Ingeniería del Software: Laboratorio

PECL2 2020

David Barreiro Zambrano

Marcel Andrei Voicu

Carlos Javier Hellín Asensio

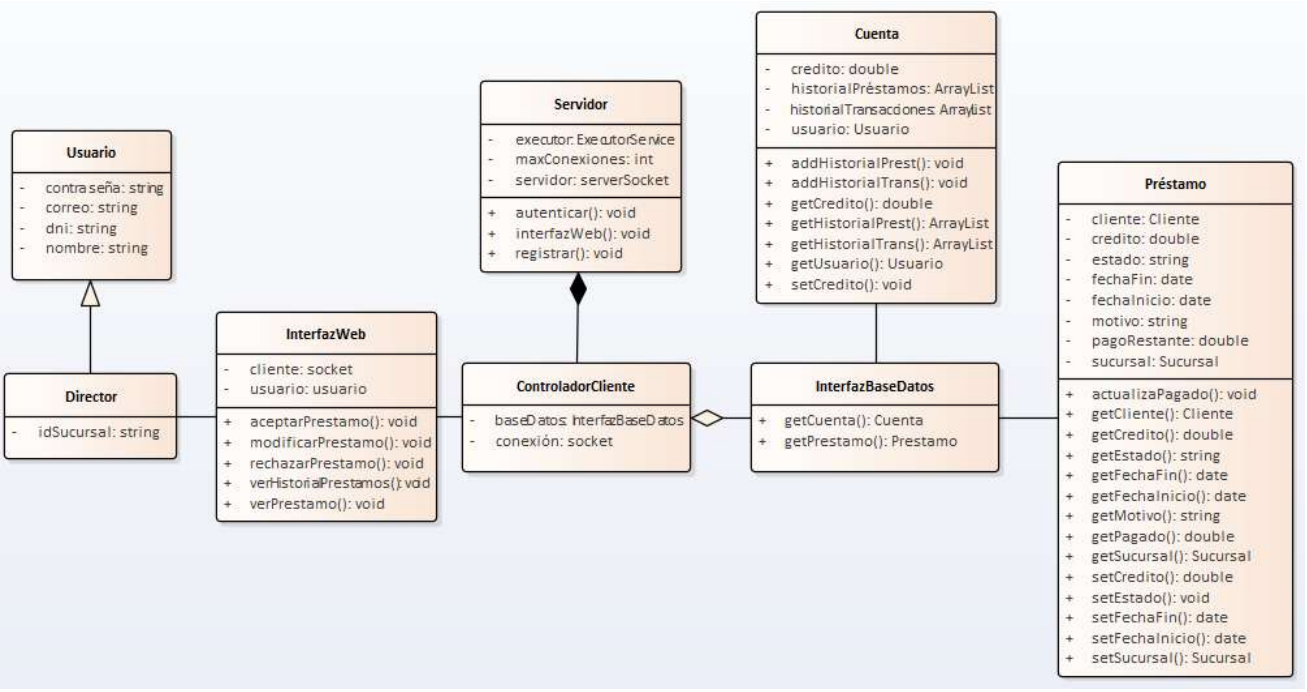
1. Fase de diseño: Sistema Informático Automatizado

Necesitamos diseñar un sistema que se ejecute en un servidor, que haga uso de una base de datos para almacenar información relacionada con los usuarios que utilizan el sistema y sus movimientos bancarios, y que los usuarios puedan acceder a este sistema a través de un ordenador o un teléfono móvil, utilizando un navegador web que les permita entrar al sitio web del sistema, el cual les servirá como interfaz de usuario. El sistema dará soporte a distintas funcionalidades, según el usuario que lo utilice: si lo utiliza un empleado de atención al cliente, podrá iniciar chats o llamadas con los clientes; si lo utiliza un director de sucursal, podrá gestionar a sus empleados y a las peticiones de préstamo de sus clientes.

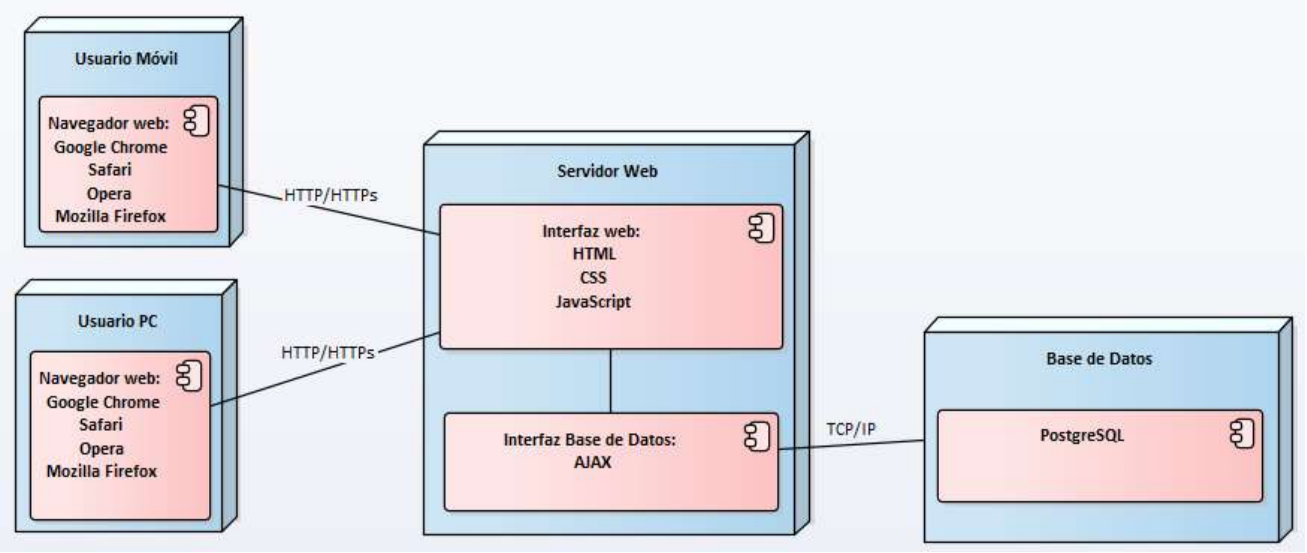
Un patrón de diseño que nos será útil es el patrón Facade, que permite realizar la comunicación entre los usuarios y el sistema de tal manera, que, para los clientes, el sistema sea una caja negra que ejecuta sus peticiones y la *fachada* un intermediario entre estos dos. Así desacoplamos el funcionamiento del sistema de sus usuarios, proporcionándonos mayor portabilidad del sistema y facilidad de reutilización y modificación. En este caso, lo implementaremos en la Interfaz Web, siendo esta la *fachada* entre el usuario y el sistema.

