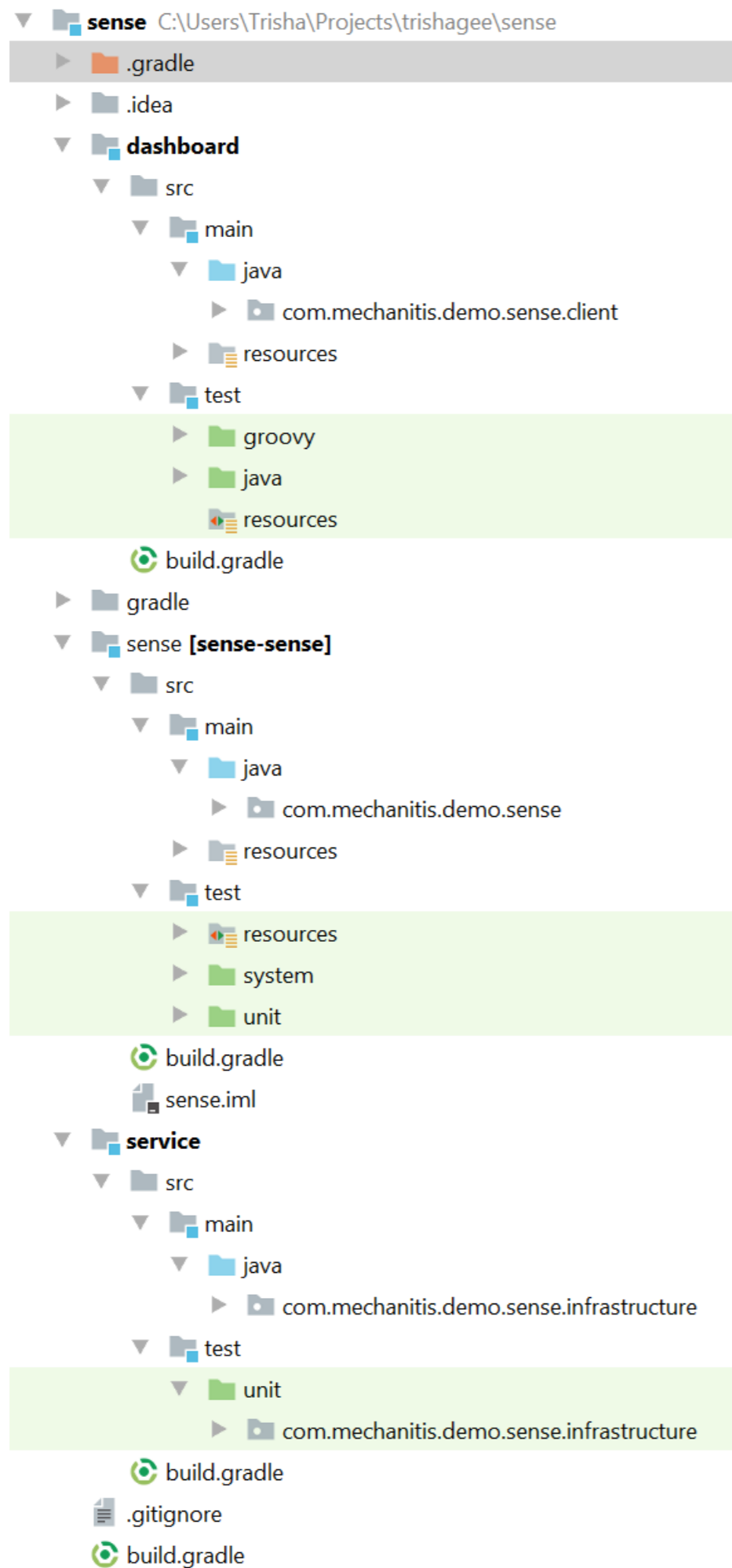
- ▶ .idea
- ▶ jcheck
- ▶ make
- ▼ src
  - ▶ bsd.doc.man
  - ▶ demo
  - ▶ **java.base**
  - ▶ **java.datatransfer**
  - ▶ **java.desktop**
  - ▶ **java.instrument**
  - ▶ **java.logging**
  - ▶ **java.management**
  - ▶ **java.management.rmi**
  - ▶ **java.naming**
  - ▶ **java.prefs**
  - ▶ **java.rmi**
  - ▶ **java.scripting**
  - ▶ **java.se**
  - ▶ **java.se.ee**
  - ▶ linux.doc.man
  - ▶ sample
  - ▶ solaris.doc.sun.man.man1

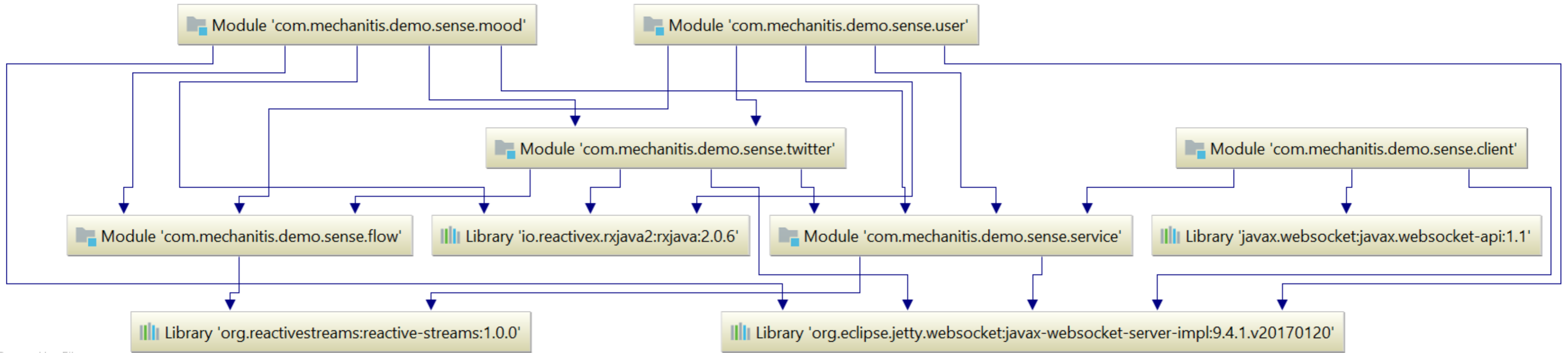
- ▼ test
  - ▶ com
  - ▶ **demo**
  - ▼ **java [test1]**
    - ▶ **awt**
    - ▶ **beans**
    - ▶ io
    - ▶ lang
    - ▶ **math**
    - ▶ net
    - ▶ **nio [nio3]**
    - ▶ **rmi**
    - ▶ **security [security3]**
    - ▶ sql
    - ▶ **text**

