UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Análisis informatizado de armonía barroca

Samuel Fuentes Fernández
Tutor: Jose Ramón Dorronsoro Ibero

Junio 2015

Contenido

Índice de tablas

Índice de figuras

1 Introducción

1.1 Descripción general

El presente trabajo trata el desarrollo de una aplicación de ayuda docente para la enseñanza de la armonía, una disciplina fundamental en los estudios musicales.

Esta aplicación deberá facilitar la labor de los docentes, como herramienta de ayuda a la corrección; por tanto deberá ser capaz de analizar un documento musical (en formato digital) e indicar los incumplimientos de las normas de la armonía barroca que ocurran en el mismo.

El uso de la armonía barroca en lugar de la de otros estilos se debe a que es la utilizada como referencia en la enseñanza de la armonía por tener unas reglas muy definidas que permiten a los alumnos entender los conceptos con más sencillez.

Pese a tener unas reglas definidas, hay situaciones aisladas que pueden funcionar como excepciones a las normas de la armonía barroca, pero sólo en el caso de que no incurran auditivamente en las sonoridades que las reglas pretenden evitar. Por tanto, para dejar en manos del docente la decisión de si un incumplimiento de norma se trata de un error o no, la aplicación se limitará a indicar la localización de los errores.

También se incluye en el trabajo un generador de casos de error, es decir, incumplimientos de las normas de la armonía barroca. Esta funcionalidad puede ser utilizada por los docentes para ejemplificar situaciones en las que las distintas normas se incumplen.

1.2 Palabras clave

Reglas de armonía barroca; detección de errores compositivos; aplicaciones informáticas para la música; MusicXML.

1.3 Keywords

Baroque harmony rules; composing errors detection; computer applications in music; MusicXML.

1.4 Armonía barroca

La elección del Barroco como estilo de referencia en las enseñanzas de armonía viene dada por dos razones principales:

- Es un periodo en el que se han abandonado definitivamente los modos eclesiásticos en beneficio de la tonalidad, por lo que la sonoridad de las composiciones de esta época es similar a la del grueso del repertorio clásico.
- Pese a la gran complejidad del estilo barroco, en cuanto al uso de funciones armónicas es un periodo de sencillez. Las funciones armónicas llevaban poco tiempo definidas y por tanto se busca asentarlas más que experimentar con ellas.

A continuación se sitúa el contexto de este estilo, la elaboración de las reglas y su aplicación actual en las enseñanzas de armonía.

1.4.1 Contexto

Se utiliza el término Barroco para referirse al estilo predominante en la música europea del siglo XVII. Ideológicamente es una etapa muy marcada por la religión: la lucha entre católicos y protestantes. Esto lleva a un uso del arte como herramienta propagandística, tanto en política como en religión, ya que también es la época en la que la monarquía absoluta trata de imponer el poder de los reyes frente a la nobleza.

En la historia de la música se trata de una época de gran esplendor: el avance en cuanto a desarrollo de instrumentos que se había hecho durante el Renacimiento lleva a los compositores a disponer de una gran diversidad de instrumentos para los que componer: clave, oboe, traverso, violín, viola, violoncelo…

En cuanto a la música vocal cabe destacar (además del nacimiento de la ópera) que aparece una gran demanda de composiciones en el protestantismo, ya que a diferencia del catolicismo que limitaba la intervención musical en la liturgia a los monjes, buscaba que todos los fieles participaran, contribuyendo la música a la unidad de los feligreses. De este modo surgen diversos maestros de capilla a los que se exige una gran producción, pero pese a ser obras sencillas requieren ciertos conocimientos, lo que también lleva a mejorar la formación musical de la gente por el simple hecho de necesitarla para la religión.

Como consecuencia de esta participación popular en la música religiosa protestante surgen los corales, piezas a cuatro voces sin demasiadas complicaciones que los fieles cantan durante la misa.

Con el tiempo estos corales se convertirán en las obras de referencia para la armonía de este estilo por mostrar con gran claridad el uso de cada acorde (al disponer de cuatro voces que sólo pueden hacer un sonido cada una se limita la variedad de notas a las del acorde que se esté usando). Como consecuencia de esto, se utiliza el mismo formato de los corales (composiciones vocales a cuatro voces) en los ejercicios de las enseñanzas de armonía.

1.4.2 Reglas

A partir del estudio de la música barroca se encuentran una serie de criterios cumplidos por la mayoría de los compositores y que contribuyen a que esta música tenga su sonoridad concreta. Los compositores de la época no seguían conscientemente las reglas, simplemente escribían música como se hacía en su tiempo. Las reglas surgen a partir del posterior estudio por parte de los musicólogos de la música de este estilo, y no la música a partir de la aplicación de las reglas. Aun así estas reglas ayudan a comprender la música de este estilo y por extensión el uso de la polifonía como corriente dominante en la música occidental.

Lo que buscan los compositores barrocos habitualmente es huir de las sonoridades medievales, tanto populares (la música clásica sigue la corriente de la música de la Corte, por ser la que quedaba escrita debido a una mayor disponibilidad de medios) como gregorianas. Estas sonoridades están

marcadas por una insistencia en notas e intervalos concretos que ayudan a definir y remarcar los sonidos importantes sobre los que se apoyan las obras.

Por eso en la música barroca se evita llegar a estos sonidos por movimientos que resulten evidentes al oído. De este modo surgen las siguientes reglas de prohibición de:

- Quintas y octavas directas.
- Quintas y octavas paralelas.
- Duplicación de la sensible.

Otro de los motivos que hacen que la música barroca (en concreto la música vocal) tenga su sonoridad característica es la restricción de algunos elementos por falta de recursos técnicos con los que ejecutarlos. Esto se debe a que la profesionalización de los cantantes es muy próxima en el tiempo y aún no han tenido tiempo de desarrollar su técnica para cubrir estas situaciones. De estas restricciones surgen las siguientes reglas de prohibición de:

- Segundas aumentadas.
- Cruzamiento de voces.
- Posiciones abiertas.
- Falsa relación cromática.

Por último también hay restricciones en el uso de determinadas sonoridades que vienen dadas por razones culturales. En este contexto aparece el tritono (un intervalo formado por una distancia de tres tonos entre dos sonidos), cuya sonoridad oscura lo llevó a ser prohibido por la Iglesia por considerarlo maligno. Tanto es así que incluso llegó a denominarse *diabolus in música*, el diablo en la música. Por tanto por razones culturales surge la prohibición del uso de tritonos.