Evidentemente, el patrón Cliente/Servidor será necesario para el diseño de nuestro sistema.

1.1. Diagrama de Clases



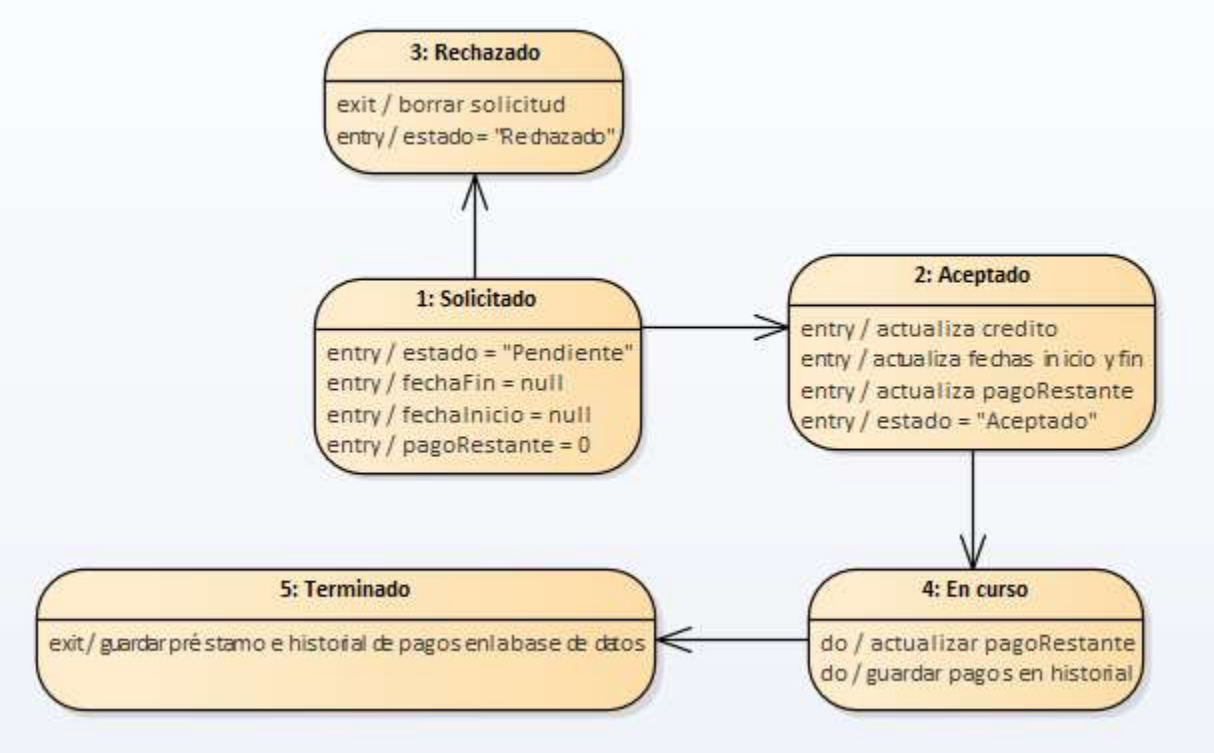
1.2. Diagrama de Despliegue



1.3. Diagrama de Estados

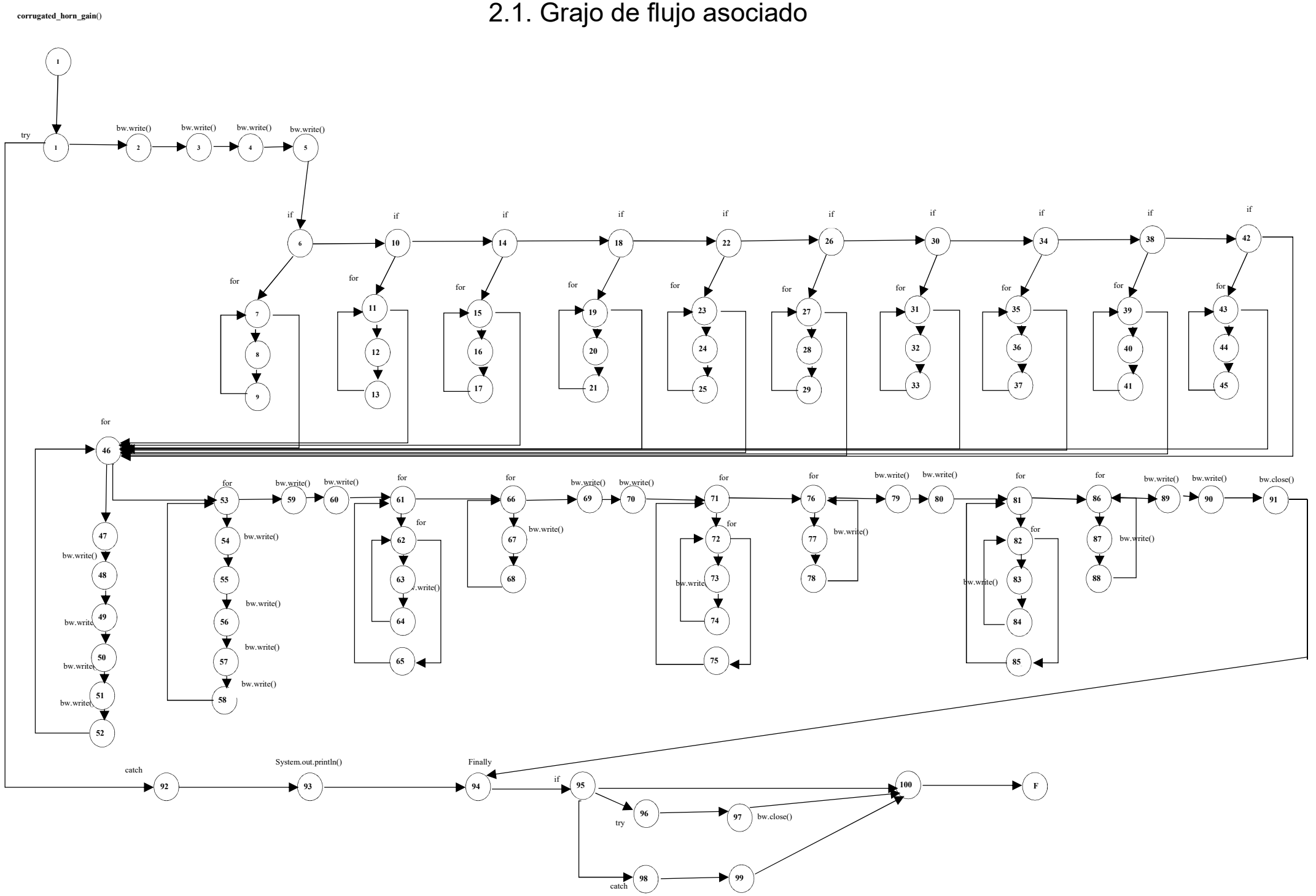
Un objeto Préstamo pasa por los siguientes estados:

1. Un cliente pide un préstamo a través de la aplicación web. Se crea marcado como “pendiente”, sin fecha de inicio ni de fin.
2. El director de la sucursal comprobará la solicitud de préstamo: si la acepta, el préstamo será marcado como “aceptado”, obtendrá una fecha de inicio y de fin, además de poder verse modificada la cantidad de crédito, según considere el director. Si no la acepta, será marcada como rechazada y, al cabo de 15 días, se borrará de la base de datos.
3. Una vez vencida la fecha de fin del préstamo y la deuda haya sido pagada, se eliminará de la base de datos.



2. Complejidad ciclomática y caminos básicos

2.1. Grajo de flujo asociado



2.2. Complejidad ciclomática

En la figura anterior se muestra un trozo de código con su grafo asociado en el que se identifican 36 posibles caminos. Aplicando la fórmula: *v(g) = e-n+2*, donde *e* representa el número de aristas y *n* el número de nodos, esto es, el número de posibles caminos del código.

Por tanto su **complejidad ciclomática es 36.**

Ya que tiene 136 instrucciones(e) y 102 nodos(n).

(*v(g)* = 136-102+2 = 36).

Por tanto, encontramos 36 caminos independientes.

2.3. Conjunto de caminos básicos

La selección de caminos básicos es la siguiente.

1. C1: 1,92,93,94,95,100.
2. C2: 1,92,93,94,95,96,97,100.
3. C3: 1,92,93,94,95,98,99,100.
4. C4:1,2,3,4,5,6,7,8,9,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.
5. C5:1,2,3,4,5,6,7,8,9,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.
6. C6:1,2,3,4,5,6,7,8,9,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.
7. C7:1,2,3,4,5,6,10,11,12,13,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.
8. C8:1,2,3,4,5,6,10,11,12,13,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.
9. C9:1,2,3,4,5,6,10,11,12,13,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.
10. C10:1,2,3,4,5,6,10,14,15,16,17,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.
11. C11:1,2,3,4,5,6,10,14,15,16,17,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.
12. C12:1,2,3,4,5,6,10,14,15,16,17,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.
13. C13:1,2,3,4,5,6,10,14,18,19,20,21,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.
14. C14:1,2,3,4,5,6,10,14,18,19,20,21,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.
15. C15:1,2,3,4,5,6,10,14,18,19,20,21,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.
16. C16:1,2,3,4,5,6,10,14,18,22,23,24,25,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.
17. C17:1,2,3,4,5,6,10,14,18,22,23,24,25,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.

