Async Webapps

SOGETI

Vert.x, AngularJS, MongoDB

Erwin de Gier
Sogeti Java CoE
Amsterdam, Februari 2015

Sogeti Nederland

Kernwaarden:

Passie.
Plezier.
Resultaat.
Vertrouwen.
Vakmanschap.

ICT dienstverlener met ruim 35 jaar ervaring en 2.500 medewerkers

Vestigingen in:

> Vianen *(hoofdkantoor)*
> Groningen
> Amsterdam Zuidoost
> Amersfoort
> Eindhoven
> Capelle a/d IJssel

SOGETI

Onze passies

iOS 7

CEH™
ertified Ethical Hacker

android

Java
Sun Microsystems

Windows Azure™

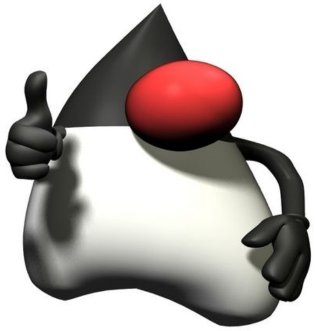
SharePoint

ICT.
Passie.
Resultaat.

SkySight

OWASP

SOGETI



SOGETI
TMAP NEXT

Studentenunit

- Voor derde en vierdejaars ICT studenten
- Cursussen, technische meetings, symposia en fun activiteiten
- Eerste keuze afstudeermogelijkheden
- Jij maakt kennis met ons
- Wij leren jou kennen!

http://www.werkenbijsogeti.nl/studenten

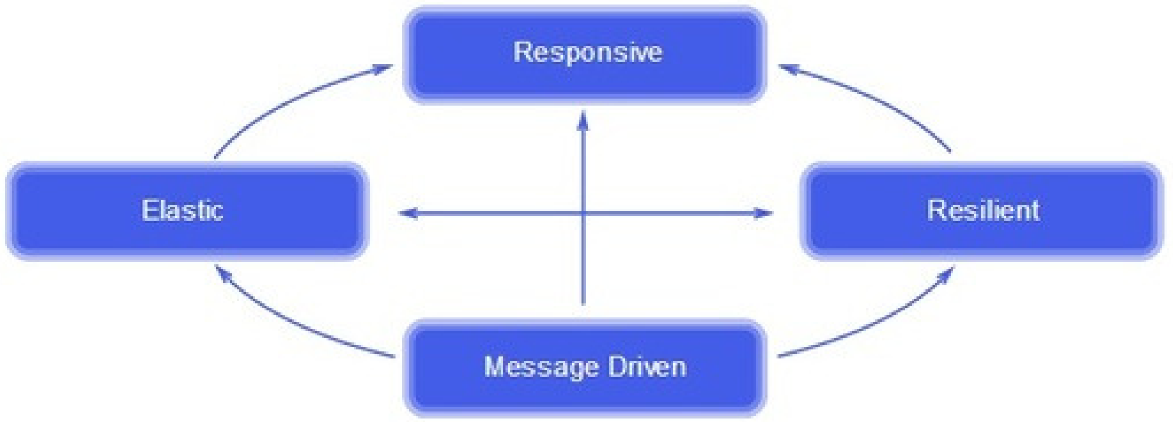
SOGETI

Demands

- Mobile
- Multicore
- Cloud computing
- Interactive & real-time
- Responsive
- Collaborative