1.4.3 Enseñanzas de armonía

Las enseñanzas actuales de armonía pretenden que los alumnos adquieran los conocimientos básicos sobre lo que ocurre en las obras "por debajo" de la melodía, esto es detectar las funciones armónicas que aparecen en el transcurso de la obra.

Una función armónica es el papel que desempeña un acorde en una tonalidad. Si define la tonalidad será tónica; si crea tensión será dominante; si no cumple ninguna de estas funciones será subdominante.

El estudio de la armonía dota a los alumnos de los conocimientos necesarios tanto para el análisis musical como para el estudio de los fundamentos de la composición.

Para poder establecer la relación entre las funciones armónicas y su sonoridad, los alumnos elaboran a modo de ejercicios pequeñas composiciones a cuatro voces que respeten las normas de la armonía barroca.

1.5 Objetivo del Trabajo Fin de Grado

El objetivo de este trabajo es por tanto el desarrollo de una herramienta informática de ayuda a los docentes de armonía. Esta aplicación deberá indicar los incumplimientos en las normas de la armonía barroca sobre un documento musical digitalizado, de manera que el docente ahorre el tiempo necesario para realizar esta comprobación de manera manual.

El contenido de este trabajo está dividido en los siguientes capítulos:

- Tecnología informática en la música: una exposición sobre el estado actual de las aplicaciones musicales, así como ejemplos de las aplicaciones relacionadas y complementarias a la del proyecto que se pueden encontrar en el mercado.
- Armonía barroca: en este capítulo se hace una introducción al estilo barroco (sobre el que se trabaja musicalmente en la aplicación) así como una descripción de las reglas de la armonía de este estilo.
- Análisis, diseño y desarrollo del sistema propuesto: descripción detallada del desarrollo software para la aplicación del proyecto.
- Ejemplos de uso: este capítulo recoge los resultados devueltos por la aplicación para una serie de ejemplos.
- Resumen, conclusiones y posibles ampliaciones: aquí se recogen el resumen y conclusiones del trabajo, así como una relación de posibles futuras mejoras o ampliaciones.

Pese a no ser el objetivo principal de trabajo, también se expone en detalle el diseño (como posible ampliación) de un sistema de generación de voces adicionales con interés melódico a partir de una voz proporcionada.

2 Tecnología informática en la música

En este capítulo se presenta la tecnología en la que se basa el proyecto, así como las aplicaciones relacionadas con la desarrollada en este trabajo: aplicaciones similares y complementarias.

2.1 Tecnología base

El sistema trabajará con archivos en formato XML, concretamente en el estándar MusicXML (formato basado en XML orientado a la notación musical). XML es un lenguaje de marcado desarrollado para contener información de manera legible por los ordenadores y así conseguir un sistema de intercambio de información sencillo y multiplataforma.

El formato se basa en el marcado de la información mediante etiquetas (*tags*) que se pueden anidar. Cada etiqueta puede contener en el siguiente nivel varias etiquetas distintas o del mismo tipo. Añadiendo a esto que la adición de etiquetas depende por completo del programador (el lenguaje es totalmente extensible) hacen de este lenguaje una herramienta muy potente y que se puede adaptar con facilidad a distintos requisitos, como es el caso de la representación de música gracias al estándar MusicXML.

A continuación se presentará con detalle el estándar de notación musical MusicXML.

2.2 Formato MusicXML

Ante la proliferación de aplicaciones digitales del campo de la música, en 2004 se lanza el formato MusicXML (http://www.musicxml.com/) para permitir el intercambio de archivos entre todas estas aplicaciones. El formato es de licencia pública y puede ser usado gratuitamente.

El formato (de extensión .xml o .mxl) poco a poco fue ganando popularidad y en agosto de 2014 ya era soportado por más de 180 aplicaciones, entre ellas los principales editores de música y secuenciadores.

La estructura del estándar consta de varios bloques de etiquetas, separando distintos elementos del documento musical, desde información general como el título o el autor hasta las propias notas. No todas las etiquetas son utilizadas por todas las aplicaciones; de hecho la gran mayoría son prescindibles. Atendiendo a la separación en los documentos de estos distintos bloques de información encontramos:

- El primer bloque de un archivo MusicXML, *identification*, es opcional y contiene la información del documento (salvo el título que es una etiqueta al mismo nivel, *movement-title*): compositor, arreglista, letrista, copyright y la información relacionada con el software utilizado para generar el documento, *encoding*.

- Posteriormente se hallan los valores que determinarán la representación gráfica del documento: márgenes, espaciado, distintos valores del layout, … Este bloque es *defaults*. Si se quieren añadir al documento campos adicionales (*defaults* solo incluye la notación

musical), deben ser añadidos de uno en uno con la etiqueta *credit,* indicando información como la página en la que se encuentran, la posición y el texto.

Después de estos bloques de información adicional se encuentran los referentes a la notación musical:

- *part-list,* que contiene la información sobre el número de voces, qué tipo de sonido utilizar para secuenciarlas (con identificadores para formato midi compartidos por algunos programas que utilizan este estándar), volumen, nombre de las voces, …

- Una vez detallada toda esta información general, aparece un grupo *part* para cada voz con la clave y todas las notas. Puede contener información sobre la distribución en el espacio de los compases y las figuras.

2.2.1 Ejemplo de MusicXML

El ejemplo siguiente muestra un documento musical muy básico, que simplemente consta de una única nota. En esta subsección se detallará dicho ejemplo, tanto desde una perspectiva musical como desde una perspectiva informática, tomando como referencia el archivo MusicXML con el que se ha generado el documento.