18. C18:1,2,3,4,5,6,10,14,18,22,23,24,25,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.

19. C19:1,2,3,4,5,6,10,14,18,22,26,27,28,29,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.

20. C20:1,2,3,4,5,6,10,14,18,22,26,27,28,29,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.

21. C21:1,2,3,4,5,6,10,14,18,22,26,27,28,29,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.

22. C22:1,2,3,4,5,6,10,14,18,22,26,30,31,32,33,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.

23. C23:1,2,3,4,5,6,10,14,18,22,26,30,31,32,33,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.

24. C24:1,2,3,4,5,6,10,14,18,22,26,30,31,32,33,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.

25. C25:1,2,3,4,5,6,10,14,18,22,26,30,34,35,36,37,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.

26. C26:1,2,3,4,5,6,10,14,18,22,26,30,34,35,36,37,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.

27. C27:1,2,3,4,5,6,10,14,18,22,26,30,34,35,36,37,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.

28. C28:1,2,3,4,5,6,10,14,18,22,26,30,34,38,39,40,41,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.

29. C29:1,2,3,4,5,6,10,14,18,22,26,30,34,38,39,40,41,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.

30. C30:1,2,3,4,5,6,10,14,18,22,26,30,34,38,39,40,41,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.

31. C30:1,2,3,4,5,6,10,14,18,22,26,30,34,38,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94, 95,100.

32. C31:1,2,3,4,5,6,10,14,18,22,26,30,34,38,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94, 95,96,97,100.

33. C33:1,2,3,4,5,6,10,14,18,22,26,30,34,38,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94, 95,98,99,100.

34. C34:1,2,3,4,5,6,10,14,18,22,26,30,34,38,42,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,100.

35. C35:1,2,3,4,5,6,10,14,18,22,26,30,34,38,42,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,96,97,100.

36. C36:1,2,3,4,5,6,10,14,18,22,26,30,34,38,42,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,94,95,98,99,100.

2.4. Casos de prueba

CAMINO	ENTRADA	SALIDA
1	TRY:ERROR, bw=null.	Print:Error
2	TRY:ERROR, bw=!null	(Se cierra el buffer)
3	TRY:ERROR, bw=!null,try:error.	-
4	Profile_index=1, bw=null	az[i]=ai+(ao-ai)*(i*p/L)
5	Profile_index=1, bw=!null	az[i]=ai+(ao-ai)*(i*p/L)
6	Profile_index=1, bw=!null, try:error.	-
7	Profile_index=2, bw=null	az[i]=ai+(ao-ai)*((1-A)*i*p/L+A*Math.pow(Math.sin(Math.PI*i*p/(2*L)),rho))
8	Profile_index=2, bw=!null	az[i]=ai+(ao-ai)*((1-A)*i*p/L+A*Math.pow(Math.sin(Math.PI*i*p/(2*L)),rho))
9	Profile_index=2, bw=!null, try:error.	-
10	Profile_index=3, bw=null	az[i]=ai+(ao-ai)*((1-A)*i*p/L+A*Math.pow(Math.tan(Math.PI*i*p/(4*L)),rho))
11	Profile_index=3, bw=!null	az[i]=ai+(ao-ai)*((1-A)*i*p/L+A*Math.pow(Math.tan(Math.PI*i*p/(4*L)),rho))
12	Profile_index=3, bw=!null, try:error.	-
13	Profile_index=4, bw=null	az[i]=ai+(ao-ai)*((1-A)*i*p/L+A*Math.pow(i*p/(L),rho));
14	Profile_index=4, bw=!null	az[i]=ai+(ao-ai)*((1-A)*i*p/L+A*Math.pow(i*p/(L),rho));
15	Profile_index=4, bw=!null, try:error.	-
16	Profile_index=5, bw=null	az[i]=ai*Math.exp(Math.log(ao/ai)*i*p/L)
17	Profile_index=5, bw=!null	az[i]=ai*Math.exp(Math.log(ao/ai)*i*p/L)
18	Profile_index=5, bw=!null, try:error.	-
19	Profile_index=6, bw=null	az[i]=Math.sqrt(Math.pow(ai,2)+(Math.pow(i*p,2)*(Math.pow(ao,2)-Math.pow(ai,2)))/(Math.pow(L,2)))
20	Profile_index=6, bw=!null	az[i]=Math.sqrt(Math.pow(ai,2)+(Math.pow(i*p,2)*(Math.pow(ao,2)-Math.pow(ai,2)))/(Math.pow(L,2)))
21	Profile_index=6, bw=!null, try:error.	-
22	Profile_index=7, bw=null	az[i]=ai+(rho+1)*(ao-ai)*(1-(rho*i*p)/((rho+1)*L))*Math.pow(i*p/L,rho)
23	Profile_index=7, bw=!null	az[i]=ai+(rho+1)*(ao-ai)*(1-(rho*i*p)/((rho+1)*L))*Math.pow(i*p/L,rho)
24	Profile_index=7, bw=!null, try:error.	-
25	Profile_index=8, bw=null	az[i]= 0
26	Profile_index=8, bw=!null	az[i]= 0
27	Profile_index=8, bw=!null, try:error.	-
28	Profile_index=9, bw=null	az[i]= 0
29	Profile_index=9, bw=!null	az[i]= 0
30	Profile_index=9, bw=!null, try:error.	-
31	Profile_index<1, bw=null	az[i]=ai+(ao-ai)*(i*p/L)
32	Profile_index<1, bw=!null	az[i]=ai+(ao-ai)*(i*p/L)
33	Profile_index<1, bw=!null, try:error.	-
34	Profile_index=??, bw=null	az[i]=ai+(ao-ai)*(i*p/L)
35	Profile_index=??, bw=!null	az[i]=ai+(ao-ai)*(i*p/L)
36	Profile_index=??, bw=!null, try:error.	-

3. Ingeniería Inversa

Se realiza ingeniería inversa sobre el código Java propuesto para averiguar su lógica y obtener un diseño de alto nivel. Además, se realiza unas pruebas de “caja blanca” como se explica en las siguientes subsecciones.

3.1. Diseño de alto nivel

1. Buscamos el programa principal.

Se busca el programa principal pero no existe un Main como tal, aunque el código Java hace referencia a una clase NewMain, por lo tanto, y en este caso, se inspecciona el procedimiento llamado corrugated_horn_gain.

2. Inspeccionamos la primera rutina y se examina.

void corrugated_horn_gain(double fmin, double fmax, double alfa, double D)

Se trata de un procedimiento, es decir, una rutina que no devuelve ningún resultado. Tiene como entrada 4 argumentos:

- fmin de tipo double: frecuencia operativa más baja en GHz
- fmax de tipo double: frecuencia operativa más alta en GHz
- alfa de tipo double: ángulo de destello
- D de tipo double: diámetro de abertura

Una de las variables locales más importantes es profile_index que puede modificarse para que la rutina tenga diferentes perfiles dependiendo del valor asignado:

1 - perfil lineal

2 - perfil sinusoide

3 - perfil tangencial

4 - perfil x^rho

5 - perfil exponencial

6 - perfil hiperbólico

7 - perfil polinómico

Junto al profile_index se encuentra los parámetros de perfil que son utilizados dependiendo del valor de profile_index. La variable local A que solo puede ser un valor entre 0 y 1 afecta solo a los perfiles 2, 3 y 4. Mientras que rho es un valor entre 0.5 y 5 pero normalmente es 2, afectando a los perfiles 2, 3, 4 y 7.