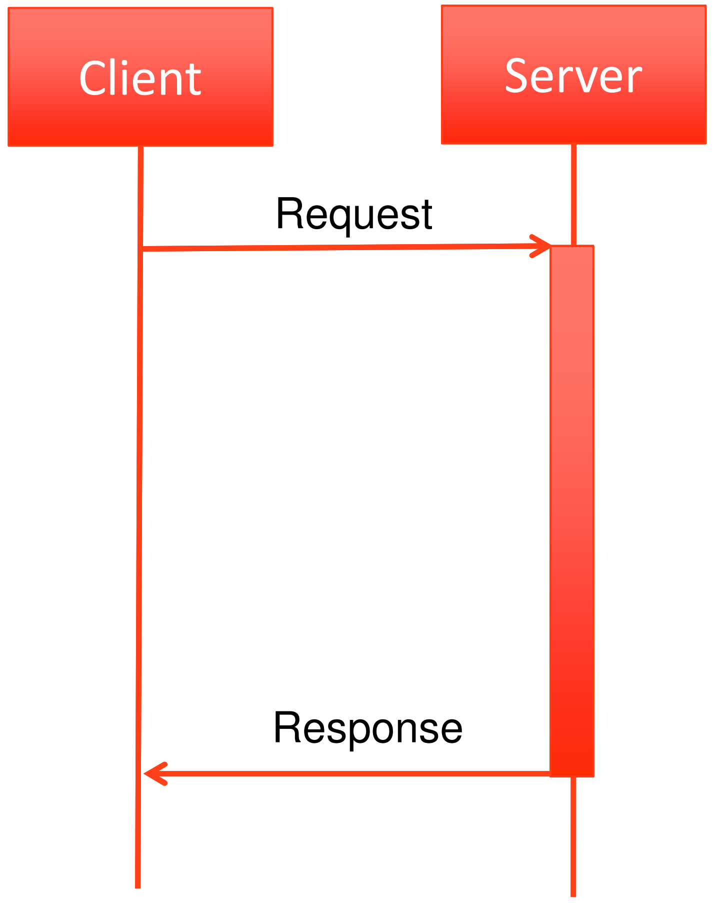
Reactive manifesto

- react to events (message driven)
- react to load (scalable)
- react to failure (resilient)
- react to users (responsive)

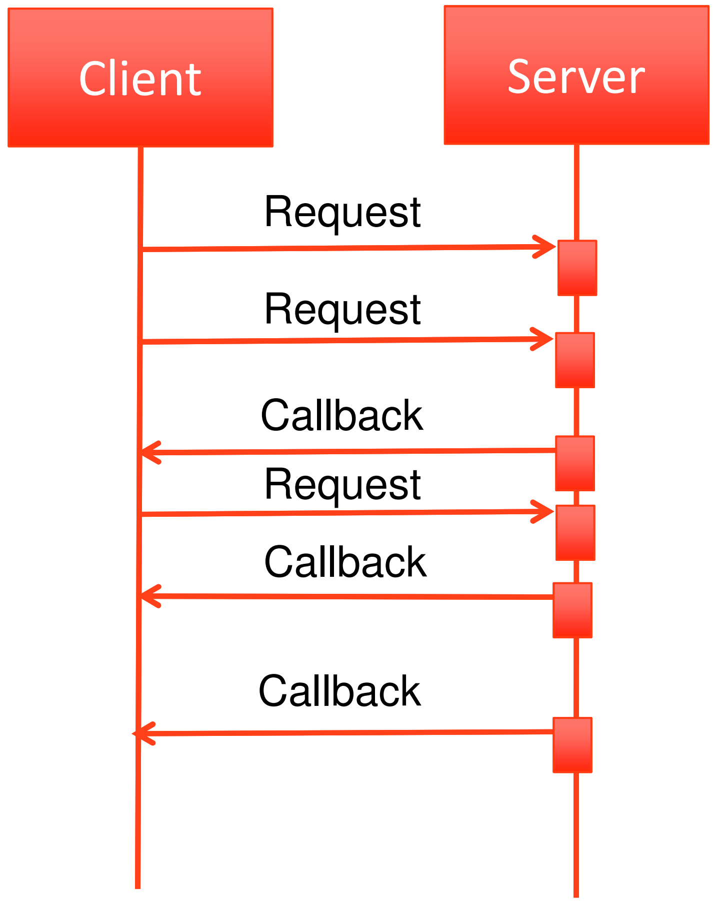


SOGETI

Blocking vs non-blocking



One thread per connection (1 client)



One thread per event-loop (multiple clients)