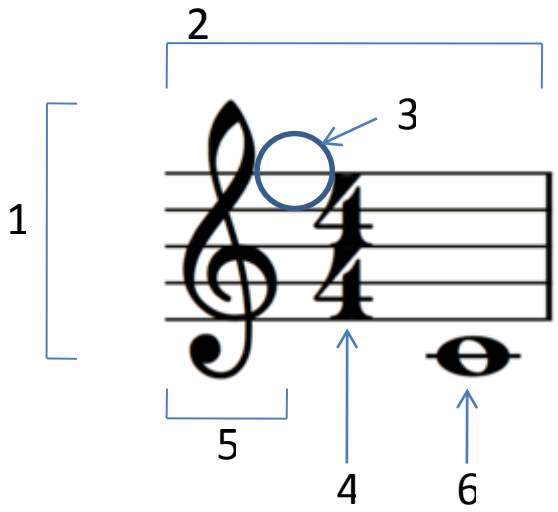


Figura 2.2-1 Elementos de una partitura

Una partitura consta de distintas voces (1) que están formadas por compases (2) y que a su vez contienen notas (6), que proporcionan la información necesaria sobre la altura y duración de cada sonido. La información general sobre cada voz se establece al principio de la misma utilizando una clave (5) (que determinará la altura de referencia de las notas, por ejemplo la clave de "fa en cuarta" representa que el sonido de la cuarta línea es un "fa"), armadura (3) (indica qué alteraciones se utilizarán durante toda la obra salvo que se especifique lo contrario; en este ejemplo la armadura no presenta ninguna alteración) y compás (4) (representado con dos números, el inferior indica la figura de referencia utilizando potencias de base 2, en este caso negra; el superior indica la cantidad de estas figuras que compondrán un compás).

A continuación se presenta un ejemplo de archivo MusicXML indicando las equivalencias entre código y notación musical:

```
<?xmlversion="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC
    "-//Recordare//DTD MusicXML 3.0 Partwise//EN"
    "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise version="3.0">
<part-list>
  <score-part id="P1">
          <part-name>Music</part-name>
  </score-part>
</part-list>
1{<part id="P1">
  2{<measure number="1">
          <attributes>
                  <divisions>1</divisions>
                  3{<key>
                          <fifths>0</fifths>
                  </key>}
                  4{<time>
                          <beats>4</beats>
                          <beat-type>4</beat-type>
                  </time>}
                  5{<clef>
                          <sign>G</sign>
                          <line>2</line>
                  </clef>}
          </attributes>
          6{<note>
                  <pitch>
                          <step>C</step>
                          <octave>4</octave>
                  </pitch>
                  <duration>4</duration>
                  <type>whole</type>
          </note>}
  </measure>}
</part>}
</score-partwise>
```

2.3 Aplicaciones relacionadas con la desarrollada

Actualmente no existen aplicaciones con soporte MusicXML para la comprobación de las reglas de la armonía barroca. Sin embargo, teniendo en cuenta la posible ampliación del trabajo presentado a la generación de voces adicionales para una melodía dada (ver subsección 6.4.3), sí se pueden encontrar aplicaciones afines a nuestra propuesta de ampliación en el sentido de que generan voces compatibles a una dada de acuerdo a diferentes criterios.

La aplicación más cercana a este concepto de generación de voces secundarias es HarmonyWiz (http://www.wizdommusic.com/products/harmonywiz.html), una aplicación para iOS de edición musical. Entre las similitudes con la ampliación propuesta para la ampliación de este trabajo encontramos que:

- Se generan tres voces adicionales para una melodía dada, identificando los grados por los que va pasando la melodía en cada unidad de tiempo.

Por otro lado entre las posibles limitaciones se puede destacar que:

- No se tienen en cuenta las reglas de la armonía barroca; por ejemplo, no se tiene en cuenta la presencia de quintas u octavas directas y paralelas. Al utilizar sonidos de los acordes y no notas extrañas, la sonoridad de las melodías armonizadas con esta aplicación es agradable, pero desde el punto de vista formal de la armonía barroca, no son correctas.
- No se intenta maximizar la duración de cada grado de la escala en la armonía: simplemente si la duración del sonido de la voz superior es mayor que la unidad de tiempo, se busca utilizar un número de unidades de tiempo que consigan cubrir esta duración.

En cualquier caso, las voces generadas son un simple relleno armónico para la melodía dada, no buscando en ningún momento el contrapunto. Por el contrario, con la ampliación de la aplicación sugerida, se podrían generar nuevas voces que generen interés, haciendo uso de un contrapunto sencillo pero apreciable.

2.4 Aplicaciones externas complementarias

Al utilizar el estándar MusicXML la aplicación del proyecto es compatible con otras aplicaciones que trabajen con este formato. Esta compatibilidad se basa en el intercambio de información mediante documentos MusicXML.

Entre estas aplicaciones complementarias podemos distinguir dos tipos: los editores de música (permiten leer, representar e imprimir las partituras desde MusicXML) y los sintetizadores (permiten reproducir la música contenida en los archivos MusicXML). En algunas ocasiones estos dos tipos de aplicaciones están combinados.

Entre estas aplicaciones externas podemos encontrar:

- **Cubase**: es un software de edición de audio y secuenciación de música. Su primera versión fue lanzada en 1989 y se trata de software propietario. Entre sus características presenta un editor de partituras perteneciendo por tanto a ambas categorías de aplicaciones. Sin

embargo, su aspecto más relevante es la generación digital de audio, consiguiendo generar archivos de gran calidad y realismo en formatos MIDI, WAV o MP3.

- **Finale**: es un software propietario de edición de partituras. La primera versión fue lanzada en 1988. Permite generar en formato PDF partituras musicales. Además permite generar los archivos MIDI y WAV a partir del mismo documento MusicXML. Explotando esta funcionalidad mediante librerías externas de sonidos, se ha conseguido mejorar la funcionalidad básica del programa (cuyos sonidos originales son bastante pobres) hasta una calidad cercana a la de algunos secuenciadores.

- **MuseScore**: es un software libre (bajo licencia pública general GNU, GNU GPL) de edición de partituras. Proporciona soporte para Windows, OS X y Linux (a diferencia de la mayoría de editores que no dan soporte a este sistema). La versión 1.0 fue lanzada en 2011. Además de ser completamente compatible con MusicXML, MIDI o PDF (entre otros formatos), da soporte a los formatos nativos de otros editores, como Sibelius o Finale. El audio generado por esta aplicación puede ser exportado en formatos como WAV, OGG o MP3. Al igual que Finale dispone de librerías de instrumentos externas, aunque su calidad es inferior.

Estas aplicaciones se utilizan para dos funcionalidades complementarias a las de la aplicación del trabajo:

- Escucha de los documentos generados (casos de prueba).
- Visionado de los documentos en formato partitura, incluyendo las anotaciones hechas por la aplicación.

La integración de ambas funcionalidades en la aplicación propuesta aquí están consideradas como posibles ampliaciones de la misma (ver subsección 0).