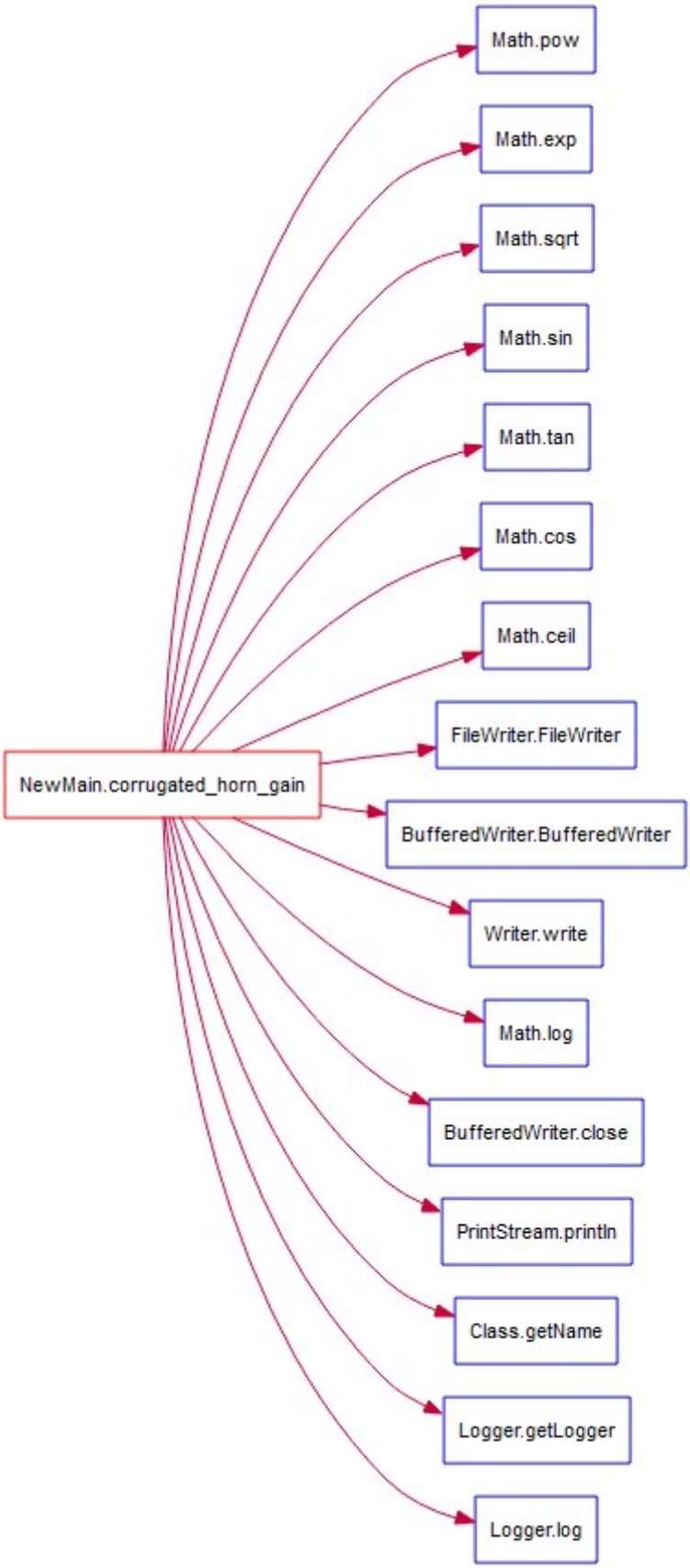
3. Inspeccionamos las rutinas llamadas por la primera rutina del programa principal, que entre ellas se encuentran las siguientes.

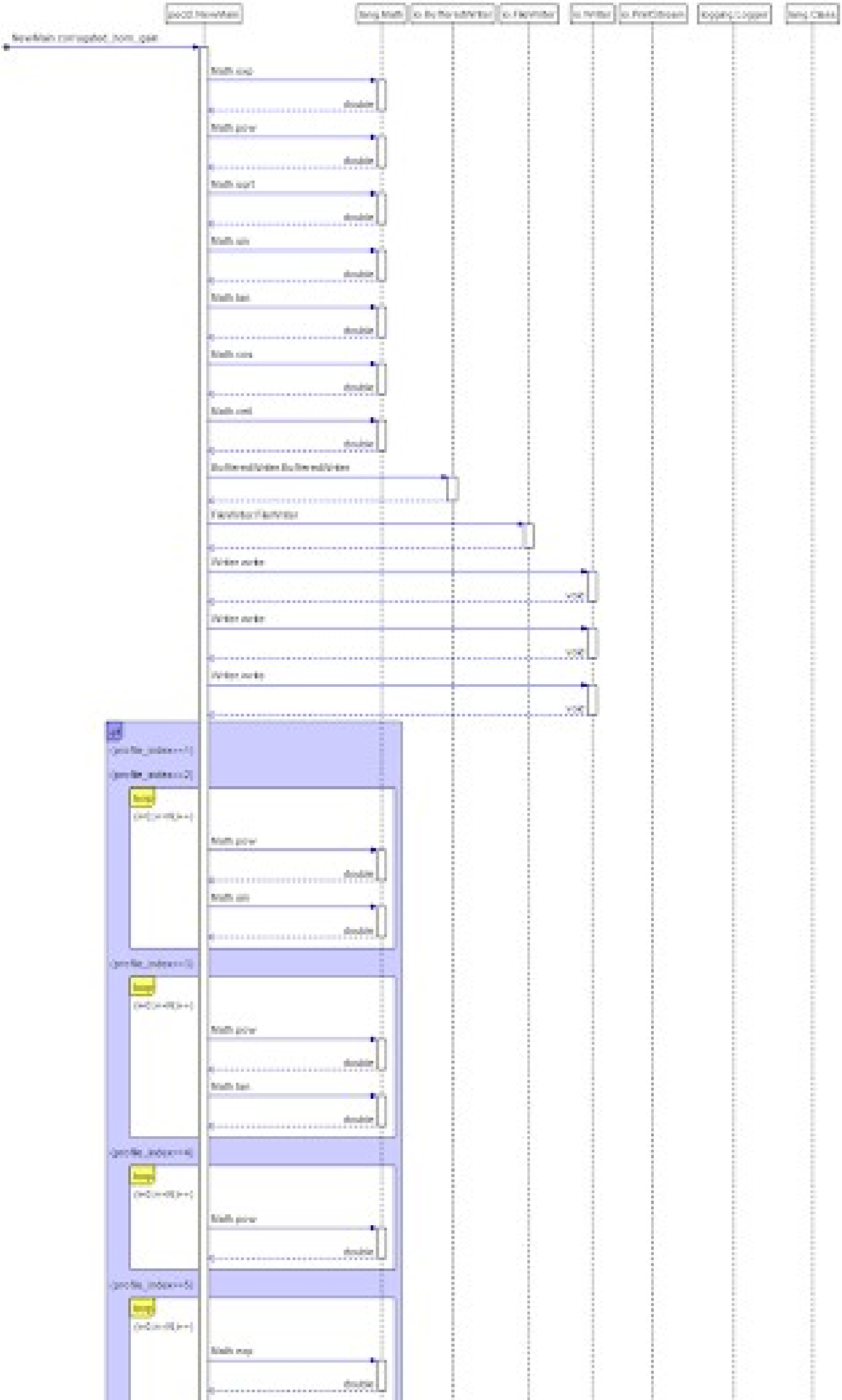
Rutinas de la clase Math como exp, pow, sqrt, sin, tan, cos y ceil