SOGETI

Vert.x

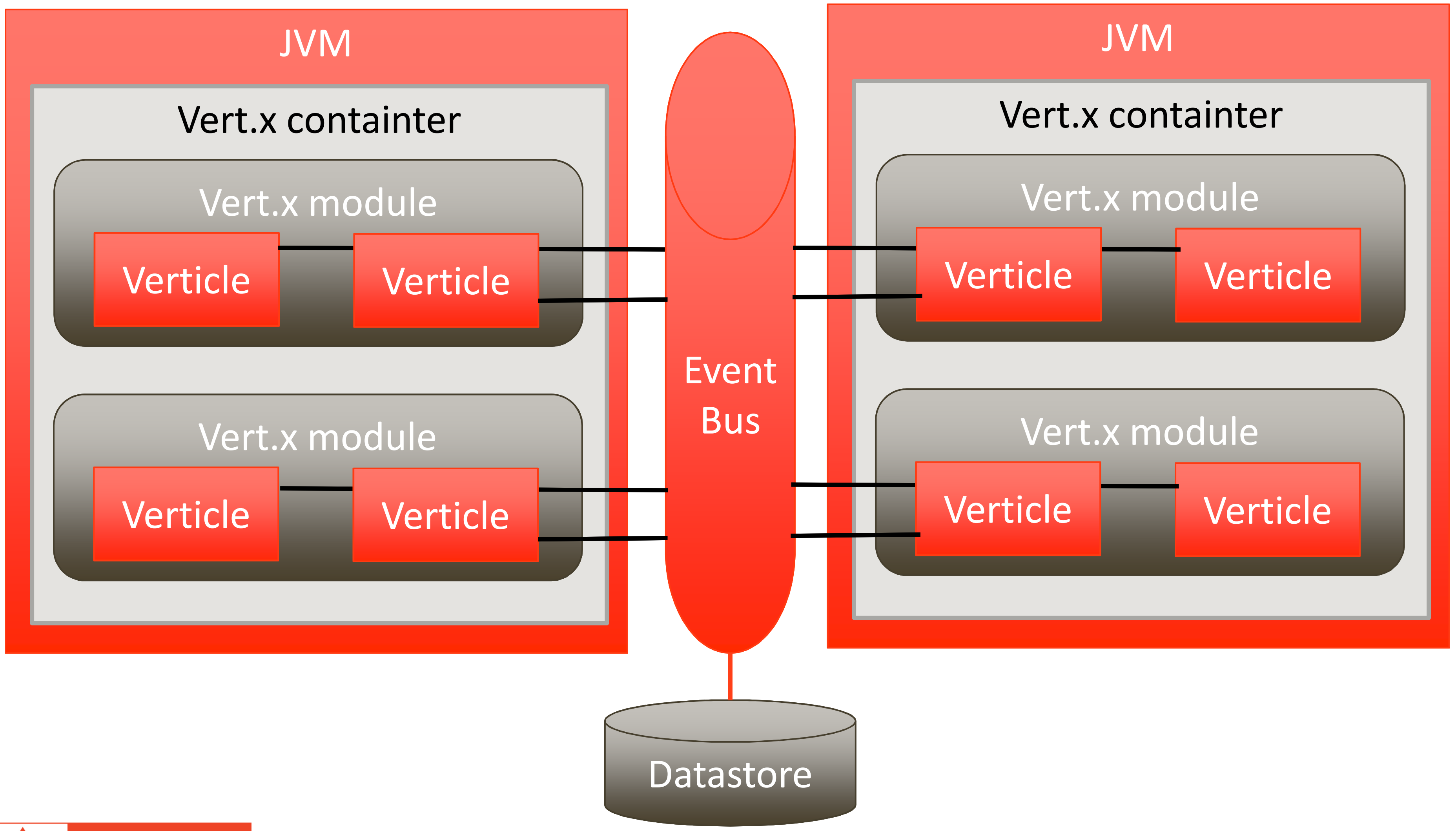
- c10k problem
- Polyglot, running on JVM
- Asynchronous and synchronous
- Scalable
- Distributed eventbus
- Thread per event-loop, non blocking
- Micro services

SOGETI

Popular Technologies

- AngularJS: Javascript MVC framework
- VertX: Asynchroon Polyglot JVM library
- MongoDB: NoSQL database

