

Modularización con virtualización en AWS

Juan Nicolás Nontoa Caballero, Escuela Colombiana de Ingeniería, Servidor Web Concurrente

Abstract— The objective of this document is to define the architecture of the laboratory realized of modularization with virtualization of the course enterprise architectures. In this case, a web server was developed that would be able to receive and display web requests or images according to their file types, in this case "html" and "PNG" concurrently. The aim was to create a concurrent server that could interpret the two most basic types of requests that we can execute when we use a web server.

I. INTRODUCCION

Este documento se toma como referencia ayudando a entender la implementación realizada en el laboratorio, así ayudaría a comprender mejor en el entorno de diseño. Todo esto tomando como base los requisitos planteados al momento de iniciar este proyecto. Se definirá todos sus componentes detalladamente, entre los cuales se pueden observar a grandes rasgos de su arquitectura, diseño, implementación, documentación y variables definidas.

II. DISEÑO

Las capas de diseño se implementaron con base a los requerimientos y/o requisitos y necesidades sobre la marcha de la implementación. Siempre buscando generar aspectos de extensibilidad y estabilidad. Dicho lo anterior en diseño para el diseño que se implementó al laboratorio se implementaron 5 capas que se presentan a continuación.

A. *Socket*:

Se encarga meramente de crear e inicializar los Sockets, tanto el del Cliente, como el del servidor. Estos dos en sus clases correspondientes. Dichas clases son llamadas desde la clase principal, esta corre el servidor creando instancias de los sockets de estos.

B. *HttpServer*:

Esta capa construye la clase principal del proyecto que en este caso es "HttpServer" desde aquí corre el servidor llamando a los sockets y se empiezan a pedir solicitudes del cliente hacia el servidor. También se recorren los POJOS dentro de la capa de framework y extrae los métodos de estos alojándolos en un

atributo de tipo HashMap y extrae los parámetros de algún tipo de solicitud si este el caso.

C. *Paginas*:

Esta capa es la que genera la visualización de todas las solicitudes de tipo HTML, png y los POJOS dentro del servidor con anotaciones de tipo @web. es acá donde tanto se leen, cómo se interpretan, visualizan y publican, los resultados de cualquier solicitud en el servidor.

D. *Resources*:

En esta capa del proyecto, podemos hacer una referencia a la abstracción de memoria donde se encuentran alojadas los recursos HTML y PNG con los que cuenta el servidor para responder las múltiples solicitudes no concurrentes por parte de cualquier cliente.

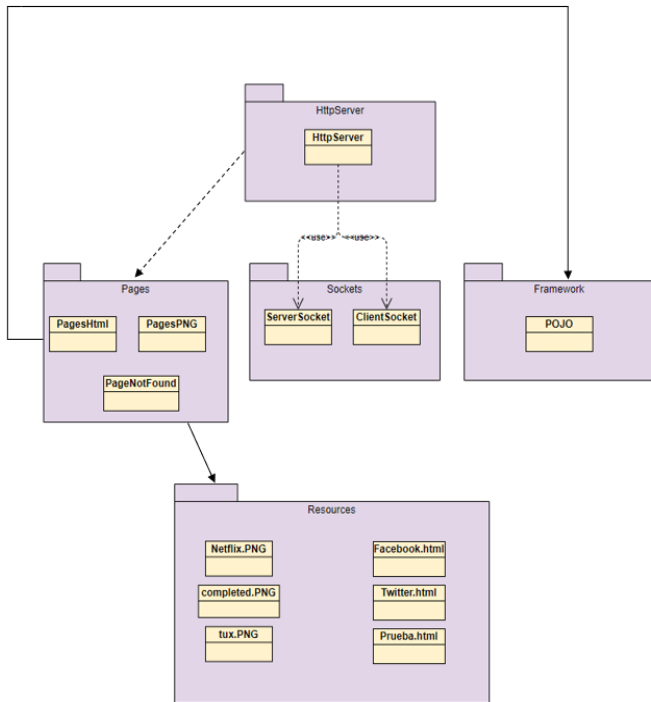
E. *FrameWork*:

En esta capa del proyecto se encuentran alojados los diferentes Pojos que se utilizaran para la construcción de aplicaciones web si el cliente lo solicita. Este será el único paquete en donde se podrán adicionar tantos Pojos como se quiera. Adicionalmente deberán de almacenar las diferentes anotaciones que se quieran implementar.