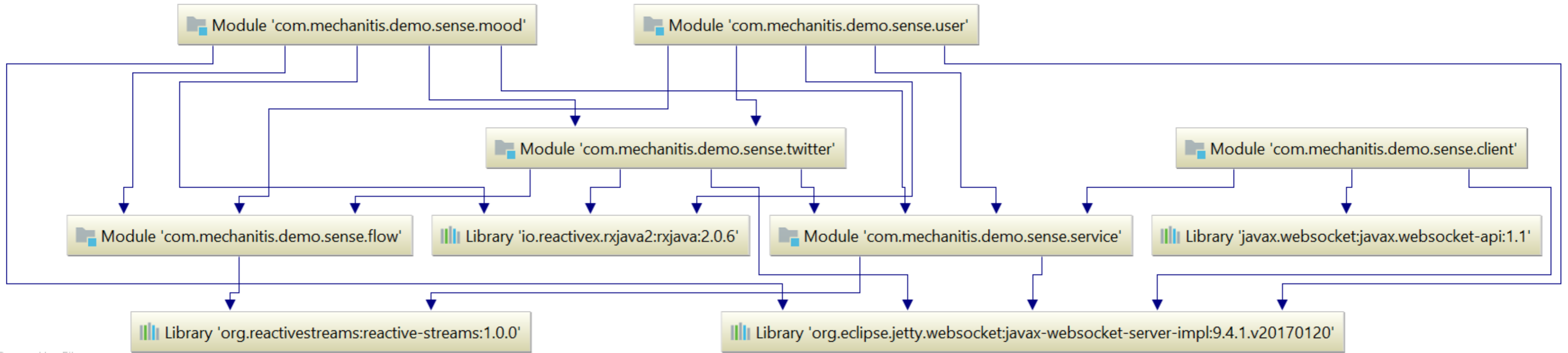
▼ **sense-nine** C:\Users\Trisha\Projects\trishagee\sense-nine

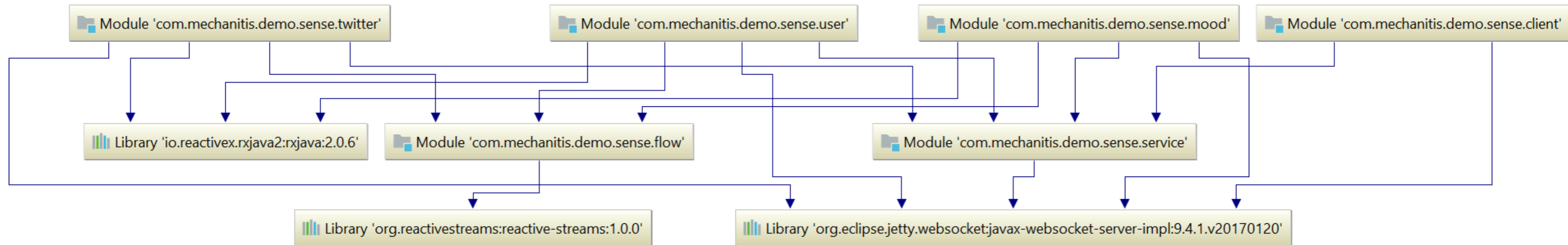
- ▶ .idea
- ▶ **out**
- ▼ src
  - ▼ **com.mechanitis.demo.sense.client** sources root
    - ▶ com.mechanitis.demo.sense.client
      - module-info.java
  - ▼ **com.mechanitis.demo.sense.flow** sources root
    - ▶ com.mechanitis.demo.sense.flow
      - module-info.java
  - ▼ **com.mechanitis.demo.sense.mood** sources root
    - ▶ com.mechanitis.demo.sense.mood
      - module-info.java
  - ▼ **com.mechanitis.demo.sense.service** sources root
    - ▶ com.mechanitis.demo.sense.service
      - module-info.java
  - ▼ **com.mechanitis.demo.sense.twitter** sources root
    - ▶ com.mechanitis.demo.sense.twitter
      - module-info.java
  - ▼ **com.mechanitis.demo.sense.user** sources root
    - ▶ com.mechanitis.demo.sense.user
      - module-info.java

- ▼ test
  - ▼ **com.mechanitis.demo.sense.client [dashboard-test]** test sources root
    - ▶ com.mechanitis.demo.sense.client
  - ▼ **com.mechanitis.demo.sense.mood [mood-test]** test sources root
    - ▶ com.mechanitis.demo.sense
  - ▼ **com.mechanitis.demo.sense.service [service-test]** test sources root
    - ▶ com.mechanitis.demo.sense.service
  - ▼ **com.mechanitis.demo.sense.twitter [twitter-test]** test sources root
    - ▶ com.mechanitis.demo.sense.twitter
  - ▼ **com.mechanitis.demo.sense.user [user-test]** test sources root
    - ▶ com.mechanitis.demo.sense.user