3 Armonía barroca

3.1 Introducción a la armonía barroca

3.1.1 El origen de la polifonía

En música se denomina armonía a la unión y combinación de sonidos simultáneos y diferentes, o lo que es lo mismo, al arte de formar y enlazar los acordes. Por su propia razón de ser, la armonía está ligada a otro concepto musical: la polifonía.

La polifonía es una textura musical en la que varias voces son independientes o imitativas entre sí, pero todas tienen una cierta importancia musical. Surge en la Edad Media como un tropo (una modificación añadida al canto gregoriano, que podía ir desde la modificación de la música o los textos hasta, como en este caso, la adición de voces completamente nuevas) inventado por los propios monjes. En origen son voces paralelas al canto gregoriano (*cantus firmus*) a una distancia de 4ª, 5ª u 8ª (una diferencia a partir de la voz original de 4, 5 u 8 notas), el denominado *organum paralelo*. Al explorar distintas distancias interválicas (intentan integrar en su música el resto de intervalos, no únicamente 4ª, 5ª y 8ª como utilizaban hasta entonces) para buscar unísonos (mismo sonido en las voces) al comienzo y final de las obras, se van añadiendo más intervalos, como las 3ª y las 6ª.

Cuando la nueva voz va cobrando una cierta independencia, el canto (pues hablamos de música religiosa ya que la Iglesia prohibió en distintas bulas la presencia de instrumentos en el templo) empieza a ser más complejo, contribuyendo a una profesionalización del mismo (los monjes solistas cantan las voces adornadas y el coro la voz gregoriana original) y fomentando la escritura, ya que al ser el canto más complejo es más complicado que sea simplemente memorizado.

Gracias a esta necesidad de escritura se va conservando esta música (la evolución desde el primer tratado, el *Musica Enchiriadis,* del siglo IX hasta el Renacimiento) en importantes fuentes para la historia de la música como el *Tropario de Winchester*, *Códice Calixtino*, *Magnus Liber Organi, Antifonario de Worcester*...

La polifonía consigue un desarrollo bastante notable, alcanzando una gran complejidad a finales de la Edad Media, pero debido a su origen sigue estando basada en los modos del canto gregoriano (los modos son un sistema de organización de los sonidos distinto al sistema utilizado hoy en día, el sistema tonal. En la Edad Media cada escala según esta organización tenía una nota *finalis* y una nota *tenor* que marcaban su ámbito y su carácter).

3.1.2 El sistema tonal

Durante el Renacimiento hay una gran cantidad de innovaciones en la música. Uno de estos avances es el desarrollo de nuevos sistemas de afinación, que frente al temperamento (sistema de afinación) utilizado en la Edad Media (temperamento pitagórico) no tiene terceras disonantes. Es decir, que los intervalos de 3ª dejan de ser disonantes y comienzan a utilizarse.

A esto se suma que la Iglesia desde el Concilio de Trento busca evitar la polifonía exagerada que se había ido volviendo más compleja desde finales de la Edad Media, por lo que se busca recoger la sonoridad polifónica en elementos que no vuelvan la música confusa.

Como consecuencia de estos dos elementos surgen los acordes (ya que se mantiene una sonoridad polifónica pero clara, pero no habían surgido antes por utilizar intervalos de 3ª), que serán la base del sistema tonal.

3.1.3 La polifonía en el Barroco

Frente a la búsqueda de una mayor sencillez que llevó a cabo la Iglesia Católica desde el Concilio de Trento, entre los protestantes aparecieron distintas corrientes: desde aquellos que evitaban la música por interferir con la doctrina (como Calvino) hasta los que veían la música como elemento fundamental de ayuda para la liturgia, como es el caso de Lutero.

En la Iglesia Luterana se hacen necesarios cánticos para los fieles, para que participen en la misa. De este modo, el mismo Lutero empieza a crear composiciones a cuatro voces, sencillas (los fieles no son cantantes profesionales) y en las que los acordes tienen gran importancia (presentan una textura homofónica, esto es que las distintas voces cambian de nota habitualmente a la vez). Estas composiciones son los corales.

Esta tradición de corales protestantes es cultivada por distintos autores desde finales del Renacimiento, y supone una parte muy importante del repertorio del Barroco.

3.2 Descripción musical detallada de los errores

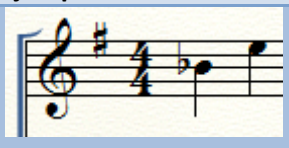
Se consideran errores en la armonía barroca aquellas combinaciones de sonidos cuya sonoridad no encaja con el estilo de la época (ya que las reglas de la armonía se establecen mediante el estudio musicológico una vez el estilo ha terminado). Esto no quiere decir que sean disonantes, de hecho en la mayoría de casos no lo son, pero o bien recuerdan a la sonoridad de los modos gregorianos (que se quiere evitar) o son mal vistos por razones culturales o por dificultades técnicas a la hora de su ejecución.

Entre los errores recogidos en las reglas de la Armonía Barroca encontramos los que afectan a una única voz, los que afectan a varias voces en una única unidad de tiempo y los que afectan a varias voces en dos unidades de tiempo consecutivas:

3.2.1 Reglas para una única voz

Tritonos

Un tritono es un intervalo en el que las dos notas que lo forman distan de tres tonos. Evitar las relaciones de tritonos es lo común en cualquier armonía clásica, ya que se asociaba esta relación con el mal, siendo prohibida por la Iglesia.

Ejemplo	Descripción
	En la primera negra encontramos un "si ♭". En la segunda negra encontramos un "mi". La distancia entre estas dos notas es de una 4ª Aumentada, por lo que se produce un tritono entre ambos sonidos.

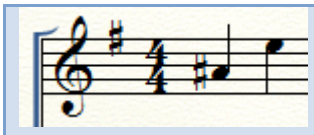
	En la primera negra encontramos un "la #". En la segunda negra encontramos un "mi". La distancia entre estas dos notas es de una 5ª Disminuida, por lo que se produce un tritono entre ambos sonidos.

Tabla 3.2-1 Ejemplos de tritonos

Segundas aumentadas

Una segunda aumentada es un intervalo en el que dos notas a distancia de segunda (correlativas) están separadas por un tono y un semitono. La prohibición en el uso de segundas aumentadas se debe simplemente al hecho de tratarse de música vocal, ya que es una relación tremendamente complicada de cantar. En música instrumental esta regla puede no aplicarse. En caso de querer escribir el mismo intervalo deberá usarse como una 3ª menor.

