

# Plateforme de compilation et d'exécution à distance, sécurisée et basée sur Docker

## Réalisé par :

- Salmane Bah
- Timothée Sollaud
- Tristan Braquelaire
- Wassim Romdan

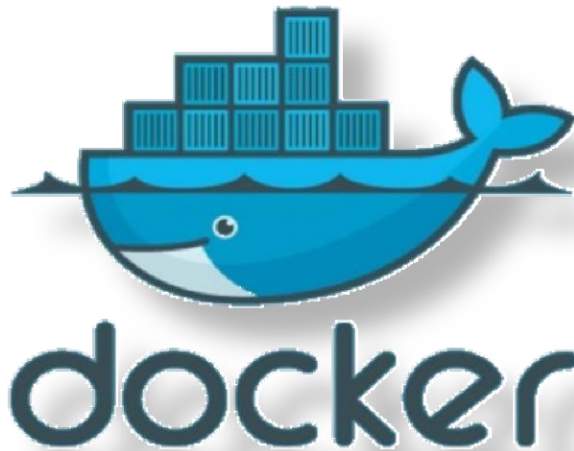


## Client :

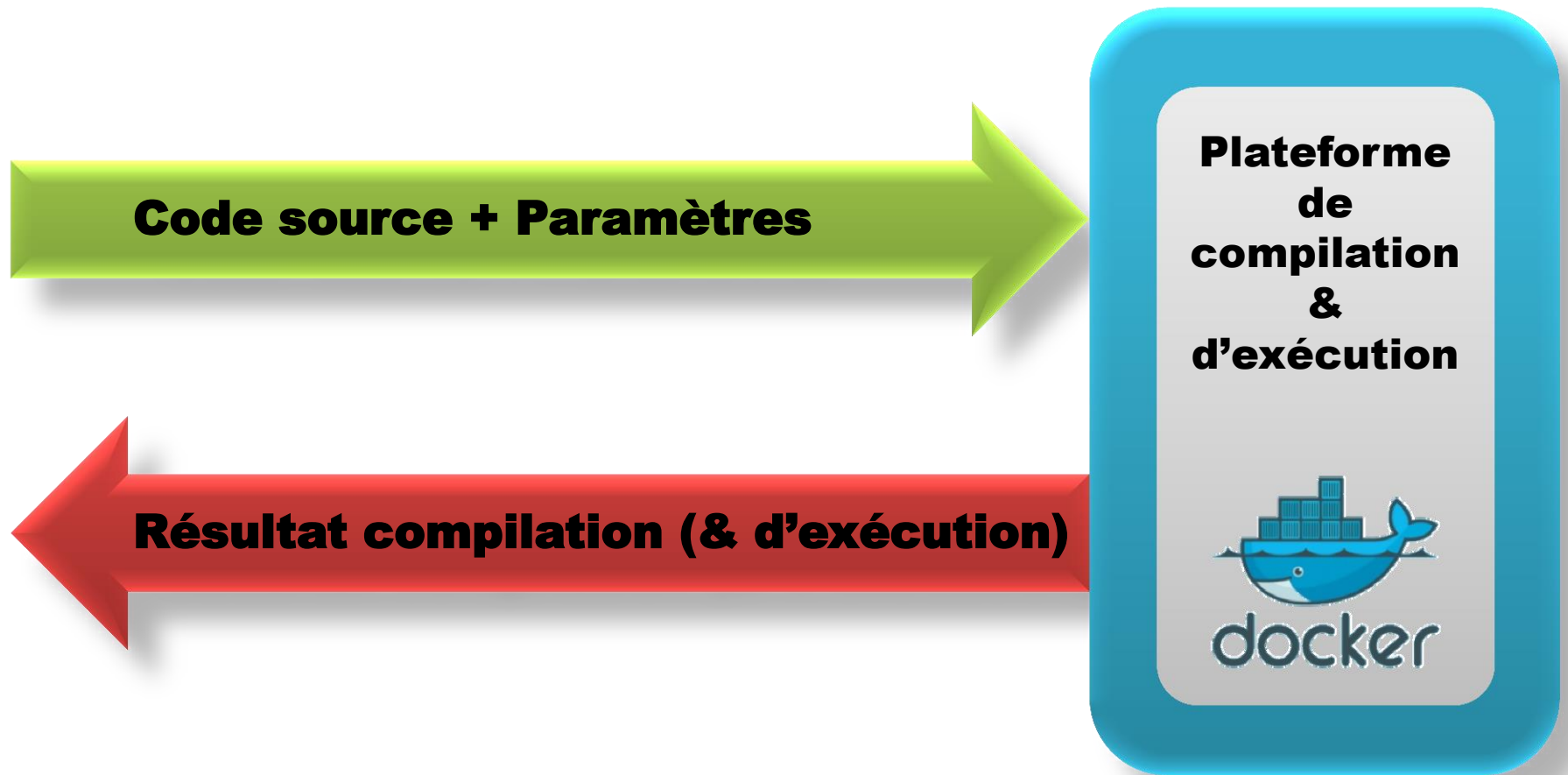
- M. Blin Guillaume

## Chargé de TD :

- M. Blanc Xavier



# Description du sujet



# Etude de l'existant : *Ideone.com*

ideone.com [new code](#) [samples](#) [recent codes](#) [sign in](#)

[edit](#) [fork](#) [download](#) [copy](#) <http://ideone.com/rxfKyk>

```
1.  /* package whatever; // don't place package name! */
2.
3.  import java.util.*;
4.  import java.lang.*;
5.  import java.io.*;
6.
7.  /* Name of the class has to be "Main" only if the class is public. */
8.  class Ideone
9.  {
10.     public static void main (String[] args) throws java.lang.Exception
11.     {
12.         // your code goes here
13.         System.out.println("hello world");
14.     }
15. }
```

language: **Java**  
created: 0 seconds ago  
visibility: public

[Share or Embed source code](#)

<script src="http://ideone.com/e.js/rxfKyk" type="text/javascript" ></script>

**Success** [comments \(0\)](#)

stdin [copy](#)  
Standard input is empty

stdout [copy](#)  
hello world

# Etude de l'existant : *compileonline.com*

The screenshot displays the compileonline.com web application. The header includes the site name and navigation links: Home, Web Editors, About, and Help. The main interface is divided into two primary sections: a code editor on the left and a results panel on the right.