**module-info.java**



# Pain

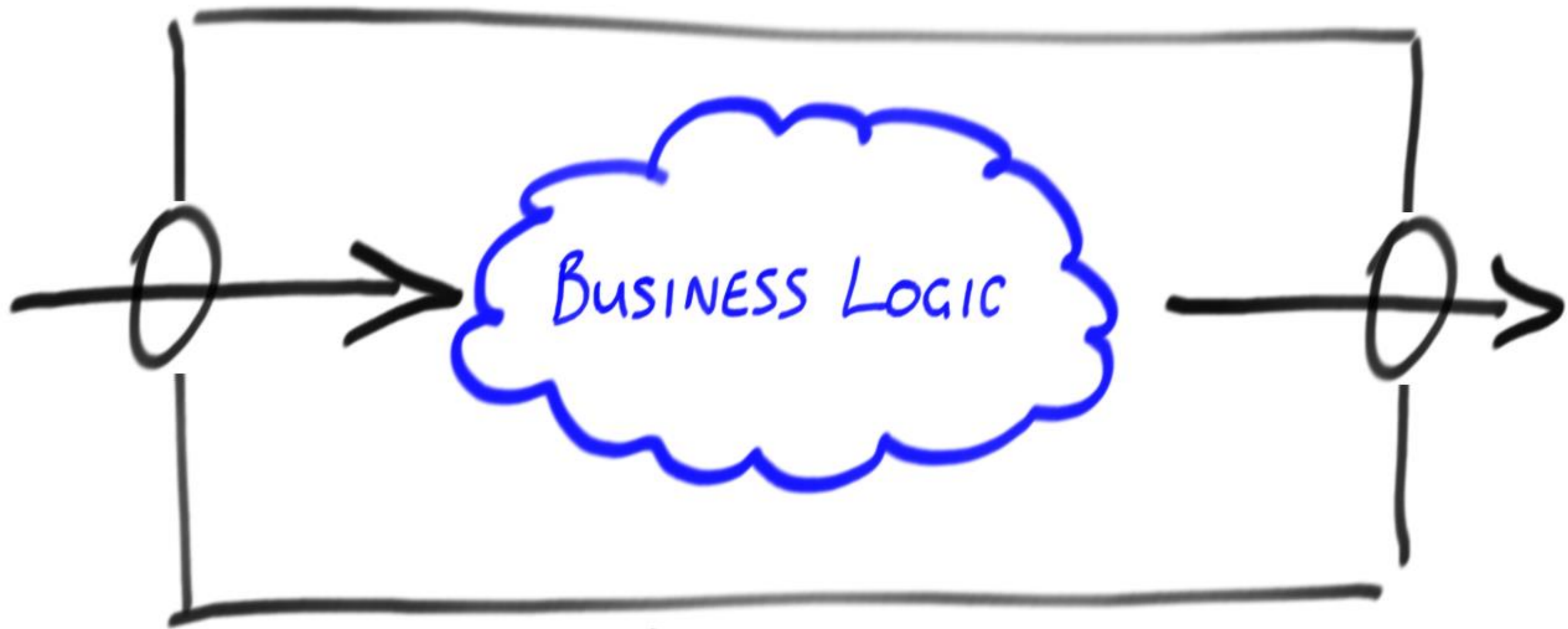
- Gradle not working
- Reorganising the file structure is non-trivial
- This project has no command-line build
- IDE support is not what we're used to
- RTFM. No, really do.