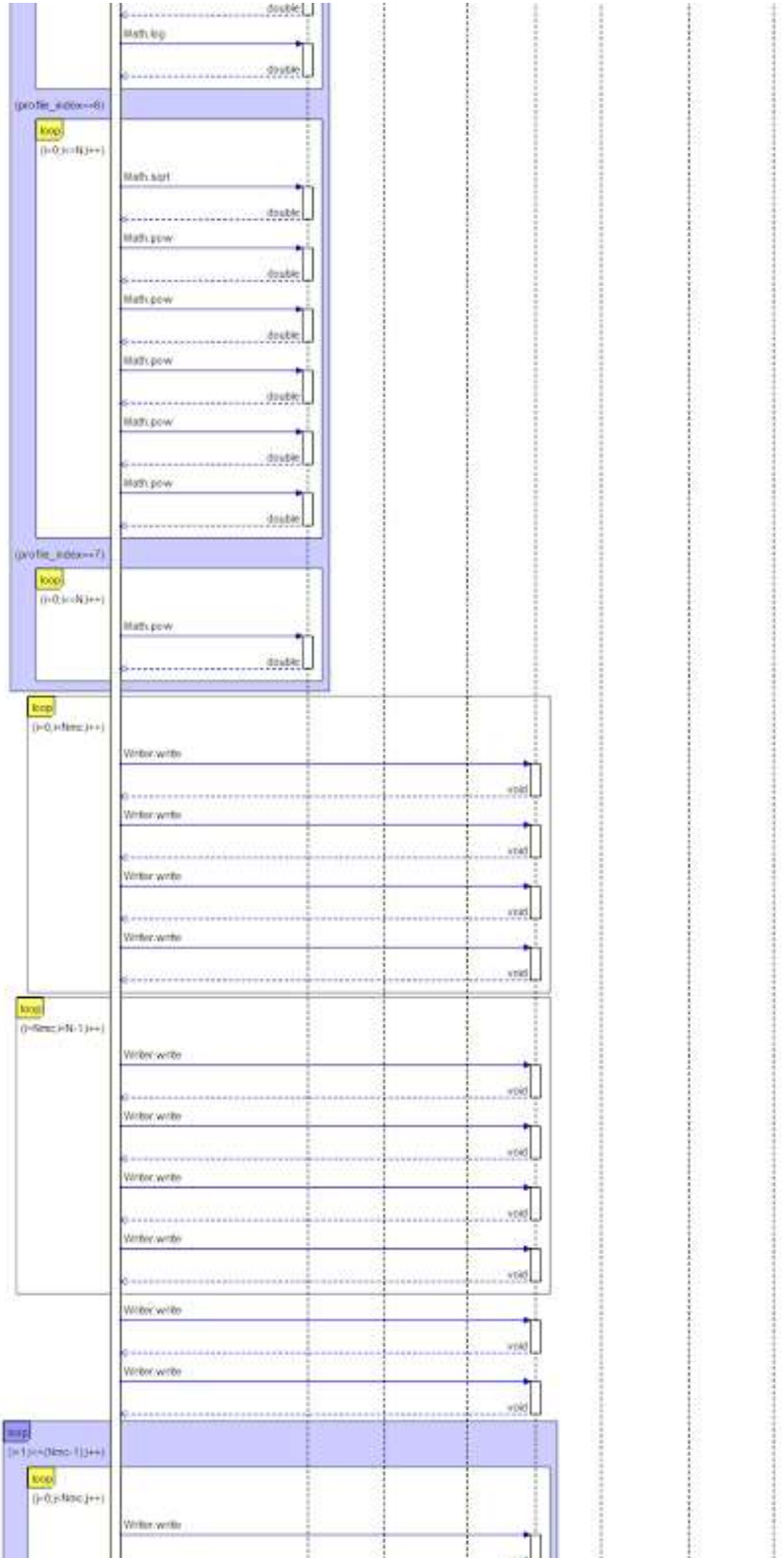
Rutinas de la clase BufferedWriter como write y close