**Code Editor:** The editor shows a Java class named `HelloWorld` with a `main` method that prints "Hello World". The code is as follows:

```
1 public class HelloWorld{
2
3     public static void main(String []args){
4         System.out.println("Hello World");
5     }
6 }
7
```

**Results Panel:** The results section, titled "Result", shows the compilation and execution process. It includes a "Download Files" link and the following output:

**Compiling the source code....**  
\$javac HelloWorld.java 2>&1

**Executing the program....**  
\$java -Xmx128M -Xms16M HelloWorld

Hello World

At the bottom of the interface, there are input fields for "Command Line Arguments:" and "STDIN Input:", both accompanied by help icons.

# Etude de l'existant : *compilr.com*

The screenshot displays the compilr.com web interface. At the top, the 'compilr' logo is on the left, and a user profile for 'Patrick Hankinson' is on the right. Below the header, there's a 'PROJECTS' sidebar on the left with a list of projects, including '20130211-Test', 'PHP: Arrays', 'PHP Website Example', 'C# Web Call', 'Java Fundamentals: Hello World', 'VB NET Example', 'PHP Info', 'Java Fundamentals: Variables and C', 'Java Fail', 'Java Fundamentals: Making Decisions', 'Math H', 'ASP.NET', 'Python 2.7 Web App', 'Java: Loops', 'Java Fundamentals: Classes & Objects', 'RIT Random', 'CSS Limit', and 'Zombie Dice'. The 'Zombie Dice' project is selected, showing its file structure: 'libraries', 'content', 'zombiedice.py', 'zombiedice\_web.py', 'jquery-1.8.3.min.js', 'imgZombieCheerleader.jpg', and 'imgTitle.png'. The main editor area shows the code for 'zombiedice.py', which is a Python script for a 'Zombie Dice Simulator desktop web app'. The code includes comments, imports, and a list of bots. Below the editor, a 'Console' panel shows the output of the build process, including messages like 'Checking for libraries... done', 'Building... done', 'Storing binaries <<< done', 'Listing build files... done', 'Checking for application data... done', and 'Running:'. A final message states 'Zombie Dice Visualization is running. Open your browser to http://localhost:60313 to view it. Press Ctrl-C to quit.' and a status bar at the bottom indicates 'Your application is now available at: http://54.198.0.167:60313'.

```
1 # Zombie Dice Simulator desktop web app
2 # By Al Sweigart al@inventwithpython.com
3 # Zombie Cheerleader photo by Gianluca Ramalho Misiti https://secure.flickr.com/photos/grmisiti/8149582049/
4
5
6 # oh god this code is awesome
7
8 import zombiedice
9
10 # =====
11 # Instructions for making your own bot can be found here: http://inventwithpython.com/blog/2012/11/21/how-to-make-ai-bots-for-z
12 # Assign the bots in the tournament here by adding "ZombieBot" objects to the BOTS list:
13
14 BOTS = [zombiedice.ZombieBot_MonteCarlo('MonteCarloBot', 40, 100),
15         zombiedice.ZombieBot_MonteCarlo('FastMonteCarloBot', 40, 20), # executes faster because it runs fewer experimental roll
16         zombiedice.ZombieBot_MinNumShotgunsThenStops('Min2ShotgunsBot', 2),
17         zombiedice.ZombieBot_MinNumShotgunsThenStops('Min1ShotgunBot', 1),
18         #zombiedice.ZombieBot_HumanPlayer('Human'), # uncomment if you want to play (learn the rules to Zombie Dice first thoug
19         zombiedice.ZombieBot_RollsUntilInTheLead('RollsUntilInTheLeadBot'),
20         zombiedice.ZombieBot_RandomCoinFlip('RandomBot'),
21     ]
22 # =====
```

