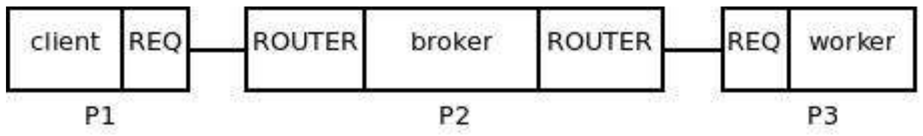
SECOND PARTIAL EXAM TSR (LABS)

This exam consists of 20 multiple-choice questions. In every case only one of the answers is the correct one. Answers should be given in a SEPARATE answer sheet you should have been provided with.

All questions have the same value. If correctly answered, they contribute 0,5 points to the final grade. If incorrectly answered, the contribution is negative, equivalent to 1/5th the correct value, which is -0,1 points. So, think carefully your answers. In case of doubt, leave them blank.

The exam can be completed within 1 hour.

Assume an application comprising 3 components: "client.js", "broker.js" and "worker.js". The components communicate via ZMQ sockets, using an architecture like the one described in Lab2:



This application is deployed launching one instance of each "`client.js`" (process P1), "`broker.js`" (process P2) and "`worker.js`" (process P3) in three different machines. Assume the client process' identity is string "`ADFG`", and the identity of the worker process string "`ERTH`". Further, assume that the broker saves no state for ongoing requests to workers. Answer the following questions concerning the application's behavior:

1. **Assume P1 sends string "`NEW REQUEST`" via its REQ socket, and will receive answer "`REQUEST PROCESSED`" through that same socket. Select the true assertion**

A	P3's program gets the following message from its REQ socket: ["ERTH","","ADFG","","NEW REQUEST"]
B	When P3 answers P1's request, P2 gets from its backend socket this message: ["ERTH","","ADFG","","REQUEST PROCESSED"]
C	P2 sends P1 the answer with message: ["ERTH","","ADFG","","NEW REQUEST"]
D	When P3 answers P1's request, P2 gets from its backend socket this message: ["ERTH","","REQUEST PROCESSED"]
E	All the above.
F	None of the above.

2. Assume now we deploy the application with an extra worker instance (process P4). Assume further that P1 sends 10 consecutive requests through its REQ socket. Select the right assertion

A	Having two instances of worker.js allows us to cut in half the average processing time of the 10 requests issued by P1
B	The frontend ROUTER socket of P2 allows concurrent sending of the 10 requests from P1, distributing them to the two worker processes (P3, P4), thus reducing the processing time of P1's requests
C	Socket REQ of P1 allows sending the various requests concurrently.
D	Having those two instances of worker.js will not help reducing the average processing time of the 10 requests issued by P1 from the case when we only have one instance of worker.js
E	All the above.
F	None of the above.

3. Assume we have now a different version of the broker, broker2.js, with a DEALER socket for the backend. Assume furthermore, that we have another version for the worker, worker2.js, using a REP socket to communicate with the backend from the broker.
Let us launch the application with one process for "client.js" (process P1), one for "broker2.js" (process P2) and another one for worker2.js (process P3). If P1 sends string "NEW REQUEST" via its REQ socket, select the right answer

A	When the request reaches P3 through its REP socket from P2, the message P3 gets is ["NEW REQUEST"]
B	P3's REP socket does not need an identity to communicate with P2
C	P1's REQ socket still needs an identity to communicate with P2
D	P2's DEALER socket (backend) needs to send the following message: ["ADFG","","NEW REQUEST"]
E	All the above.
F	None of the above.

Consider the following code fragment:

```
var zmq = require('zmq')
  , A = zmq.socket(X)
  , B = zmq.socket(Y);

A.bindSync('tcp://*:1111');
B.bindSync('tcp://*:2222');

A.on('message', function() {
  var args = Array.apply(null, arguments);
  B.send(args);
});

B.on('message', function() {
  var args = Array.apply(null, arguments);
  A.send(args);
});
```

4. The above code cannot be that of a broker if …

A	Clients connect with the broker via a REQ socket
B	X is 'router' and clients connect to port 1111 of the broker's node.
C	X is 'dealer' and workers connect to port 2222 of the broker's node.
D	Workers connect with a REP socket
E	All the above.
F	None of the above.