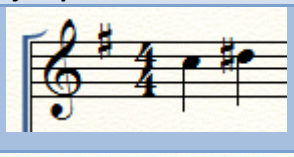
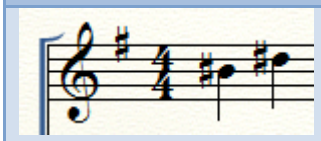
Ejemplo	Descripción
	En la primera negra encontramos un "do". En la segunda negra encontramos un "re #". La distancia entre estas dos notas es de un tono y un semitono, una segunda aumentada.
	En la primera negra encontramos un "si #". En la segunda negra encontramos un "re #". La distancia entre estas dos notas es de un tono y un semitono, sin embargo a diferencia del ejemplo anterior el intervalo es una 3ª menor, por lo que no hay ningún incumplimiento de reglas.

Tabla 3.2-2 Ejemplos de segundas aumentadas

3.2.2 Reglas para varias voces en una unidad de tiempo

Posición abierta

Las posiciones abiertas son aquellas en las que hay una distancia mayor que una octava entre dos voces. Se aplica a las voces contiguas excepto entre soprano y tenor. La explicación de este fallo es que al tratarse de música vocal si se excede esta distancia de octava los cantantes pueden perder las referencias a la hora de interpretar su propia voz.

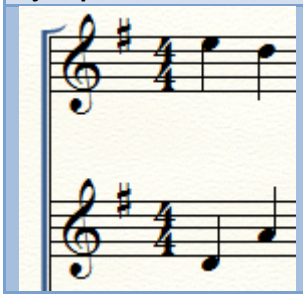
Ejemplo	Descripción
	En la primera nota de estas dos voces contiguas encontramos un "re" y un "mi". La distancia entre ellas es de 9ª, por tanto se trata de una posición abierta.

Tabla 3.2-3 Ejemplo de posición abierta

Cruzamiento de voces

El cruzamiento de voces consiste en que una voz haga sonidos más graves que la inmediatamente inferior o más agudos que la inmediatamente superior. Se aplica para todas las voces con sus

contiguas. La explicación de este fallo es que puede provocar confusión y pérdida de referencias en los cantantes.

La regla se extiende al sonido anterior de las voces contiguas como límite de altura.

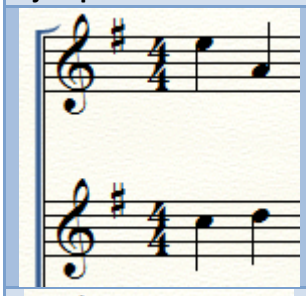
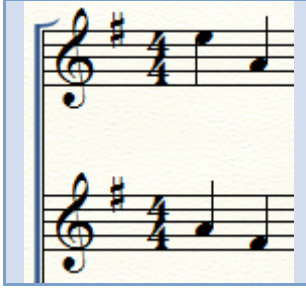
Ejemplo	Descripción
	En la primera negra encontramos un "do" y un "mi", estando la voz superior una 3ª Mayor por encima de la inferior. En la segunda negra encontramos un "re" y un "la", estando la voz superior una 4ª Justa por debajo de la inferior. Por tanto, se produce un cruzamiento de voces en la segunda negra.
	En la primera negra encontramos un "do" y un "mi", estando la voz superior una 3ª Mayor por encima de la inferior. En la segunda negra encontramos un "fa" y un "la", estando la voz superior una 3ª Mayor por encima de la inferior. Sin embargo, el sonido de la segunda negra para la voz superior ("la") es más grave que el sonido de la primera negra para la voz inferior ("do"), por lo que hay un cruzamiento de voces en la segunda negra.

Tabla 3.2-4 Ejemplos de cruzamientos de voces

Duplicación de la sensible

Sensible es la denominación para el séptimo grado de una escala cuando dista un semitono del primer grado, en caso de distar un tono se denomina subtónica. Por ejemplo, en la tonalidad de do menor, la nota si tiene la función de sensible mientras que si ♭ actúa como subtónica.

En la armonía barroca la duplicación de la sensible es una prohibición muy estricta, ya que es un sonido de mucha tensión y por sus requisitos especiales de resolución y preparación, provocaría octavas paralelas.

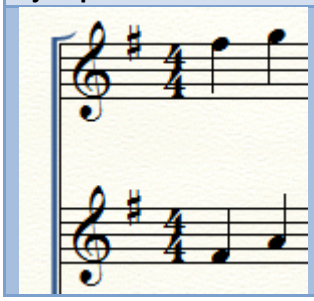
Ejemplo	Descripción
	Este ejemplo se encuentra en la tonalidad de Sol Mayor, en la que la sensible es el "fa #" (al llevar el "fa" sostenido en la armadura, todos los "fa" que encontremos en esta partitura serán sostenidos salvo que se indique lo contrario). En la primera negra encontramos un "fa #" para ambas voces, por lo que se produce una duplicación de la sensible.

Tabla 3.2-5 Ejemplo de duplicación de la sensible

3.2.3 Reglas para varias voces en varias unidades de tiempo

Octavas y quintas paralelas

Los intervalos paralelos consisten en la repetición en dos unidades de tiempo consecutivas del mismo intervalo (de manera directa o con su intervalo simplificado). En la armonía barroca la

prohibición de su uso se limita a las octavas y a las quintas (las excepciones son: en la versión francesa de la armonía barroca que la segunda quinta sea disminuida, en la versión alemana que cualquier quinta sea disminuida si al menos una voz no es extrema). Esta regla trata de evitar que la sonoridad recuerde a la de la música medieval (modalidad en lugar de tonalidad), ya que es de lo que tratan de huir los compositores barrocos.