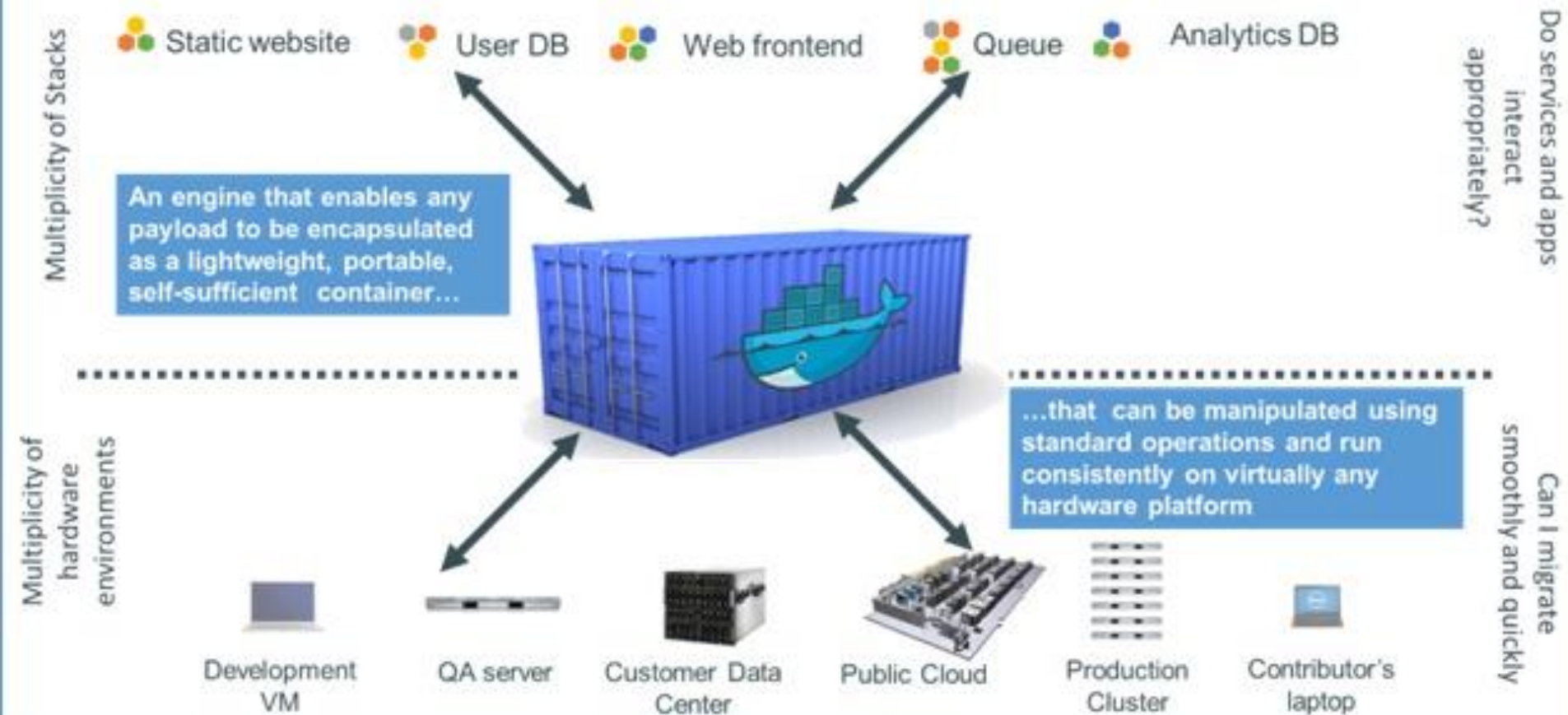
Line 1, Col 1 334 Lines Python Spaces 4

Console

Checking for libraries... done  
Building... done  
Storing binaries <<< done  
Listing build files... done  
Checking for application data... done  
Running:  
-----  
Zombie Dice Visualization is running. Open your browser to <http://localhost:60313> to view it.  
Press Ctrl-C to quit.  
✓ Your application is now available at: <http://54.198.0.167:60313>

# Docker & la sécurité

Docker is a shipping container system for code



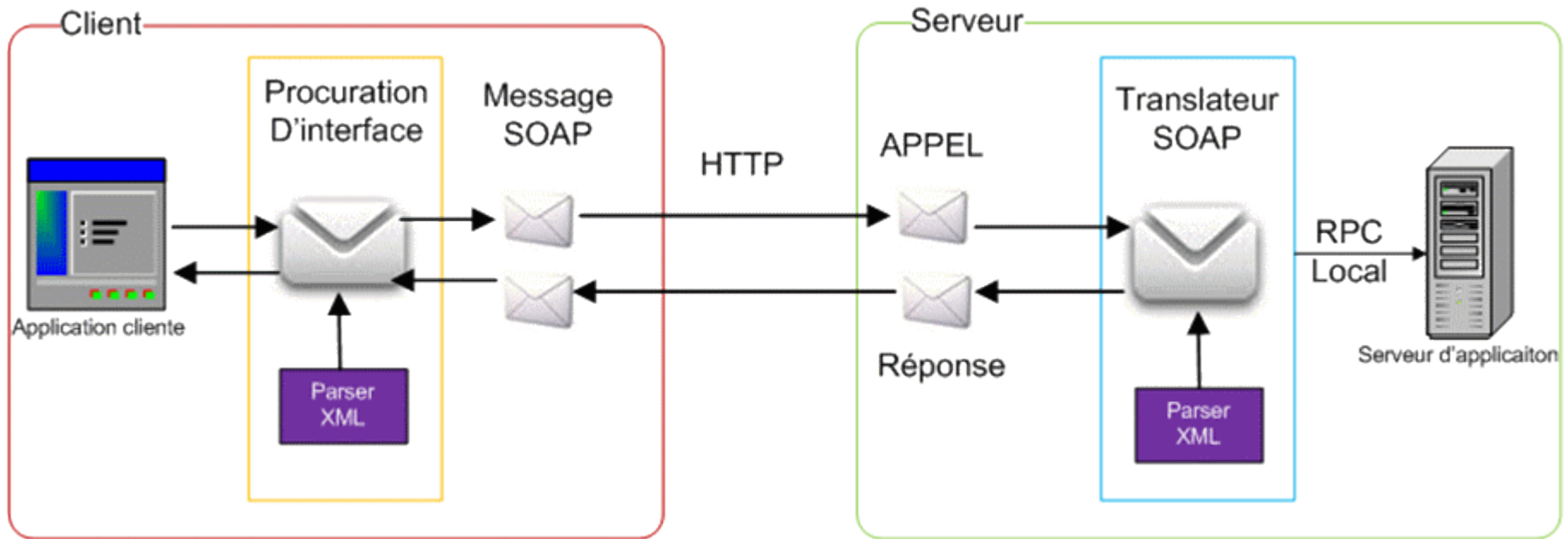


# Docker & la sécurité

- `--user=<username>`
- `--networking=<true/false>`
- `-m <space>`
- `--volume <path>`
- `timeout` (*fonction Linux*)