# Reactive API

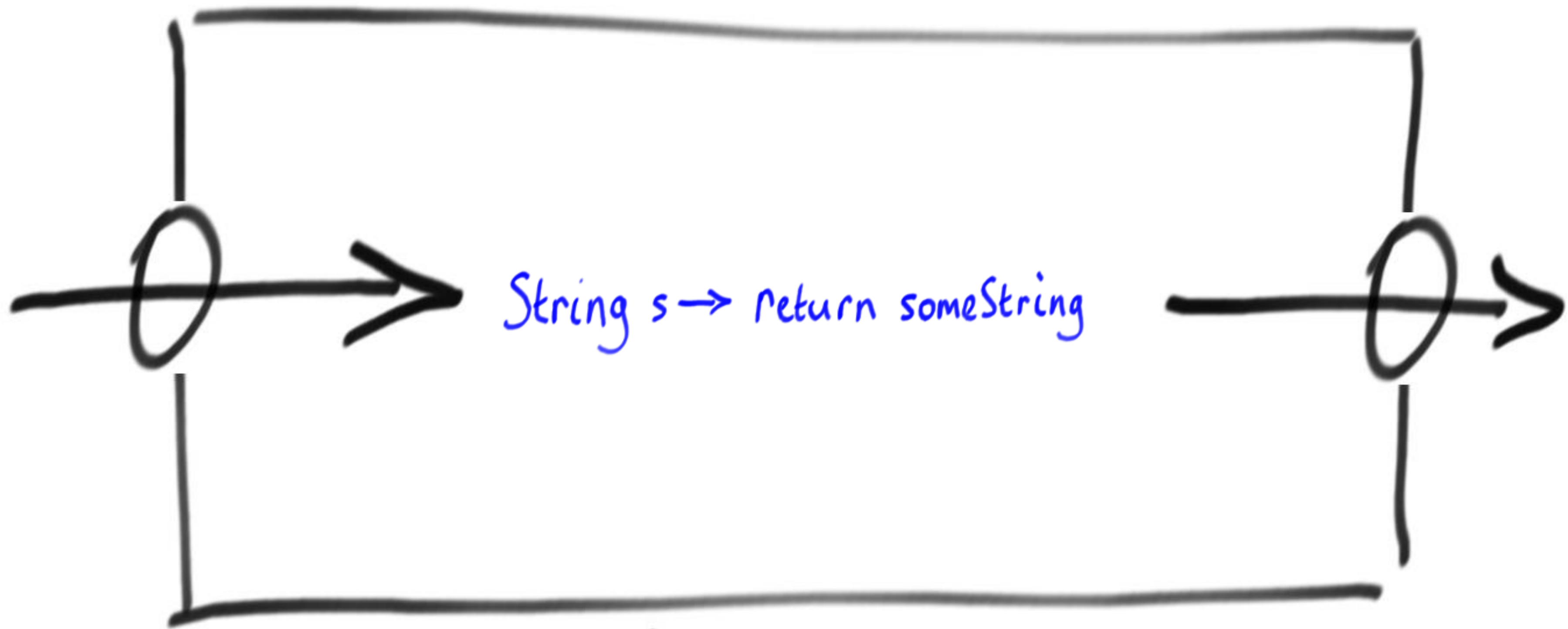




SERVICE



SERVICE

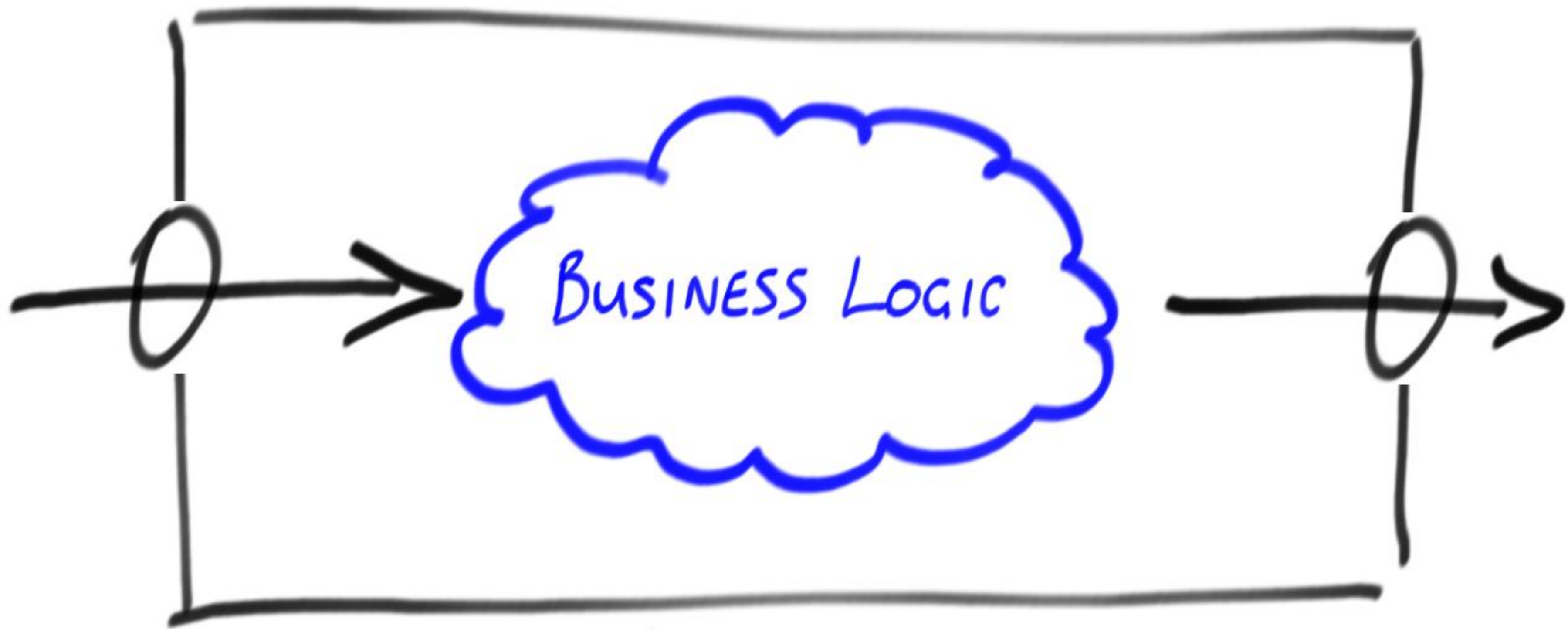


String s → return someString

SERVICE

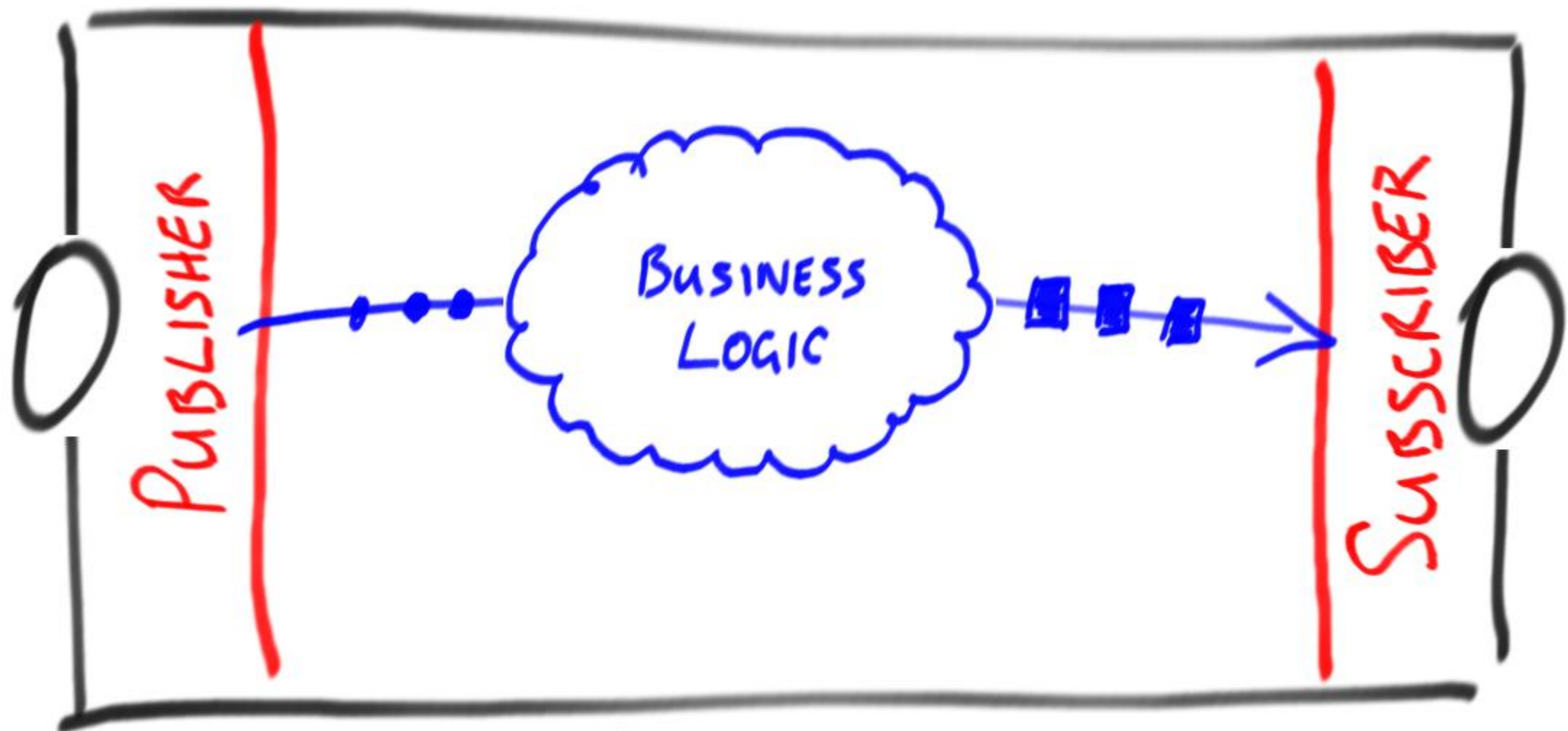
```
public Service(String endpointToConnectTo,  
               String serviceEndpointPath,  
               int servicePort,  
               Function<String, String> messageHandler) {  
    this.endpointToConnectTo = endpointToConnectTo;  
    this.messageHandler = messageHandler;  
    this.serviceEndpointPath = serviceEndpointPath;  
    this.servicePort = servicePort;  
}
```

```
public UserService() {  
    service = new Service(endpointToConnectTo: "ws://localhost:8081/tweets/",  
        serviceEndpointPath: "/users/", PORT,  
        fullTweet -> TweetParser.getTwitterHandleFromTweet(fullTweet));  
}
```