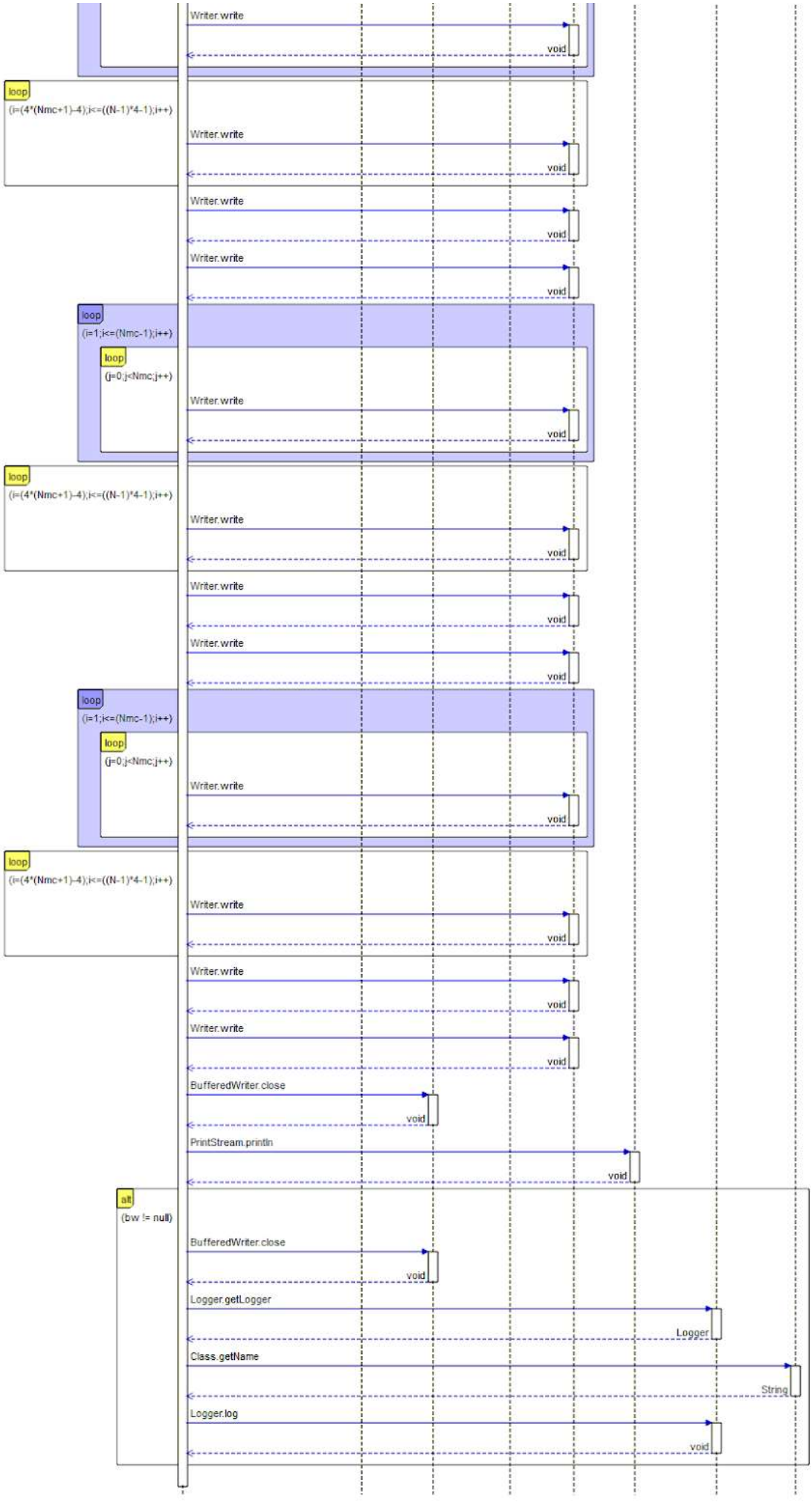
Rutinas para mostrar errores en ejecución como System.out.println y Logger.getLogger

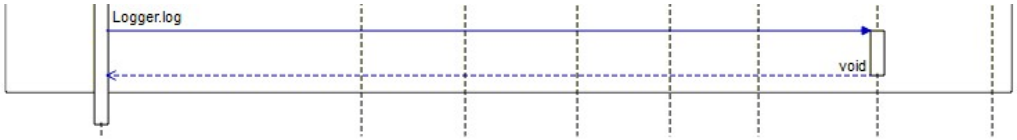
A continuación, se muestra en una imagen con todas las llamadas que realiza el procedimiento corrugated_horn_gain y el diagrama de secuencias:


Math.pow
Math.exp
Math.sqrt
Math.sin
Math.tan
Math.cos
Math.ceil
FileWriter.FileWriter
NewMain.corrugated_horn_gain
BufferedWriter.BufferedWriter
Writer.write
Math.log
BufferedWriter.close
PrintStream.println
Class.getName
Logger.getLogger
Logger.log




double
Math.log
double
(profile_index==6)
loop
(i=0;i<=N;i++)
Math.sqrt
double
Math.pow
double
Math.pow
double
Math.pow
double
Math.pow
double
Math.pow
double
(profile_index==7)
loop
(i=0;i<=N;i++)
Math.pow
double
loop
(i=0;i<=Nmc;i++)
Writer.write
void
Writer.write
void
Writer.write
void
Writer.write
void
loop
(i=Nmc;i<N-1;i++)
Writer.write
void
Writer.write
void
Writer.write
void
Writer.write
void
Writer.write
void
Writer.write
void
loop
(i=1;i<=Nmc-1;i++)
loop
(j=0;j<Nmc;j++)
Writer.write


Writer.write
void
loop
(i=(4*(Nmc+1)-4);i<=((N-1)*4-1);i++)
Writer.write
void
Writer.write
void
Writer.write
void
loop
(i=1;i<=(Nmc-1);i++)
loop
(j=0;j<Nmc;j++)
Writer.write
void
loop
(i=(4*(Nmc+1)-4);i<=((N-1)*4-1);i++)
Writer.write
void
Writer.write
void
Writer.write
void
loop
(i=1;i<=(Nmc-1);i++)
loop
(j=0;j<Nmc;j++)
Writer.write
void
loop
(i=(4*(Nmc+1)-4);i<=((N-1)*4-1);i++)
Writer.write
void
Writer.write
void
Writer.write
void
BufferedWriter.close
void
PrintStream.println
void
alt
(bw != null)
BufferedWriter.close
void
Logger.getLogger
Logger
Class.getName
String
Logger.log
void



4. Recopilamos las rutinas más importantes y se explican su funcionamiento.

. Clase Math:

Métodos usados para operaciones matemáticas como en la inicialización de variables, por ejemplo:

double fc = Math.sqrt(fmin*fmax); // Frecuencia central

Clase BufferedWriter:

Usado para escribir el script llamado script_corrugated_horn_gain.nfs y que por ejemplo el resultado que da en las últimas líneas son las siguientes:

```
919 line -n line915 -p 419.5202684807099 0.0 6066.976182580577 419.5202684807099 0.0 6072.279483439476
920 line -n line916 -p 419.5202684807099 0.0 6072.279483439476 472.5532770697009 0.0 6072.279483439476
921 line -n line917 -p 472.5532770697009 0.0 6072.279483439476 472.5532770697009 0.0 6093.492686875073
922 line -n line918 -p 472.5532770697009 0.0 6093.492686875073 420.90993958513366 0.0 6093.492686875073
923 line -n line919 -p 420.90993958513366 0.0 6093.492686875073 420.90993958513366 0.0 6098.795987733972
924 line -n line920 -p 420.90993958513366 0.0 6098.795987733972 473.9429481741247 0.0 6098.795987733972
925 line -n line921 -p 473.9429481741247 0.0 6098.795987733972 473.9429481741247 0.0 6120.009191169568
926 line -n line922 -p 473.9429481741247 0.0 6120.009191169568 422.2996106895574 0.0 6120.009191169568
927 line -n line923 -p 422.2996106895574 0.0 6120.009191169568 422.2996106895574 0.0 6125.312492028467
928 line -n line924 -p 422.2996106895574 0.0 6125.312492028467 475.33261927854846 0.0 6125.312492028467
929 line -n line925 -p 475.33261927854846 0.0 6125.312492028467 475.33261927854846 0.0 6146.525695464064
930 line -n line926 -p 475.33261927854846 0.0 6146.525695464064 423.6892817939811 0.0 6146.525695464064
931 line -n line927 -p 423.6892817939811 0.0 6146.525695464064 423.6892817939811 0.0 6151.828996322963
932 line -n line928 -p 423.6892817939811 0.0 6151.828996322963 476.7222903829721 0.0 6151.828996322963
933 line -n line929 -p 476.7222903829721 0.0 6151.828996322963 476.7222903829721 0.0 6173.042199758559
934 line -n line930 -p 476.7222903829721 0.0 6173.042199758559 425.07895289840485 0.0 6173.042199758559
935 line -n line931 -p 425.07895289840485 0.0 6173.042199758559 425.07895289840485 0.0 6178.345500617458
936 line -n guideline -p 101.28558556767442 0.0 0.0 101.28558556767442 0.0 -1050.0
937 revolve -s line100000   line100001  line100002  line100003  line100004  line200000  line200001  line200002  line200003  line200004  line300000  line300001
938 revolve -s line100000   line100001  line100002  line100003  line100004  line200000  line200001  line200002  line200003  line200004  line300000  line300001
939 delete -s line100000    line100001  line100002  line100003  line100004  line200000  line200001  line200002  line200003  line200004  line300000  line300001
940 disk -n tapa -p 0.0 0.0 -1050.0 101.28558556767442
941
```

Script que tiene distintas instrucciones como los comandos line, resolve, delete o disk y que seguramente es leído y ejecutado por otro software.

Los métodos println y getLogger no se consideran más importantes ya que no tienen un uso funcional del programa en sí, sino más bien para mostrar los errores en consola cuando se producen excepciones o fallos en la rutina.

El código dispone de las siguientes métricas en cuanto líneas de código, comentarios, etc…:

Classes:	1
Files:	1
Program Units:	2
Lines:	196
Lines Blank:	30
Lines Code:	149
Lines Comment:	46
Lines Inactive:	0
Executable Statements:	121
Declarative Statements:	38
Ratio Comment/Code:	0.31

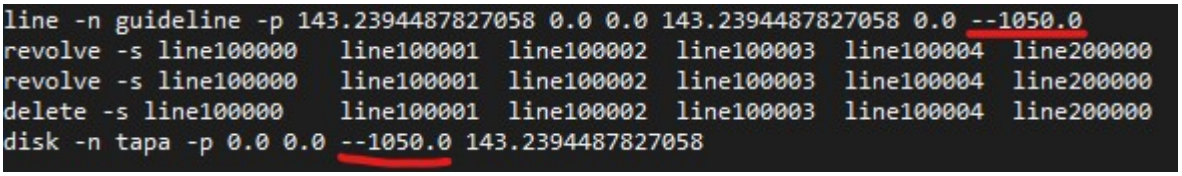
3.2. Pruebas realizadas

Se realiza las siguientes pruebas desde un método principal de la clase NewMain que incluye la rutina corrugated_horn_gain con todo su código modificado para que compile y pueda funcionar bajo Java 11.

```
public static void main(String[] args) {
    corrugated_horn_gain(0.0, 0.0, 0.0, 0.0);
    corrugated_horn_gain(-1.0, -1.0, -1.0, -1.0);
      corrugated_horn_gain(15.0, 5.0, -1.0, -1.0);
}
```

Usando el depurador de Java se llegan a las siguientes recomendaciones para posibles valores de entrada:

- fmix, fmax, alfa, D no puede ser 0.0 o se producen valores NaN o Inifnity en las variables locales
- En principio no parece haber problemas con valores negativos, pero fmin puede ocasionar problemas al software encargado de leer y ejecutar el script nfs. Cuando fmin es negativo provoca que la variable local Lg también sea negativa, y cuando escribe la última línea del comando “line” y posteriormente “disk”, existe un guion de más por el número negativo como se muestra a continuación:



Siempre que el software encargado de interpretar este script tenga en cuenta esto, no debería de haber mayor problema.

- Tal como se explica en el código de fuente y, realizando la prueba oportuna, el argumento de entrada fmin tiene que ser menor que fmax.
- alfa y D en principio podría tener cualquier valor (excepto 0.0) sin que repercuta demasiado en el resultado del fichero que se escribe.