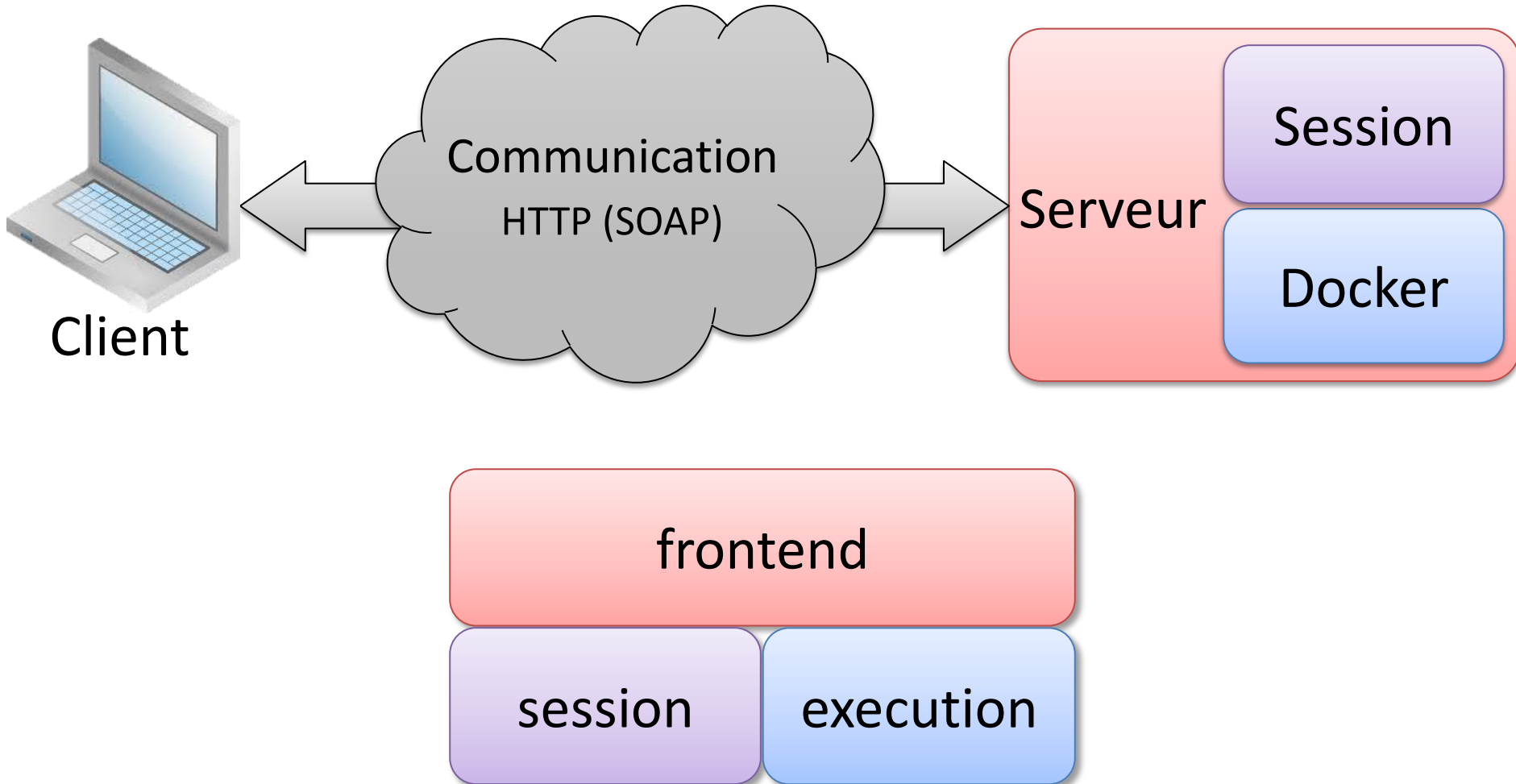
# Un Service Web (WS)



- XML & HTTP
- Indépendant du système
- Indépendant du langage
- Beaucoup de bibliothèques