III. ESTRUCTURA DEL SERVIDOR WEB

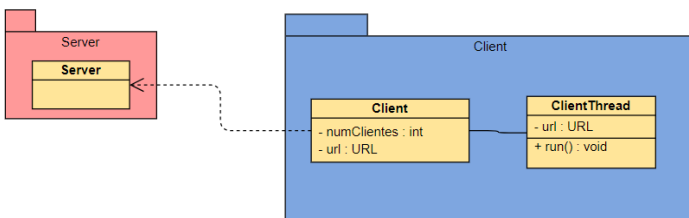
Este compuesto por los diferentes recursos, los cuales se podrán acceder directamente desde el navegador. De aquí se generan todos los vínculos a las demás dependencias del servidor, Incluyendo tanto las páginas HTML, como las imágenes usadas para visualizarlas. Para acceder a la visualización de los diferentes métodos con notación web (Pojos) se debe de colocar la extensión del método a llamar con su respectivo parámetro, como ejemplo

<https://proyectoarep.herokuapp.com/pojo/param:Nicolas>.



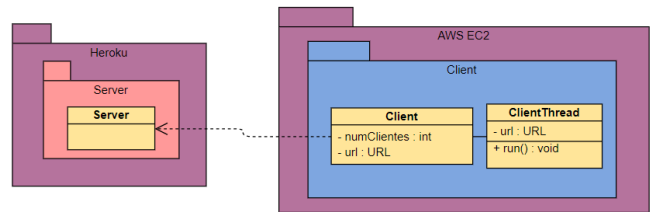
IV. ESTRUCTURA DEL CLIENTE

El cliente está compuesto por dos clases, una de ellas se encarga de administrar la cantidad de solicitudes que se realizarán en paralelo como también de a que URL se harán las solicitudes. La otra clase se encarga de leer los datos de la URL y mostrar su contenido.



V. ESTRUCTURA DEL SERVIDOR

El cliente realizará solicitudes desde una máquina virtual en AWS a un servidor desplegado en Heroku el cual estará en capacidad de recibir múltiples solicitudes.



VI. EXPERIMENTO

Servidor no concurrente vs Cliente concurrente:

En este caso como el servidor no es concurrente no tiene la capacidad de recibir múltiples solicitudes por lo cual solo responde la primera y las siguientes sin importar la cantidad responde con un error 503.

```

16 public static int numClientes=10;
17 public static URL url;
18
19 public static void main(String[] args) throws Exception {
20     url = new URL ("https://proyectoarep.herokuapp.com/index.html");
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Servidor concurrente vs Cliente concurrente:

En este caso como el servidor es capaz de recibir múltiples solicitudes responderá a cada una con el recurso solicitado. El cliente esta desplegado en una máquina virtual de AWS (EC2).

```

Simbolo del sistema - powershell
PS C:\Users\Nicolas\Downloads\LAB-AMS-AREP> ssh -i "myPrivateKey.pem" ec2-user@ec2-54-145-61-86.compute-1.amazonaws.com
Last login: Tue Oct 15 15:18:48 2019 from 45.239.88.78

_ _ _ _ _
| |   | |   | |
|_|   |_|   |_|
Amazon Linux AMI

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
$ package(s) needed for security, out of 7 available
$ sudo yum update -y to apply all updates.
[ec2-user@ip-172-31-24-195 ~]$ ls
ClientFromAWSToHeroku-1.0-SNAPSHOT.jar
[ec2-user@ip-172-31-24-195 ~]$ ls
ClientFromAWSToHeroku-1.0-SNAPSHOT.jar
[ec2-user@ip-172-31-24-195 ~]$ java -jar ClientFromAWSToHeroku-1.0-SNAPSHOT.jar