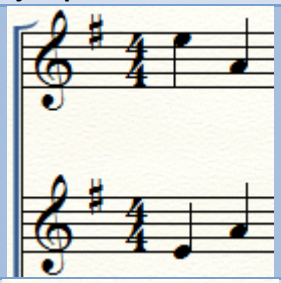
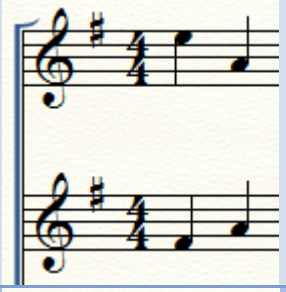
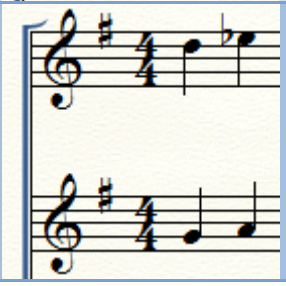
Ejemplo	Descripción
	En la primera negra encontramos un "mi" en ambas voces, habiendo una distancia de 8ª entre ellas. En la segunda negra encontramos un "la" en ambas voces, siendo unísono (una simplificación de la 8ª). Por tanto, entre estas dos voces se produce una octava paralela.
	En la primera negra encontramos un "fa" y un "mi", habiendo una distancia de 7ª entre ellas. En la segunda negra encontramos un "la" en ambas voces, siendo unísono (una simplificación de la 8ª). Por tanto, entre estas dos voces no hay ni octavas ni quintas paralelas (queda corregido el error del ejemplo anterior).
	En la primera negra encontramos un "sol" y un "re", habiendo una distancia de 5ª Justa entre ellas. En la segunda negra encontramos un "la" y un "mi ♭", habiendo una distancia de 5ª Disminuida. Por tanto (asumiendo que al menos una voz no es extrema) no hay ni octavas ni quintas directas entre estas dos voces.

Tabla 3.2-6 Ejemplos de octavas y quintas paralelas

Octavas y quintas directas

Las restricciones sobre intervalos directos se limitan a las octavas y a las quintas (las quintas sólo cuando son entre voces extremas). Consisten en llegar a los dos sonidos que formen el intervalo con la misma dirección: subiendo o bajando ambos. Si una de las voces hace el mismo sonido que en la unidad de tiempo anterior o una voz llega bajando y otra subiendo, este error se evita. La existencia de esta regla se debe (al igual que las quintas y octavas paralelas) a un afán de evitar las sonoridades medievales.

Ejemplo	Descripción

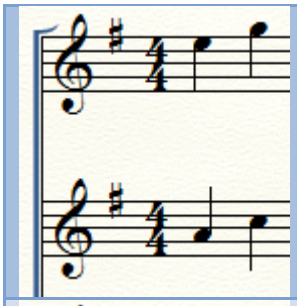
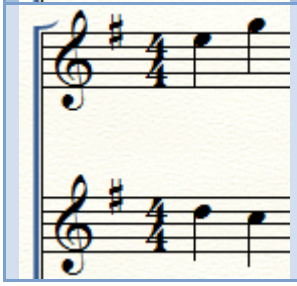
	En la voz superior encontramos un intervalo ascendente (3ª menor). En la voz inferior encontramos un intervalo ascendente (3ª menor). En la segunda negra encontramos un "do" y un "sol", habiendo una distancia de 5ª entre ellas. Por tanto, al tratarse de una 5ª alcanzada en ambas voces por movimiento ascendente, se trata de una quinta directa.
	En la voz superior encontramos un intervalo ascendente (3ª menor). En la voz inferior encontramos un intervalo descendente (2ª Mayor). En la segunda negra encontramos un "do" y un "sol", habiendo una distancia de 5ª entre ellas. Por tanto, al tratarse de una 5ª alcanzada en ambas voces por movimiento contrario, no se trata de una quinta directa.

Tabla 3.2-7 Ejemplos de octavas y quintas directas

Falsa relación cromática

La falsa relación cromática se produce cuando una alteración no se resuelve en la misma voz si no en otra. Este fallo puede conllevar confusión (por no cantar el sonido habitual tras una alteración) en los cantantes.

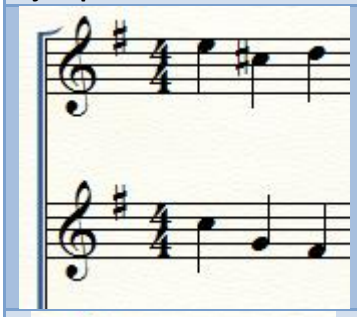
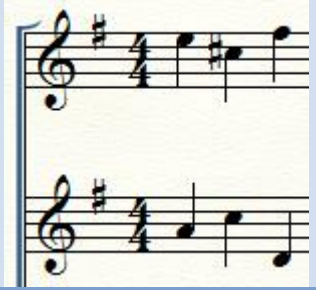
Ejemplo	Descripción
	En la segunda negra de la voz superior encontramos un "do #", que se resuelve correctamente a un "re" en la tercera negra. Por tanto se trata de una resolución correcta y no una falsa relación cromática.
	En la segunda negra de la voz superior encontramos un "do #". Las posibles resoluciones serían "do" y "re", pero ninguna aparece en la tercera negra para esta voz. Sin embargo, en la tercera negra para la voz inferior si aparece un "re". Por tanto se produce una falsa relación cromática entre el "do #" de la segunda negra en la voz superior y el "re" de la tercera negra en la voz inferior.

Tabla 3.2-8 Ejemplos de falsas relaciones cromáticas

4 Análisis, diseño y desarrollo del sistema propuesto

4.1 Análisis de requisitos

El objetivo de la aplicación es ser una herramienta de ayuda en la actividad docente para los profesores de armonía. La forma habitual de enseñar armonía es la explicación de las distintas reglas y técnicas y la corrección de los ejercicios de los alumnos en los que aplican los nuevos conocimientos adquiridos. La aplicación de las reglas puede ser flexible si auditivamente no se aprecian los errores que dichas reglas pretenden evitar. Por eso, la aplicación simplemente deberá indicar los incumplimientos de las reglas, dejando a la elección del docente decidir si se trata de errores y deben ser corregidos o por el contrario son usos no habituales que el alumno ha empleado (intencionadamente o por accidente).

Como tal, la aplicación tiene interés tanto para profesores de armonía, como herramienta automática de ayuda en la corrección de composiciones, como para estudiantes, que pueden así evitar errores en sus propuestas, como incluso en musicología como herramienta de estudio de composiciones.

La aplicación debe ser capaz de detectar e indicar los errores (siguiendo los criterios de la armonía barroca) en obras a cuatro voces.

Lo anterior lleva a definir los siguientes requisitos.