# Architecture



# Architecture : *frontend*

RequestListener



RequestHandler

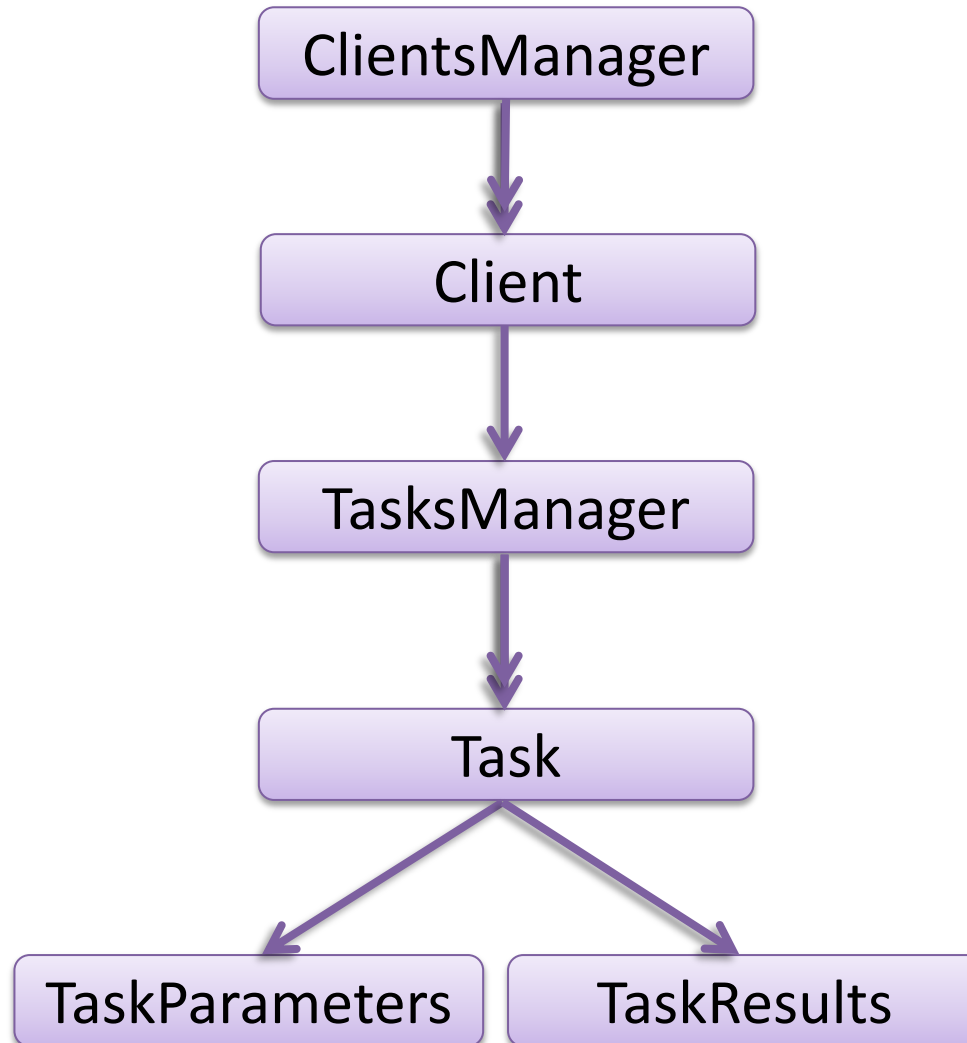
```
<message name="connect"/>
▼<message name="connectResponse">
  <part name="connectResult" type="tns:connectRet"/>
</message>
▼<message name="disconnect">
  <part name="sessionKey" type="xsd:string"/>
</message>
▼<message name="disconnectResponse">
  <part name="disconnectResult" type="tns:infoKey"/>
</message>
<message name="getLanguages"/>
▼<message name="getLanguagesResponse">
  <part name="getLanguagesResult" type="tns:getLanguagesRet"/>
</message>
```

```
@WebService(name="CompilePlatform" ,
  targetNamespace="http://pdp.compileplatform.frontend")
@SOAPBinding(style=Style.RPC)
public interface RequestListenerInterface {
  * web method to connect on the server in order to compile, interpret or
  @WebMethod
  @WebResult(partName="connectResult")
  ConnectRet connect();

  * web method to disconnect from the server , all client specific data
  @WebMethod
  @WebResult(partName="disconnectResult")
  InfoKey disconnect(@WebParam(partName="sessionKey") String sessionKey);

  * web method to retrieve languages supported by the platform , does not
  @WebMethod
  @WebResult(partName="getLanguagesResult")
  GetLanguagesRet getLanguages();
```

# Architecture : *session*

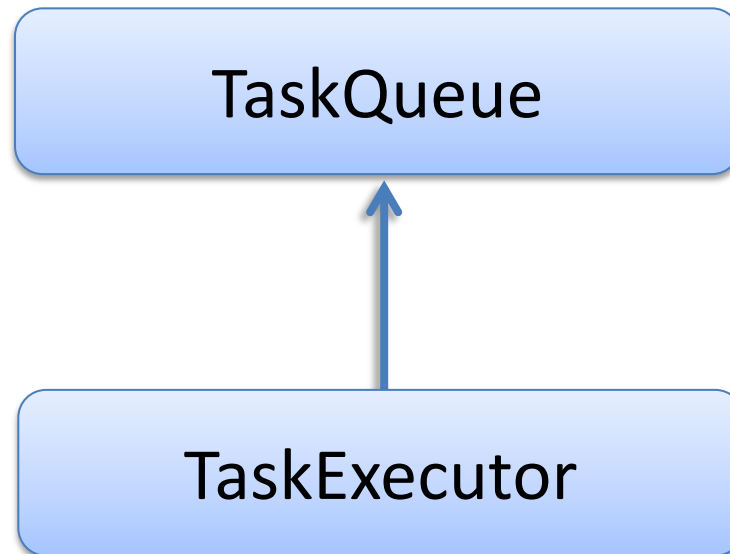


# Architecture : *session, le Cleaner*

```
/**
 * Clean directories for inactive sessions and tasks
 */
private void sessionCleaner() {
    Timer timer = new Timer();
    timer.schedule(new TimerTask() {
        @Override
        public void run() {
            RequestHandler.printLog("/!\ \ cleaning session(s)");
            List<ClientInterface> lstOldClients = new ArrayList<ClientInterface>();
            for(Entry<String, ClientInterface> entry : mClients.entrySet()) {
                ClientInterface client = entry.getValue();
                if (client.inactiveSince() > maxClientInactivityDuration){
                    lstOldClients.add(client);
                }
            }
            for (ClientInterface client : lstOldClients) {
                ClientsManager.this.removeClient(client);
            }

            RequestHandler.printLog("/!\ \ cleaning task(s)");
            for(Entry<String, ClientInterface> entry : mClients.entrySet()) {
                ClientInterface client = entry.getValue();
                client.cleanTasks(maxTaskInactivityDuration);
            }
        }
    }, sessionCleanerTimer, sessionCleanerTimer);
}
```

# Architecture : *execution*



# Architecture : *exécution, les langages*

```
public LangInternal(int idLang,
                    String langName,
                    String compiler,
                    boolean interpreted,
                    String extension,
                    String compileCommand,
                    String executionCommand)
{
    super(idLang, langName, compiler);
}

/**
 * construct a well formed compilation command
 * @param fileName the file name to use
 * @param compileOptions compilation options
 * @return a well formed compilation command depending on this language
 * (e.g {@code gcc -std=c99 src.c})
 */
public String generateCompileCommand(String fileName, String compileOptions){
    return compileCommand.replaceAll(opt, compileOptions).replaceAll(src, fileName);
}

languages[0] = new LangInternal(1 , "c" , "gcc " , false , ".c",
                                "gcc " + LangInternal.opt + " " + LangInternal.src,
                                "./a.out");
languages[2] = new LangInternal(3 , "python" , "python " , true , ".py",
                                null,
                                "python " + LangInternal.opt + " " + LangInternal.src);
languages[3] = new LangInternal(4 , "java" , "javac " , false , ".java",
                                "javac " + LangInternal.opt + " " + LangInternal.src,
                                "java " + LangInternal.src);
```

```
Pattern.compile("^public[ \\t\\n\\r]+class[ \\t\\n]+[a-zA-Z][a-zA-Z0-9_]*");
```