Vert.x

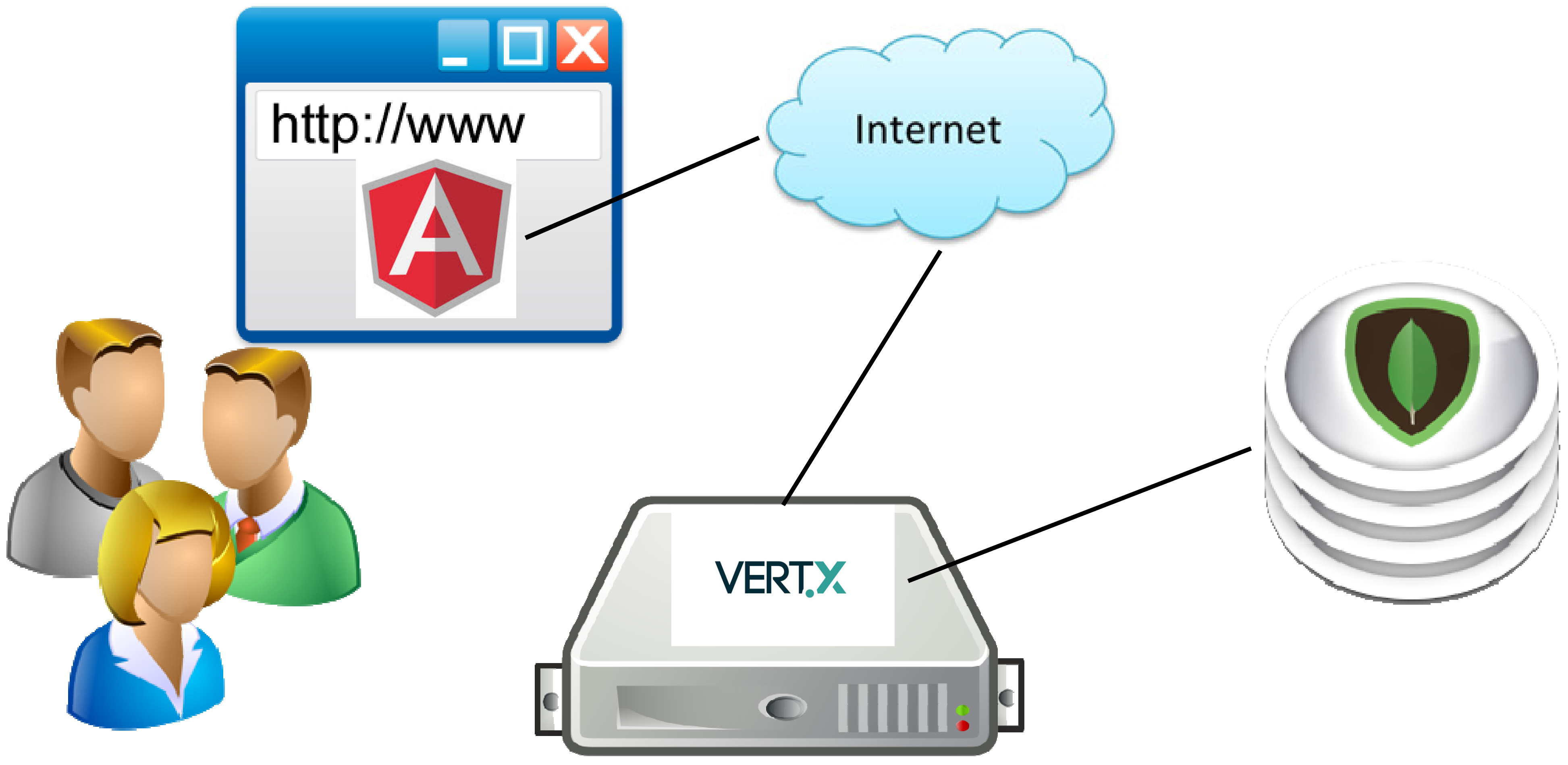


SOGETI

Chat application

- Send messages
- Receive messages
- Network communication
- Persist messages
- Sync missed messages

Architecture



Architecture

SOGETI

Getting started

- Java SDK (JDK) 8
- Maven
- IDE (Eclipse)
- MongoDB
- PATH settings (shell.bat)
- git (optional)

Archetype

- git clone https://github.com/erwindeg/vertx-mongo-angular-archetype
- mvn install

Clean install

- create folder workshop
- copy mvn, mongo, jdk, repo, settings.xml, shell
- change settings.xml, shell.bat, bin\mvn
- copy archetype-catalog.xml to <user-home>/.m2
- Run mvn –X to check

Create project

- Eclipse:
  - new maven project
  - select nl.edegier.vertx-mongo-angular-archetype
- Maven:
  - mvn archetype:generate
- Choose name (and version)
- mvn clean install

SOGETI

Start MongoDB

- mongo\bin\mongod.exe –dbpath data

Run it

- Eclipse:
  - run as java application
  - main class: io.vertx.core.Starter
  - run nl.sogeti.MainVerticle
- CLI:
  - Java –jar target\<jar-name>-fat.jar

SOGETI

Client angular main.html

```
<div ng-controller="MainCtrl">
    <select ng-model="selected" ng-options="message.text as
    message.name + ': ' + message.text for message in messages
    | orderBy:'date':true"
    multiple size="20" style="min-width: 90%;">
</select>
<form ng-submit="sendMessage()">
    <input type="text" ng-model="message.name"
    placeholder="Type your name here" /> <input type="text" ng-
    model="message.text" placeholder="Type your message here"/>

    <input type="submit" value="Send" />
</form>
</div>
```