In the part **`zmqavanzado`** of lab 2, there are messages composed of 3, 4 and 5 frames

5. Messages received/sent by P2 through the `backend` socket are not 5 frames long when…

A	The worker reports its availability.
B	The worker reports an error.
C	The worker answers a request.
D	The message hasn't been sent by the worker.
E	All the above.
F	None of the above.

6. Messages received/sent by P2 through its `frontend` socket are not 3 frames long when…

A	The client makes a request
B	The broker returns a result
C	The broker returns an error
D	The client connects for the first time
E	All the above.
F	None of the above.

The **zmqexperto** section of Lab2 suggests using promises…

7. Promises are transmitted from the broker to the client…

A	Directly, as an additional message frame
B	Encoded with JSON, as an additional frame
C	Modifying one of the existing segments
D	In no way, as promises are global variables, and do not need to be transmitted
E	All the above.
F	None of the above.

8. In 0MQ …

A	Messages are strings of characters
B	Messages are arrays of values
C	Messages are dictionaries composed of key/value pairs
D	Messages are formed of various segments, delivered atomically
E	All the above.
F	None of the above.

9. In Lab2, …

A	The set of workers is fixed, and its size is given to the broker as a parameter
B	New workers can be added, sending their URL to the client processes
C	New workers can be added, sending their URL to the broker
D	The set of workers is fixed, and its size is passed to the clients as a parameter.
E	All the above.
F	None of the above.

10. In zmqexperto, the dynamic broker configuration is carried out through…

A	… the frontend socket of the broker
B	… the backend socket of the broker
C	… an additional REQ socket of the broker
D	… an additional PUB socket in the broker
E	All the above.
F	None of the above.

11. In Lab3, the command "`docker run -i -t tsir/baselab3`"

A	Creates a container running the shell on the `tsir/baselab3` image
B	Starts a virtual machine running, based on the `tsir/baselab3` image
C	Starts a sh process within a preexisting virtual machine running the `tsir/baselab3` image
D	Starts the sh program within the `tsir/baselab3` image running in the current environment.
E	All the above.
F	None of the above.

12. In Lab3 …

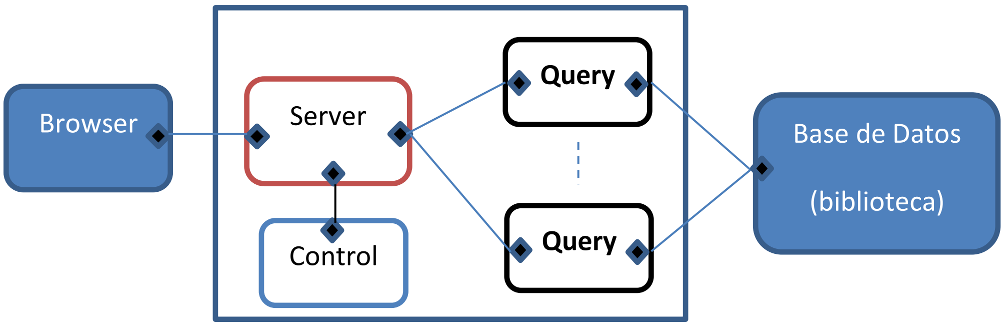
A	A component is defined by a directory containing a particular `Dockerfile`
B	The default file executed when a component's image is run cannot be changed by the developer of the component
C	The default file executed when a component's image is run is `index.js`, within the root of the component's directory.
D	The component does not need to specify a `package.json` file
E	All the above.
F	None of the above.

13. Assume we have deployed a service using docker containers with identifiers "client", "server", "server_1", "server_2"
We can stop the whole service with the command…

A	docker rm –f client server server_1 server_2
B	docker rm –f client server
C	docker rmi –f client server
D	docker rmc client server server_1 server_2
E	All the above.
F	None of the above.