# Tests : *unitaires*

- JUnit
  - Le package session
- Problème détecté:
  - Gestion du temps

```
@Before
public void setUp()
{
    c = new Client("11001");
    String src = "#include <stdio.h>"
                + "void main(void){"
                + "    printf(\"Hello world !!\\n\\n\");"
                + "}";
    t1 = new Task(c, 1, src, "" , false, null, null);
    t2 = new Task(c, 1, src, "" , true, "", "");
    t3 = new Task(c, 1, src, "" , true, null, null);
}

@Test
public void testInactiveSince() {
    long time1 = t1.inactiveSince();
    try {
        Thread.sleep(2000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    long time2 = t1.inactiveSince();
    assertTrue(time1 < time2);
    t1.inActivity();
    long time3 = t1.inactiveSince();
    assertTrue(time2 > time3);
}
```

# Tests : *unitaires*

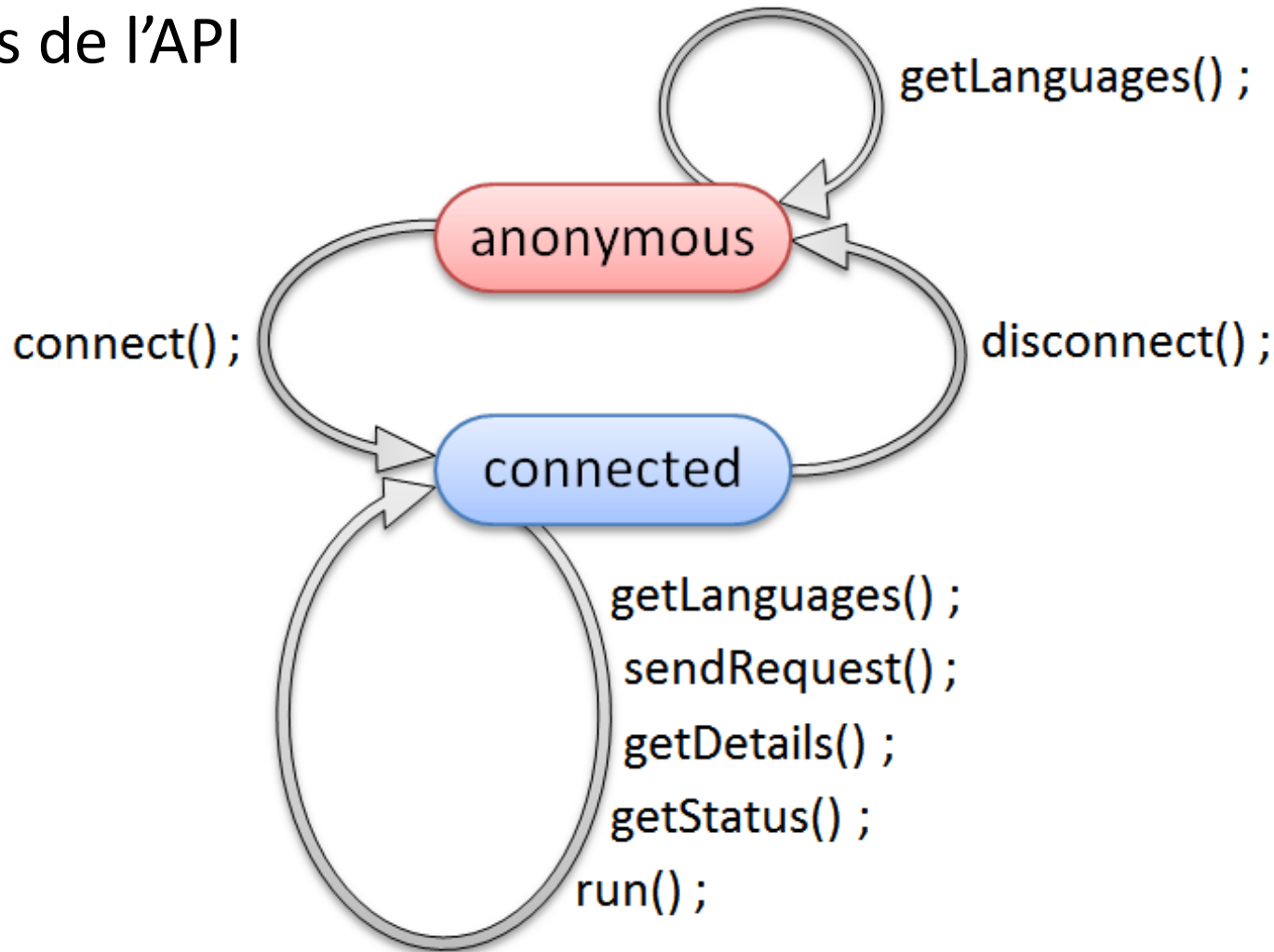
- Séparateurs dans les chemins

```
@Test
public void testKill() {
    ClientInterface myClient = new Client("1111111");
    TaskInterface t = new Task(myClient, 0, "test", "test", false, "test",
        "test");
    String oldDir = myClient.getPath();
    TaskInterface t1 = new Task(myClient, 0, "test", "test", false, "test",
        "test");
    TaskInterface t2 = new Task(myClient, 0, "test", "test", false, "test",
        "test");
    TaskInterface t3 = new Task(myClient, 0, "test", "test", false, "test",
        "test");
    TaskInterface t4 = new Task(myClient, 0, "test", "test", false, "test",
        "test");
    myClient.kill();
    File dir = new File(oldDir);

    assertFalse(dir.exists());
}
```