Requisitos funcionales:

- **Req.FC01 - Leer un archivo MusicXML:** la aplicación debe ser capaz de leer un archivo en el estándar MusicXML y obtener de él toda la información que necesite, siendo su único input.
- **Req.FC02 - Comprobar las reglas de la armonía barroca:** la aplicación debe ser capaz de comprobar la aplicación de las reglas de la armonía barroca (especificadas en la sección 3.2) sobre el documento que reciba.
- **Req.FC03 - Escribir un archivo de salida:** la aplicación debe ser capaz de escribir los resultados en un archivo de texto.
- **Req.FC04 - Modificar un archivo MusicXML:** para el generador de casos de prueba, la aplicación debe ser capaz de modificar un archivo en el estándar MusicXML, escribiendo también un registro con los cambios introducidos.

Requisitos no funcionales:

- **Req.NC01 - Interfaz gráfica:** la aplicación debe tener una interfaz gráfica que permita su uso sin necesidad de formación adicional.
- **Req.NC02 - Mostrar los resultados en la interfaz gráfica:** además de ser almacenados en un archivo, los resultados deben ser mostrados en la interfaz para facilitar su lectura por parte del usuario.

4.2 Diseño

El diseño de la aplicación se ha realizado partiendo de las reglas de la armonía barroca y del formato MusicXML. Con esos dos elementos se ha diseñado el sistema de estructuras de datos y algoritmos que se describe a continuación:

4.2.1 Estructuras de datos

Siguiendo el modelo utilizado en el estándar MusicXML y para facilitar el trabajo con dicho formato, se han implementado una serie de estructuras de datos para organizar la información. La nomenclatura utilizada para las estructuras y sus miembros proviene en la mayoría de casos del nombre que tienen en su representación MusicXML. Las estructuras de datos son las siguientes:

- **Sound**
  La estructura*Sound* representa el nombre de una nota, compuesto por el propio nombre y las posibles alteraciones que lo acompañen. Por tanto los posibles valores para *Sound* son combinaciones de los nombres (*Step*) "A", "B", "C", "D", "E", "F" y "G" (estos son los nombres utilizados en la notación anglosajona, equivalentes a "la", "si", "do", "re", "mi", "fa", "sol" respectivamente en la notación del sur de Europa)con las distintas alteraciones (*Accidental*): bemoles (flat, flat-flat), sostenidos (sharp, double-sharp) y becuadros (natural).

- **Note**
  Note representa un sonido concreto y los elementos que lo constituyen en MusicXML, en concreto:
  - Su nombre (de tipo *Sound*).
  - Su altura (existe un sistema numérico, el índice acústico científico**,** para distinguir la altura de un sonido en función de la octava a la que pertenezca, siendo por ejemplo la nota La del segundo espacio en clave de Sol a 440Hz en esta notación "la 4").
  - La figura (*Type*, como una cadena de texto).
  - La duración (no está asociada directamente a la figura como se hace en la notación musical estándar; para simplificar cálculos el estándar MusicXML toma como unidad temporal de referencia la figura más pequeña en el documento, siendo por tanto una medida relativa).

- **Measure**
  Un compás (*Measure*) es simplemente una lista de notas. Su función es principalmente organizativa siguiendo el formato de MusicXML. La lista no tiene ninguna restricción en cuanto a cantidad de elementos, ya que en este nivel se desconoce la duración de cada nota. De este modo, también MusicXML da soporte a la música sin compasear.
  El único control realizado es en los propios editores de música, que no permiten escribir figuras que excedan la duración del compás y que lo autocompletan con silencios en caso de tener una duración inferior a la debida.

- **Key**
  La representación de la armadura (*key* en inglés) es una transcripción de la utilizada en MusicXML. Hay un valor entero (*fifths*) que indica el número de alteraciones: si es positivo se

tratará de sostenidos y si es negativo de bemoles. El otro elemento de esta estructura es el modo: menor o mayor. Con estos elementos se obtienen todas las escalas mayores y menores, las que se utilizarán en el contexto de la armonía barroca, según se describe en la siguiente figura:

	7♭	6♭	5♭	4♭	3♭	2♭	1♭	-	1♯	2♯	3♯	4♯	5♯	6♯	7♯
fifths	-7	-6	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5	+6	+7
mayor	Do♭	Sol♭	Re♭	La♭	Mi♭	Si♭	Fa	Do	Sol	Re	La	Mi	Si	Fa♯	Do♯
menor	La♭	Mi♭	Si♭	Fa	Do	Sol	Re	La	Mi	Si	Fa♯	Do♯	Sol♯	Re♯	La♯

Tabla 4.2-1 Escalas mayores y menores para distinto número de alteraciones

- **Voice**
  Cada una de las voces del documento está representada por una estructura de datos *Voice*. Esta estructura de datos contiene:
  - Una lista de compases (de tipo *Measure*).
  - El nombre de la voz, la clave (como combinación del tipo de clave y la línea en la que se encuentra).
  - La armadura (de tipo *Key*).
  - Un flag para comprobar si es una voz intermedia.
  - La unidad mínima de tiempo, relacionada con la variable duración del tipo *Note*. Este campo duración es de hecho redundante ya que se podría calcular a partir del tiempo mínimo y la figura, pero dado que se incluye en el estándar MusicXML se almacena para evitar calcular el valor cada vez que se requiera.

Además de las estructuras de datosque extienden el formato MusicXML han sido necesarias otras estructuras de datos complementarias para adaptar los datos a las necesidades de la aplicación:

- **Chord**
  Con *Chord* se representa un acorde, determinado por una tonalidad, un grado de la misma y la nota fundamental. Es una representación sencilla (incluso el hecho de tener el grado y la fundamental es información redundante que se guarda para evitar que se calcule cada vez que se acceda) gracias a que en la armonía los acordes que se usan son tríadas (compuestos de tres sonidos aunque alguno esté duplicado en varias voces). Para determinar los acordes se utiliza un sistema de emparejamientos que se describirá más adelante.

- **Scale**
  Scale es una estructura más alejada del estándar MusicXML que las anteriores. Además de representar una escala (determinada por una fundamental de tipo *Sound* y un modo) incluye información sobre el uso en el tiempo de la misma con las coordenadas (en formato <compás, unidad de tiempo>) de su inicio y su fin. Está muy relacionada con la estructura *Key* y de hecho se han implementado funciones para convertir la una en la otra.