SERVICE

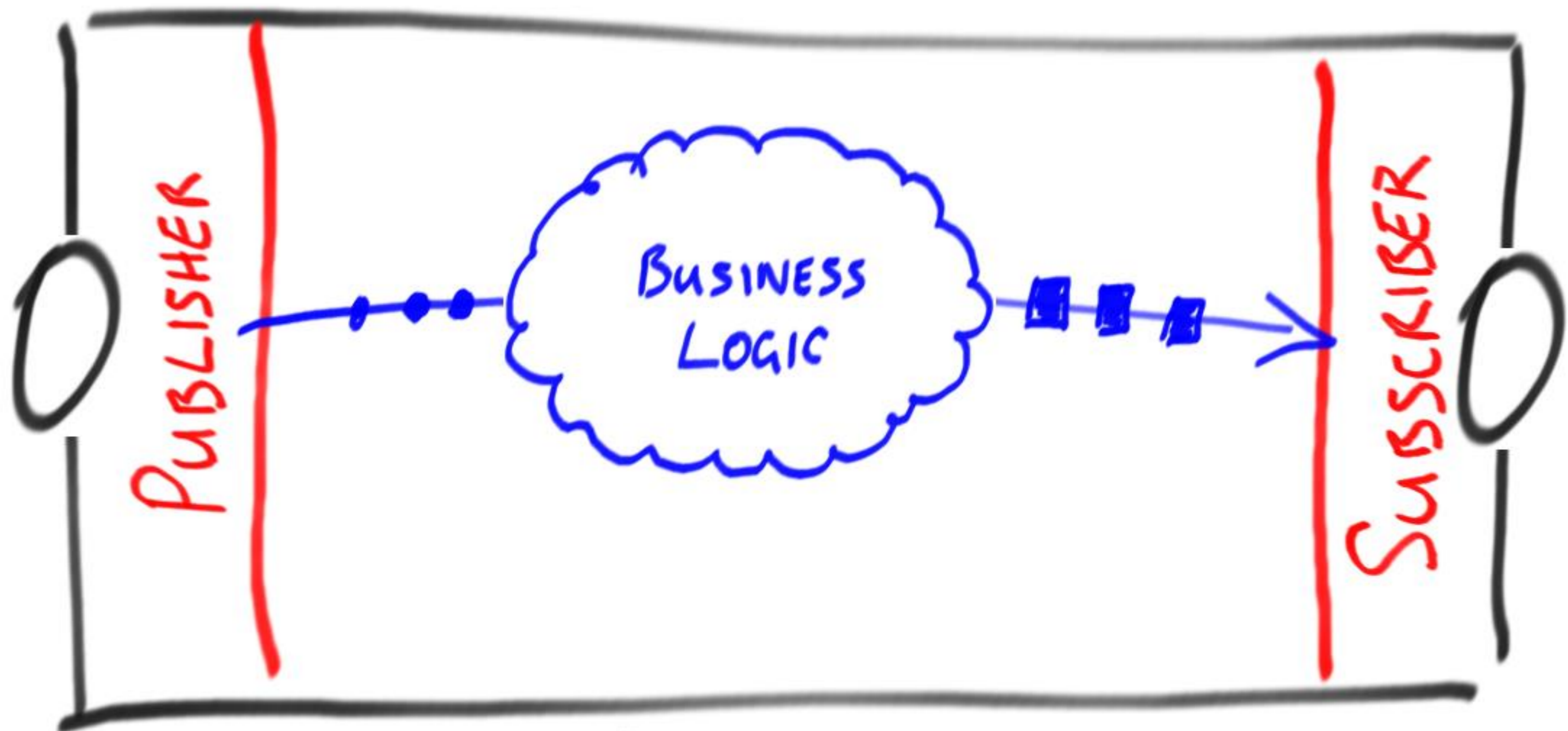




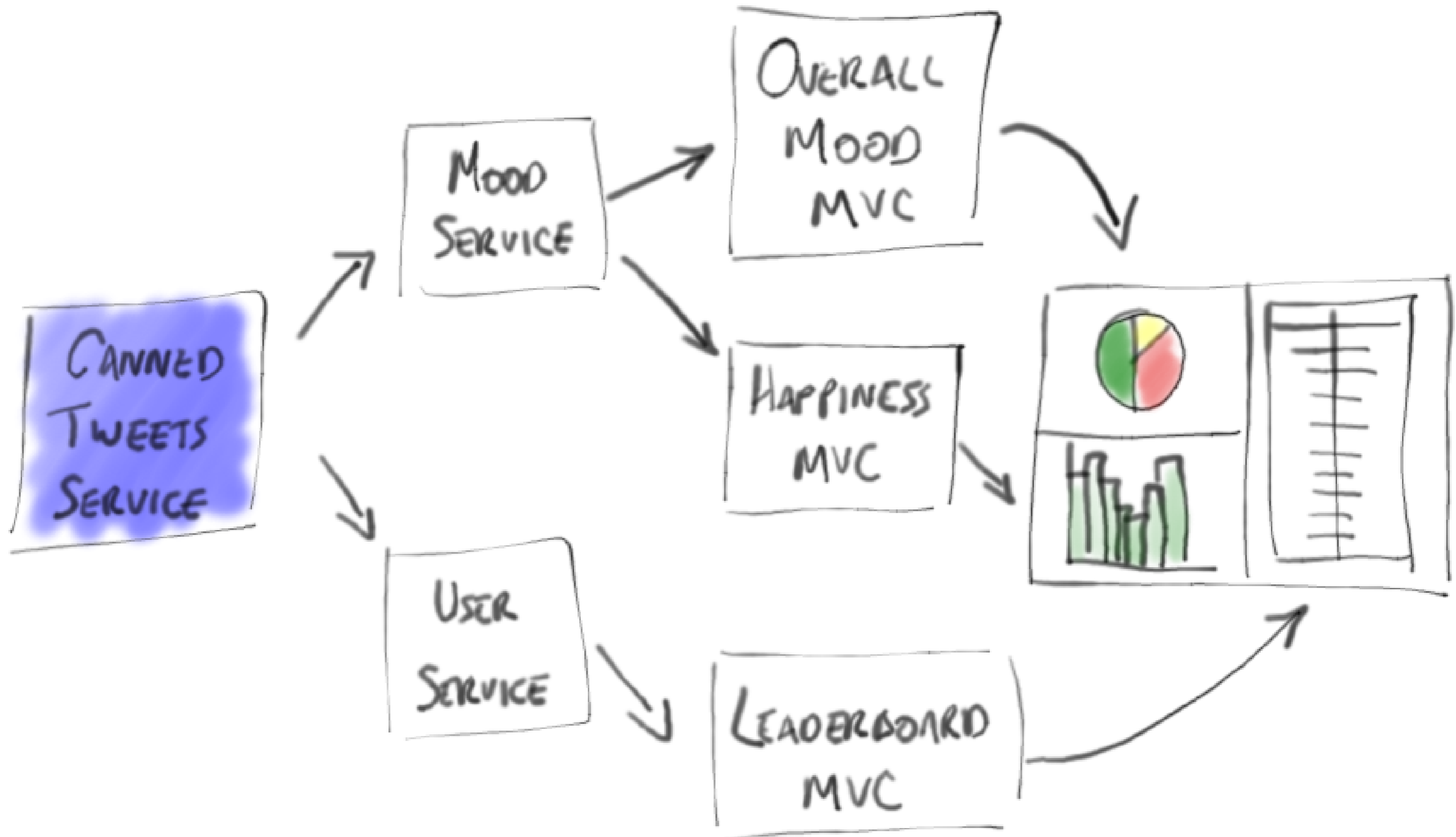
SERVICE

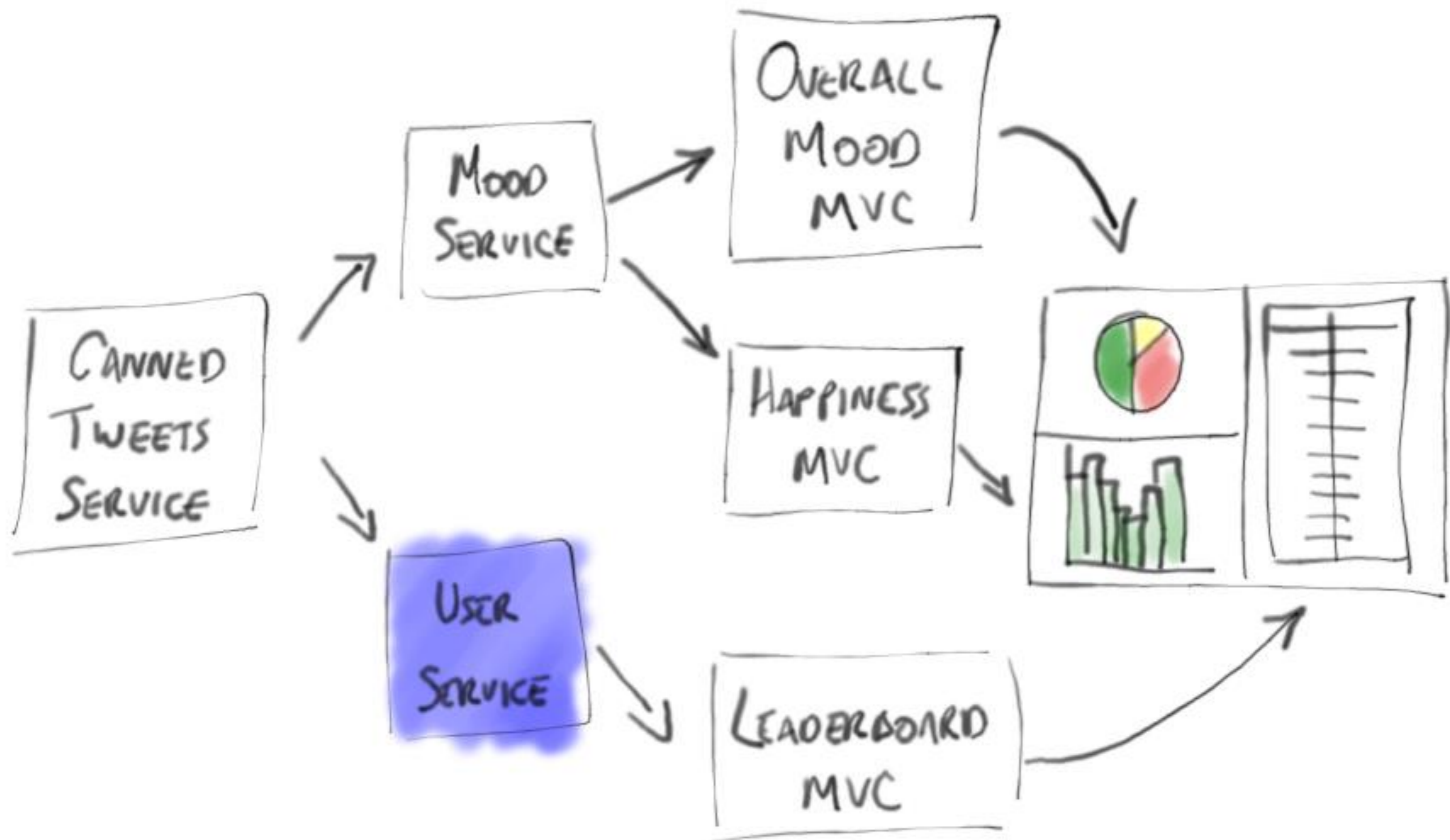
# Reactive In Practice

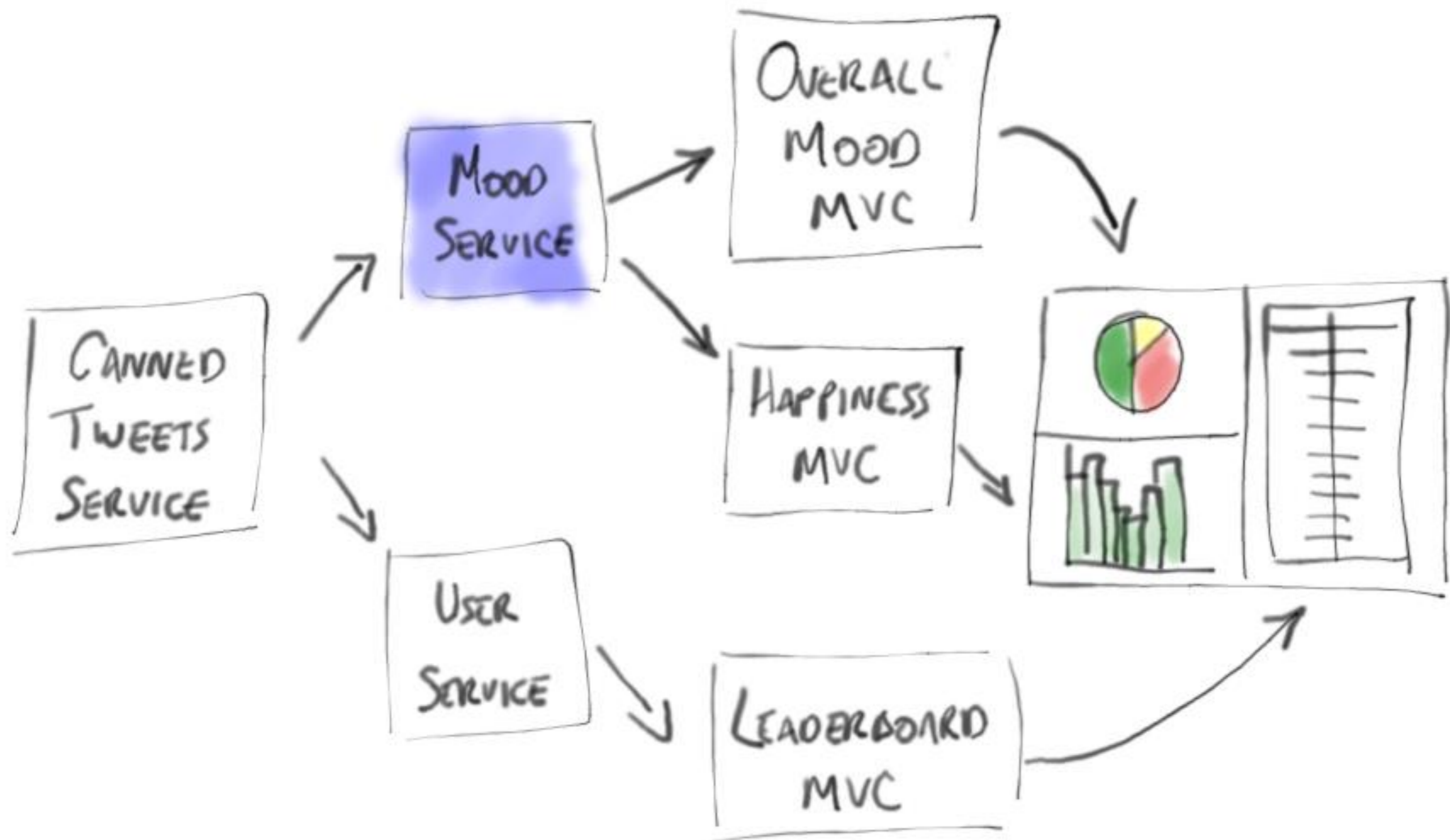




SERVICE







```
"{"created_at":"Tue Jan 27 12:37:11 +0000 2015"," +  
"id":560053908144275456,"id_str":"560053908144275456"," +  
"text":"%s","source":"twitter"}";