Client angular main.js

```
angular.module('resourcesApp').controller('MainCtrl',
function($scope, $resource) {
        $scope.messages = [];
        var eb = new
vertx.EventBus('http://'+window.location.host+ '/eventbus');
    eb.onopen = function() {
        eb.registerHandler('chat', function(message) {
            $scope.messages.push(message);
            $scope.$apply();
        });
}
$scope.sendMessage = function() {
    $scope.message.date = Date.now();
    eb.publish('chat', $scope.message);
    $scope.message.text = "";
    };
});
```

SOGETI

Server MainVerticle.java

```java
HttpServer server = vertx.createHttpServer(new
HttpServerOptions().setPort(8080).requestHandler(req ->
matcher.accept(req));

SockJSServer.sockJSServer(vertx, server).bridge(new
SockJSServerOptions().setPrefix("/eventbus"),
new BridgeOptions().addInboundPermitted(new
JsonObject()).addOutboundPermitted(new JsonObject()));

server.listen();
```

SOGETI

Cluster mode

- –cluster –cluster-host <ip_address>

MongoDB persistence

```
private static final String MONGO_ADDRESS = 
UUID.randomUUID().toString();
MongoService proxy;

public void start() throws Exception {
proxy = setUpMongo();
…
private MongoService setUpMongo() {
    DeploymentOptions options = new
     DeploymentOptions().setConfig(new
     JsonObject().put("address", MONGO_ADDRESS));
    vertx.deployVerticle(new MongoServiceVerticle(), options,
     res -> System.out.println(res.result()));
    return MongoService.createEventBusProxy(vertx,
     MONGO_ADDRESS);
}
```

SOGETI

Save messages

```
 public void start() throws Exception {
    …
    vertx.eventBus().consumer("chat", this::saveMessages);
    …

 private void saveMessages(Message message) {
    proxy.insert("messages", new
    JsonObject(message.body().toString()), res ->
    System.out.println(res.succeeded()));
}
```

List messages

```
$scope.messages = $resource('/api/history').query();
```

```
matcher.matchMethod(HttpMethod.GET, "/api/history", req ->
proxy.find("messages", new JsonObject(), res ->
req.response().end(new JsonArray(res.result()).toString())));
```

Run it again

SOGETI

Mongo DB unique index

- mongo\bin\mongo.exe
- use default_db
- db.messages.createIndex({ date: 1, name: 1, text: 1 },{ unique : true })

SOGETI

Receive messages

```
private final String channel = UUID.randomUUID().toString();

public void start() throws Exception {
   …
   vertx.eventBus().consumer(this.channel, this::saveMessages);
   …
}

 private void sendHistoryRequest(AsyncResult<String> result){
    vertx.eventBus().publish("history", new
    JsonObject().put("channel", this.channel));
 }
```

SOGETI

Send messages

```
vertx.eventBus().consumer("history", m -> proxy.find("messages",
new JsonObject(), res -> sendMessages(((JsonObject)
m.body()).getString("channel"), res)));
```

```
  private void sendMessages(String channel,
     AsyncResult<List<JsonObject>> result) {
     if(!this.channel.equals(channel)){
         for (JsonObject message : result.result()) {
         System.out.println("sending message: "+message);
         vertx.eventBus().send(channel, message);
         }
     }
 }
```

SOGETI

Send request for messages

```
private MongoService setUpMongo() {
    …
    vertx.deployVerticle(new MongoServiceVerticle(), options,
    this::sendHistoryRequest);
    …
}
```

SOGETI

SOGETI

Unit Testing

```java
@RunWith(VertxUnitRunner.class)
public class MyJUnitTest {

    private Vertx vertx;

    @Before
    public void setUp(TestContext context) {
      Async async = context.async();
      vertx = Vertx.vertx();
      vertx.deployVerticle(MainVerticle.class.getName(), ar -> {
          if (ar.succeeded()) {
              async.complete();
          } else {
              context.fail("Could not deploy verticle");
          }
      });
    }
```

Unit Testing

```
@Test
    public void testHello(TestContext context) {
        Async async = context.async();
        HttpClient client = vertx.createHttpClient();
        HttpClientRequest req = client.get(8080, "localhost", "/app/test.html");
        req.exceptionHandler(err -> {
            context.fail();
        });
        req.handler(resp -> {
            context.assertEquals(200, resp.statusCode());
            Buffer entity = Buffer.buffer();
            resp.handler(entity::appendBuffer);
            resp.endHandler(v -> {
                context.assertEquals("test", entity.toString("UTF-8"));
                async.complete();
            });
        });
        req.end();
    }
```

SOGETI

Unit Testing

```
@After
   public void tearDown(TestContext context) {
        Async async = context.async();
        vertx.close(ar -> {
            async.complete();
        });
   }
```