# Tests : *fonctionnels*

- Tests de l'API



# Tests : *fonctionnels*

- Client Python
  - API suds
- Problème résolu :
  - 1 requête = 1 port
  - Système d'identification par IP/Port invalide

# Tests : *fonctionnels*

- Test du cleaner

```
Server successfully loaded!
>>> Sat Apr 12 15:13:15 CEST 2014:
    getLanguages() => Languages sending
>>> Sat Apr 12 15:13:15 CEST 2014:
    connect() => Client connection :1b32145b2b4b5bf55a60f1d8d7ca1144febef1b7
...
>>> Sat Apr 12 15:13:16 CEST 2014:
    sendRequest() => Client successfully sends a request : 1b32145b2b4b5bf55a60f1d8d7ca1144febef1b7 : 0
>>> Sat Apr 12 15:13:16 CEST 2014:
    connect() => Client connection :ce6d3ad3d398871235f558e6f4eedd4903d0
...
>>> Sat Apr 12 15:13:17 CEST 2014:
    sendRequest() => Client successfully sends a request : ce6d3ad3d398871235f558e6f4eedd4903d0 : 0
>>> Sat Apr 12 15:13:17 CEST 2014:
    session deleted : /tmp/ce6d3ad3d398871235f558e6f4eedd4903d0
>>> Sat Apr 12 15:13:17 CEST 2014:
    disconnect() => Client successfully disconnected : ce6d3ad3d398871235f558e6f4eedd4903d0
```

```
kikouyou@kikouyou-K53SC:~$ tree /tmp/1b32145b2b4b5bf55a60f1d8d7ca1144febef1b7/
/tmp/1b32145b2b4b5bf55a60f1d8d7ca1144febef1b7/
├── 0
├── a.out
└── srcFile.c
```

# Tests : *fonctionnels*

- Test du cleaner

```
>>> Sat Apr 12 15:13:17 CEST 2014:
      disconnect() => Client successfully disconnected : ce6d3ad3d398871235f558e6f4eedd4903d0
>>> Sat Apr 12 15:18:01 CEST 2014:
      /!\ cleanning session(s)
>>> Sat Apr 12 15:18:01 CEST 2014:
      session deleted : /tmp/1b32145b2b4b5bf55a60f1d8d7ca1144febef1b7
>>> Sat Apr 12 15:18:01 CEST 2014:
      /!\ cleanning task(s)
```

```
kikouyou@kikouyou-K53SC:~$ ls /tmp/
hsperfdata_kikouyou  hsperfdata_root  pulse-ivq7A0C14aj0  suds
```



# Tests : *performance*

- Test de vitesse de traitement
  - Même algorithme dans 4 langages

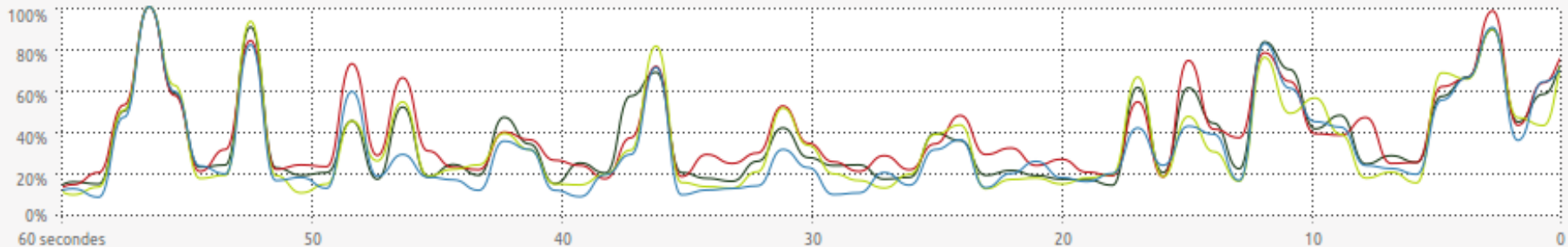
```
python_code = """import sys
i = int(sys.argv[1])
for j in range(0 , i) :
    print "Hello world from Docker , it rocks!, i'm the " + str(j) + " iteration"
line = sys.stdin.readlines();
print "the string is %s" %line
"""
```

	Temps de transfert	Temps de compilation	Temps d'exécution
Java	13	1236	544
C	15	577	456
C++	15	576	479
Python	8	525	522
Moyennes (ms)	12.75	728.5	500.25
Répartition du temps (%)	1.03	58.68	40.29

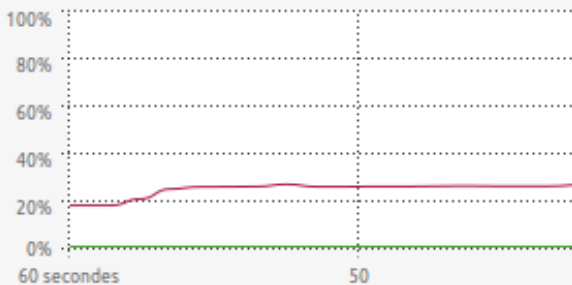
# Tests : *performance*

- Test de montée en charge
  - 1 Serveur
  - 3 Clients distants ouvrant 50 sessions de 32 requêtes.
  - Soit : **4800** requêtes lancées simultanément !