```



"HAPPY"

```
"{"created_at":"Tue Jan 27 12:37:11 +0000 2015"," +  
"id":560053908144275456,"id_str":"560053908144275456"," +  
"text":"%s","source":"twitter"}";
```



"SAD"



```
"{"created_at":"Tue Jan 27 12:37:11 +0000 2015"," +  
"id":560053908144275456,"id_str":"560053908144275456"," +  
"text":"%s","source":"twitter"}";
```



"SAD, HAPPY"

```
"{"created_at":"Tue Jan 27 12:37:11 +0000 2015"," +  
"id":560053908144275456,"id_str":"560053908144275456"," +  
"text":"%s","source":"twitter"}";
```



```
" "
```

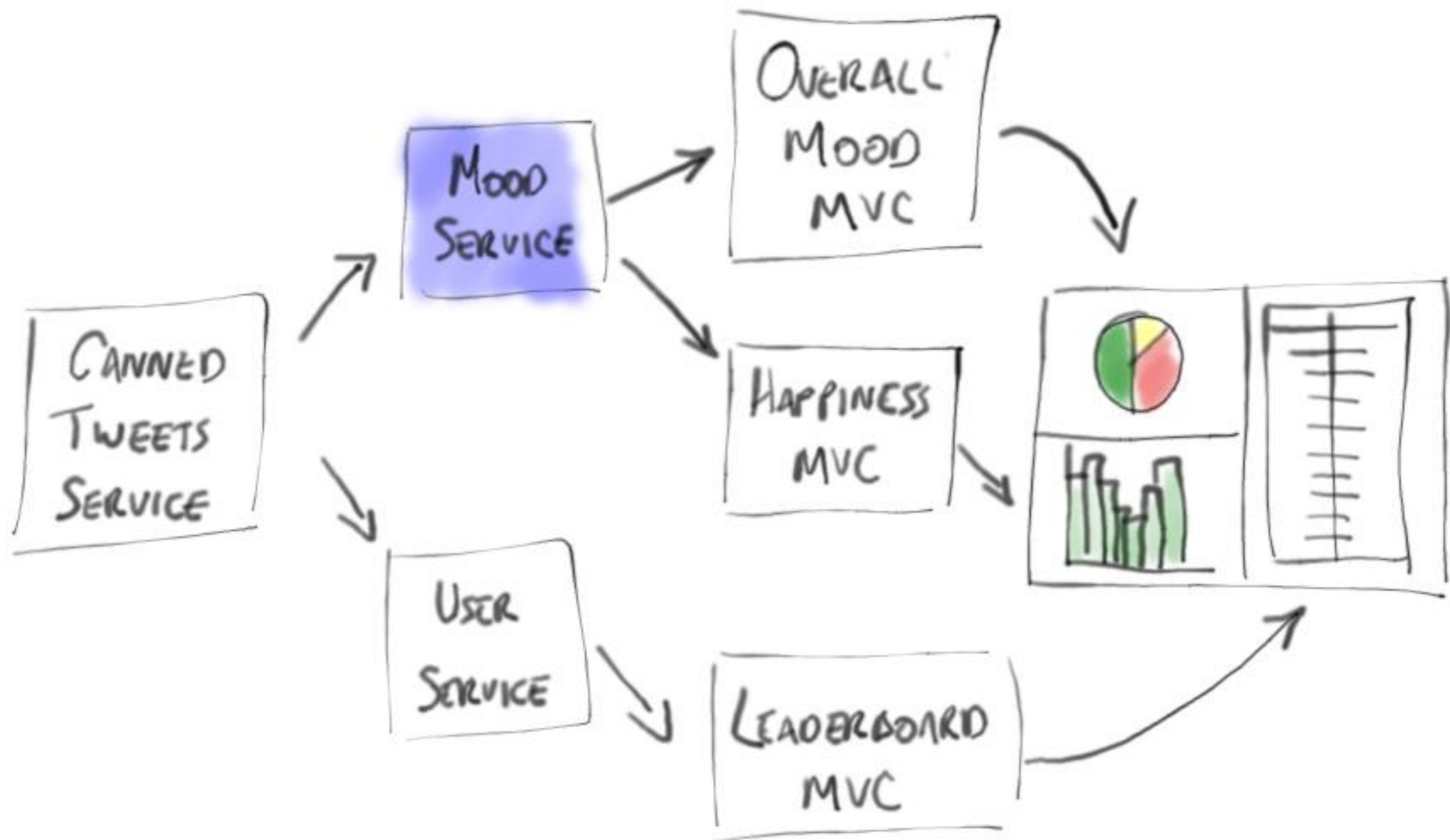
```
"{"created_at":"Tue Jan 27 12:37:11 +0000 2015"," +  
"id":560053908144275456,"id_str":"560053908144275456"," +  
"text":"%s","source":"twitter"}";
```