Consider a web service with name QueryLibrary, which obtains information about a Library's database. The service consists of 3 components: ***Server***, managing the interaction with the browsers. ***Query***, managing accesses to the data base. And ***Control***, which, through a control port can dynamically modify the web's interface look and feel.

QueryBiblio



We want to deploy this application using the technologies of dependency resolution used in Lab3.

We have three directories: Components/Server, Components/Query and Components/Control, containing each the definition of the components. Assume the following file contents:

"Components/Server/config/default.js"

```
module.exports = {
  provides : {
     controlPort : 8001,
     queryPort : 8002
     },
  external : {
   webPort : 8000 }
}
```

"Components/Query/config/default.js"

```
module.exports = {
  requires : {
     serverUrl: tcp://localhost:8002
     },
  parameter : {
   BDServer : 192.168.1.1:8003 }
}
```

"Components/Control/config/default.js"

```
module.exports = {
  requires : {
     serverUrl: tcp://localhost:8001
     }
}
```

"Components/Control/server.js"

```
var utils = require('../utils.js')
……
```

"Components/Query/server.js"

```
var utils = require('../utils.js')
……
```

"Components/utils.js"

```
…
…
…
…
..
```

14. Select the right value for the “`links`” attribute within the service descriptor for QueryLibrary …

A	`links: {`
	`      Query : {  serverUrl : [ “Server”, “queryPort”] },`
	`      Control : { serverUrl : [ “Server”, “controlPort”] }`
	`}`

B	`links: {`
	`      Query : {  QueryUrl : [ “Server”, “queryPort”] },`
	`      Control : { ControlUrl : [ “Server”, “controlPort”] }`
	`}`

C	`links: {`
	`      Query : [ “Server”, “queryPort”] ,`
	`      Control : [ “Server”, “controlPort”]`
	`}`

D	`links: {`
	`      Query : {  queryPort : [ “Server”, “serverUrl”] },`
	`      Control : { controlPort : [ “Server”, “serverUrl”] }`
	`}`

E	All the above.
F	None of the above.

15. In the QueryLibrary service …

A	The values for the external references (in Server) can be modified in the service descriptor
B	The values for the ports specified within the “provides” attributes will remain unchanged for any deployment.
C	The values of the external references (in Server) can be changed in the deployment descriptor.
D	The value of the url specified within the “requires” attribute of “Components/Control/config/default.js” cannot change at deployment.
E	All the above.
F	None of the above.

16. After finishing the deployment of the QueryLibrary service …

A	The only running instances belong to the Server component
B	There are at most two running instances of the Query component
C	There are at most five running instances of the Control component
D	There is the same number of running instances of the Query and Control components
E	All the above.
F	None of the above.

17. The `noLocationDeployer` mentioned in lab3 is built modifying …

A	The main code of `deployer.js`
B	The `deploy` method of `Deployer`
C	The `buildImage` method of `Deployer`
D	The `configureComponent` function in `basicdeployer.js`
E	All the above.
F	None of the above.

18. The `noImageDeployer` mentioned in lab3 is bult modifying:

A	The main code of `deployer.js`
B	The `deploy` method of `Deployer`
C	The `buildImage` method of `Deployer`
D	The `configureComponent` function in `basicdeployer.js`
E	All the above.
F	None of the above.

19. The command '`docker run –t -r –entrypoint=/bin/cat tsir/balancer /app/config/default.js`':
NOTE: The –entrypoint option is equivalent to the Dockerfile ENTRYPOINT keyword.

A	Outputs to the console the contents of the default configuration file for component `balancer`
B	Runs component `balancer` normally
C	Ends with an error
D	Launches an instance of component `balancer`
E	All the above.
F	None of the above.

20. The command "`docker build –t tsir/balancer Components/balancer`"…

A	Builds image tsir/balancer for component `balancer` from the material found within directory `Components/balancer`
B	Crashes because it is not launched within the `Components/balancer` directory.
C	Launches the balancer component
D	Kills an instance of the balancer component
E	All the above.
F	None of the above.