Historique d'utilisation du CPU



Historique d'utilisation de la mémoire physique



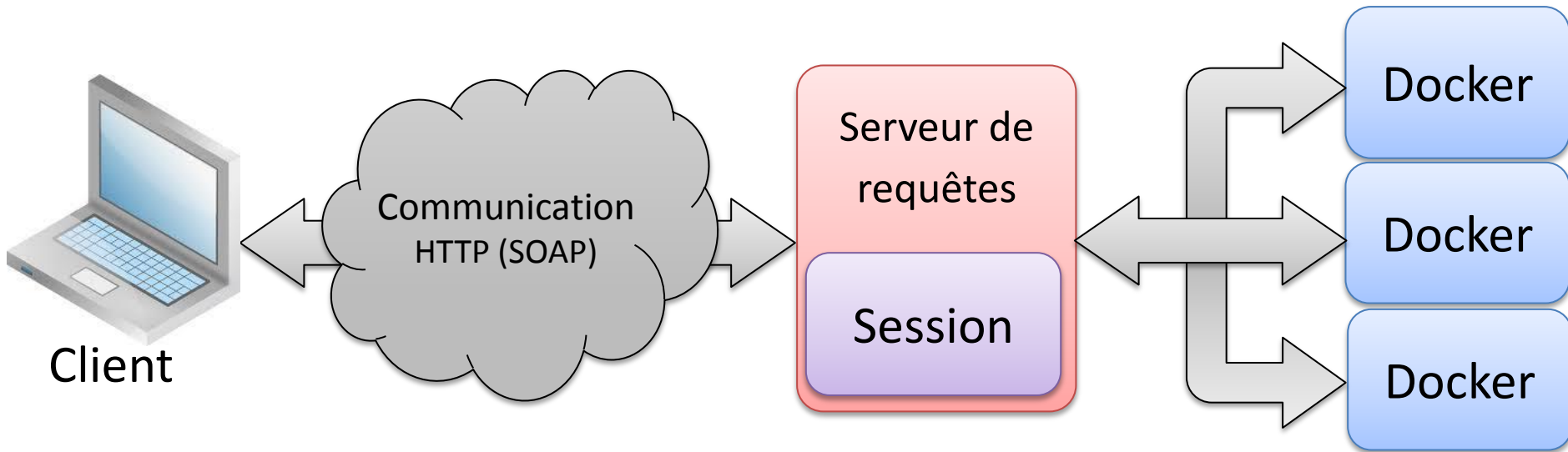
- Problème détecté et résolu:
  - Collisions du Hash de clé de session.

# Améliorations possibles

- Java Native Interface
  - Déléguer la partie Docker à du code C.
- API Java Logging
  - Utiliser la classe java dédiée à la gestion des logs.

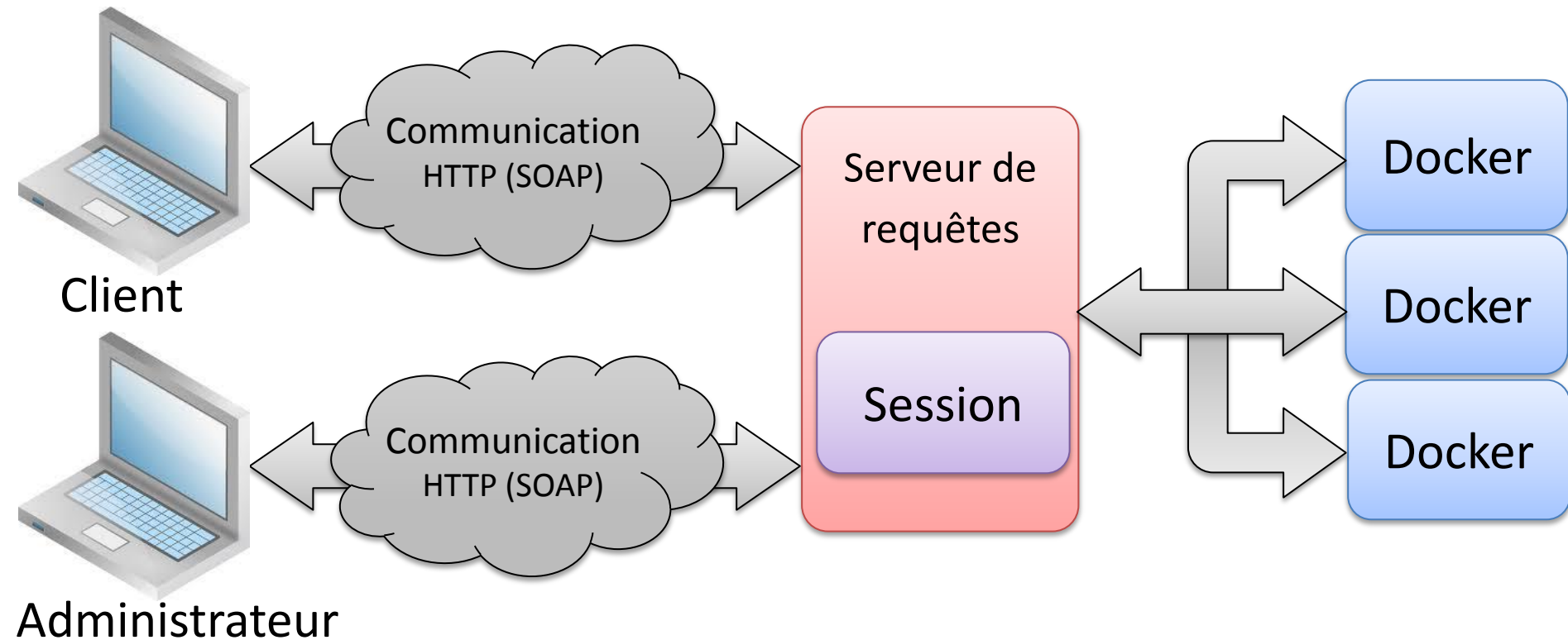
# Version 2

- Plusieurs serveurs physiques docker



# Version 3

- Administration à chaud



**Merci pour votre attention !**

université  
de **BORDEAUX**



**docker**