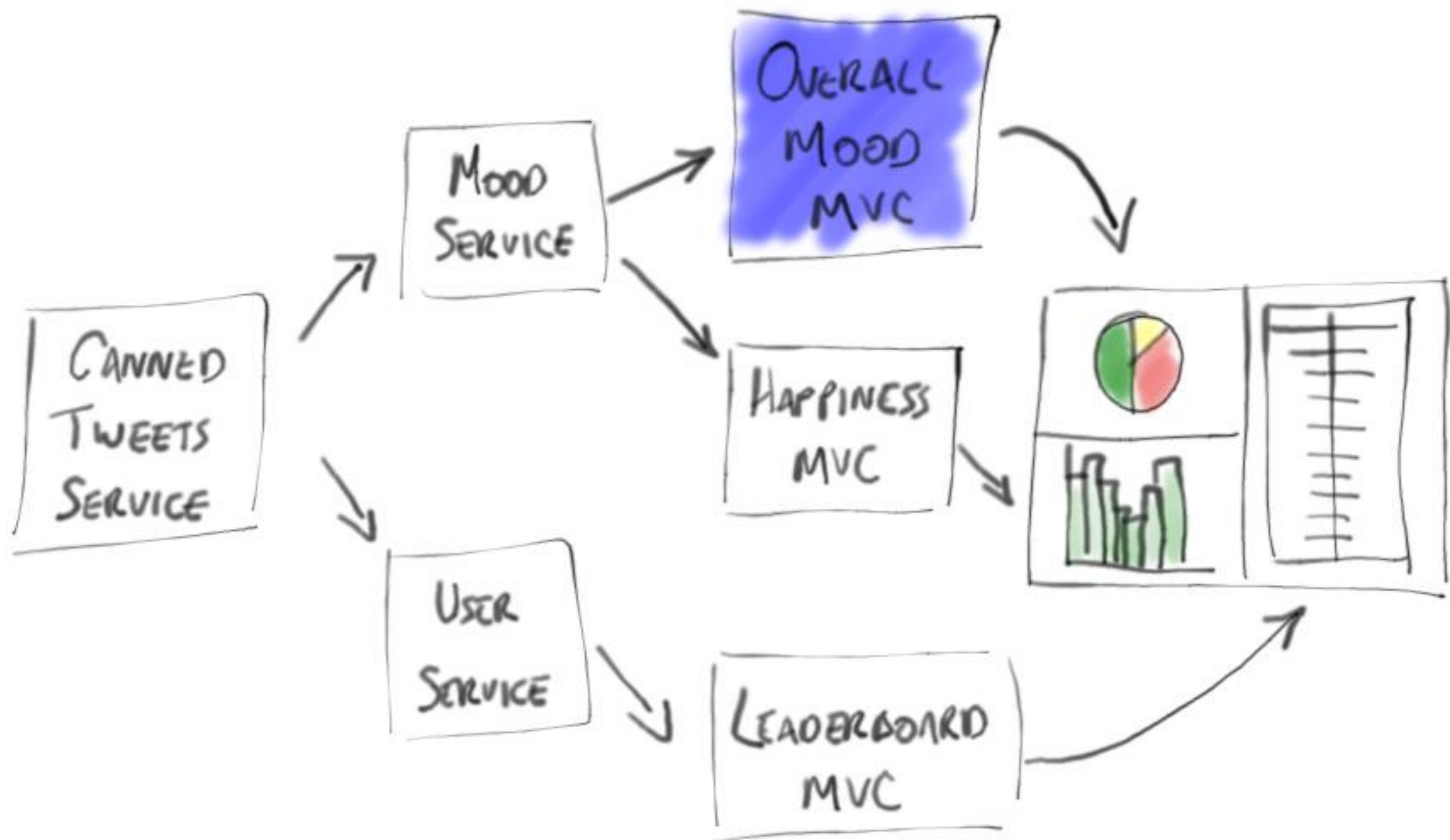


"HAPPY" "HAPPY" "SAD"

```
"{"created_at":"Tue Jan 27 12:37:11 +0000 2015"," +  
"id":560053908144275456,"id_str":"560053908144275456"," +  
"text":"%s","source":"twitter"}";
```







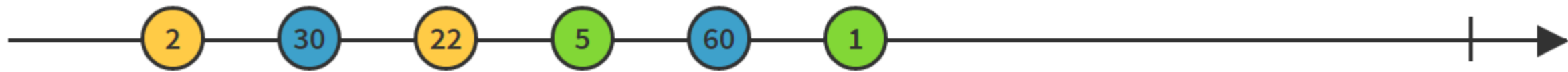
# Java 8 vs Reactive



```
try (Stream<String> lines = lines(filePath)) {  
    lines.filter(s -> !s.equals("OK"))  
        .peek(s -> this.addArtificialDelay())  
        .forEach(tweetsEndpoint::onNext);  
}
```

```
try {  
    Flowable.fromIterable(Files.readAllLines(filePath)  
        .filter(s -> !s.equals("OK"))  
        .doOnNext(s -> this.addArtificialDelay()))  
        .forEach(tweetsEndpoint::onNext);  
}
```





---

`filter(x => x > 10)`

---



# Reactive

- Work with multiple streams
- Have multiple consumers
- Backpressure
- Very different concurrency model

# Other Java 9 Features



# **269: Convenience Factory Methods for Collections**

# **213 : Private Methods on Interfaces**

# **New Methods on the Streams API**

# The Pain



**Your code might break!**

Try it out now to find out sooner, not later



# Going Away

- 214: Remove Deprecated GC Combinations
- 231: Remove Launch-Time JRE Version Selection
- 240: Remove the JVM TI hprof Agent
- 241: Remove the jhat Tool
- 260: Encapsulate Most Internal APIs
- 289: Deprecate the Applet API
- 298: Remove Demos and Samples

# Possible Unexpected Behaviour

- 158: Unified JVM Logging
- 223: New Version-String Scheme
- 245: Validate JVM Command-Line Flag Arguments
- 248: Make G1 the Default Garbage Collector
- 271: Unified GC Logging

**Java 9 is still evolving**

So are the tools we use

# **Don't rush in to modularisation**

It might not be for you

# **Reactive Streams are not the same as Java 8 Streams**

Different behaviour, different use cases

# The Gain



**Encourages good design**

**Encourages new ways of working**



**Just a bit nicer to write code**



<http://bit.ly/RealJava9>