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>index</title>
  </head>
  <body>
    <center>
      <h1>Bienvenido al aplicativo</h1>
      <h2>Para ver una pagina o una imagen de click en el link.</h2>
      <ul>
        <li>
          <h2> Pages: </h2>
          <br>
          <a href=\\pp.html>Principal Page</a> <br> <br>
          <a href=\\Prueba.html>Test Page</a> <br> <br>
          <a href=\\notFound.html>Not Found Page</a> <br> <br>
        </li>
        <li>
          <h2> Images: </h2>
          <br>
          <a href=\\completed.png>Completed Image</a> <br> <br>
          <a href=\\Netflix.png>Netflix Image</a> <br> <br>
          <a href=\\tux.png>Tux Image</a> <br> <br>
        </li>
        <li>
          <h2> App: </h2>
          <br>
          <a href=\\pojo.html>App</a> <br> <br>
        </li>
      </ul>
    </center>
  </body>
</html>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>index</title>
  </head>
  <body>
    <center>
      <h1>Bienvenido al aplicativo</h1>

```

VII. CONCLUSION

Gracias al Desarrollo de este laboratorio se pudo apreciar de un modo más claro la importancia de delegar, diseñar y estructurar de una forma eficiente en las diferentes capas la implementación del servidor, dado que resultaría demasiado complejo la comunicación entre los diferentes sockets en solo una clase y la implementación de las diferentes solicitudes que se hagan. Se pudo observar la gran importancia que tienen los sockets a la hora de implementar y/o desarrollar un servidor web que, aunque si bien se sabe son de bajo nivel, ayudan notablemente en la comunicación Cliente-Servidor.

Juan Nicolás Nontoa Caballero nació en Tunja, Boyacá, Colombia en 1999. Graduado del Colegio San Jerónimo Emiliani en el 2015, inició sus estudios de Ingeniería de Sistemas en 2016 en la Escuela Colombiana de Ingeniería Julio Garavito.

```

Simbolo del sistema - powershell

      <h2> Images: </h2>
      <br>
      <a href=\\completed.png>Completed Image</a> <br> <br>
      <a href=\\Netflix.png>Netflix Image</a> <br> <br>
      <a href=\\tux.png>Tux Image</a> <br> <br>
    </li>
    <li>
      <h2> App: </h2>
      <br>
      <a href=\\pojo.html>App</a> <br> <br>
    </li>
  </ul>
</center>
</body>
</html>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>index</title>
  </head>
  <body>
    <center>
      <h1>Bienvenido al aplicativo</h1>
      <h2>Para ver una pagina o una imagen de click en el link.</h2>
      <ul>
        <li>
          <h2> Pages: </h2>
          <br>
          <a href=\\pp.html>Principal Page</a> <br> <br>
          <a href=\\Prueba.html>Test Page</a> <br> <br>
          <a href=\\notFound.html>Not Found Page</a> <br> <br>
        </li>
        <li>
          <h2> Images: </h2>
          <br>
          <a href=\\completed.png>Completed Image</a> <br> <br>
          <a href=\\Netflix.png>Netflix Image</a> <br> <br>
          <a href=\\tux.png>Tux Image</a> <br> <br>
        </li>
        <li>
          <h2> App: </h2>
          <br>
          <a href=\\pojo.html>App</a> <br> <br>
        </li>
      </ul>
    </center>
  </body>
</html>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>index</title>
  </head>
  <body>
    <center>
      <h1>Bienvenido al aplicativo</h1>
      <h2>Para ver una pagina o una imagen de click en el link.</h2>
      <ul>
        <li>
          <h2> Pages: </h2>
          <br>
          <a href=\\pp.html>Principal Page</a> <br> <br>

```