4.2.2 Flujo

En la Figura 4.2-1 se puede observar el flujo de la información en la aplicación. Las flechas gruesas indican entradas y salidas, las cajas verdes el formato de los datos, las cajas azules las aplicaciones y las rojas los módulos de la aplicación que intervienen. De este modo la aplicación recibe como entradas un archivo MusicXML y la configuración seleccionada en la propia interfaz de la aplicación. La información contenida en el archivo se convierte al formato de estructuras de datos utilizado por la aplicación, y esta información se procesa a través de las reglas seleccionadas previamente.

A continuación se imprime en la interfaz gráfica el resultado de las reglas, y tras procesar este texto, se escribe una versión detallada en el archivo que haya sido indicado como salida de la aplicación.

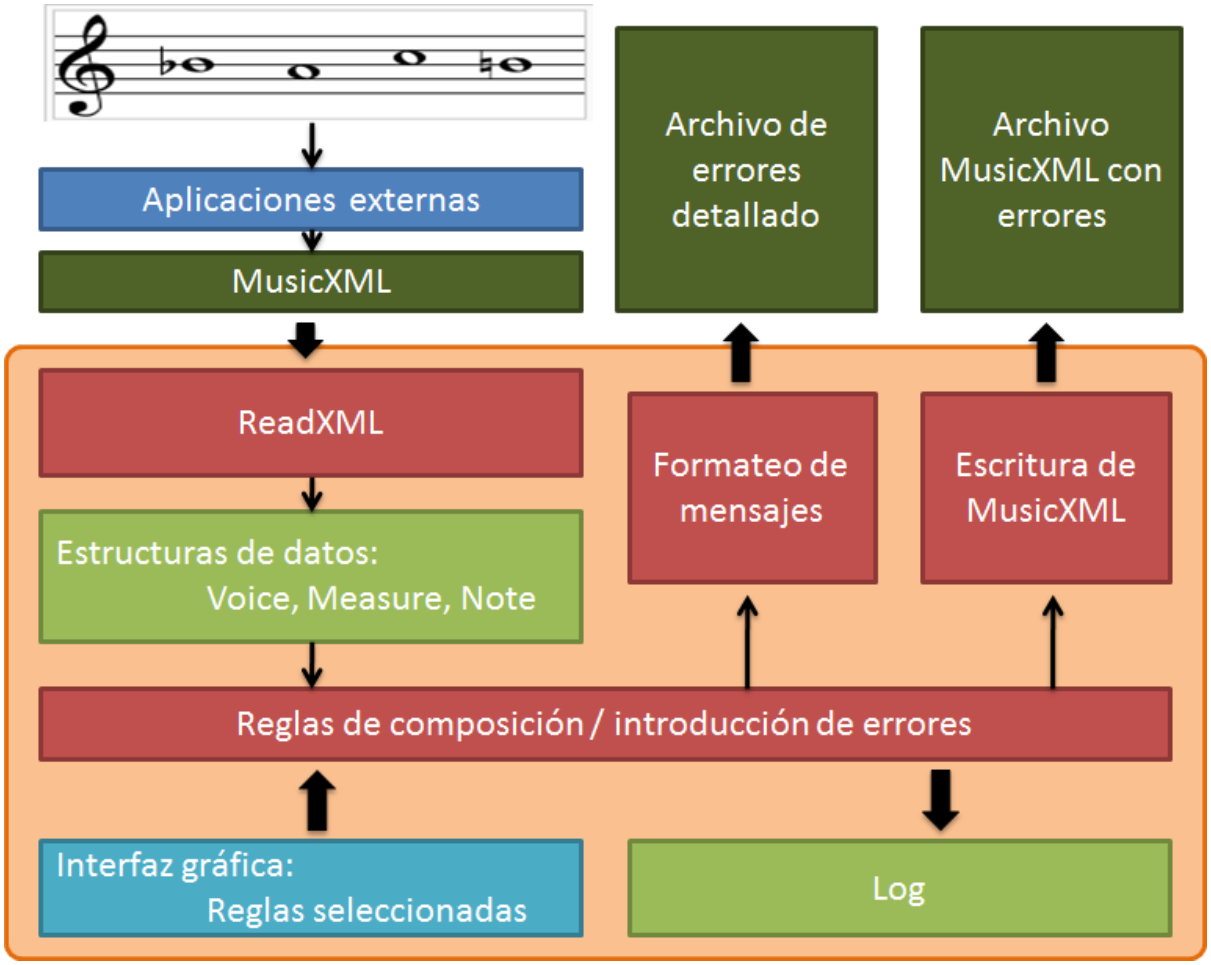


Figura 4.2-1 Diagrama de flujo de la aplicación

Código de colores:

- Rojo: módulos de la aplicación
- Azul claro: interfaz gráfica de la aplicación
- Azul oscuro: aplicaciones externas
- Verde claro: datos internos
- Verde oscuro: datos externos

4.2.3 Reglas

En cuanto al funcionamiento de las reglas se pueden distinguir dos casos: las que afectan a una sola voz y las que afectan a varias. A continuación se describe el tratamiento general de estas reglas:

- Las reglas que afectan a una voz iteran sobre cada nota de la voz comparándola con la anterior (salvo la primera que no se compara con ninguna otra) y comprobando con estas dos notas la aplicación de las reglas. Si estas notas son iguales, la comprobación no se realiza (ahorrando tiempo de cálculo). El pseudocódigo general es el siguiente:

```
foreach (measure in voice)
      noteP = null; //nota previa
      foreach (note in measure)
            if (note==noteP) continue;
            if(noteP!=null) regla (note, noteP);
            noteP=note;
```

- Las reglas que afectan a varias voces siempre toman las voces por pares. Iteran sobre cada nota de una de ellas y la comparan con la nota que suene en la misma unidad de tiempo en la otra voz. Si la nota en cualquiera de las voces es igual al anterior, la comprobación no se realiza. Cuando hay comprobación, se consideran dos casos: con/sin nota previa. Para el primero de estos casos, el pseudocódigo general es el siguiente:

```
foreach (measure in voiceA)
      notePA = null; notePB = null; //notasprevias de ambasvoces
      foreach (noteA in measure)
            noteB = noteAt(time(noteA, voiceA), voiceB);
            if (noteA == notePA || noteB == notePB) continue;
            if (notePA != null &&notePB != null)
                  regla (noteA, noteB, notePA, notePB);
            notePA = noteA; notePB = noteB;
```

- Para las reglas que afectan a varias voces y en las que no es necesario el almacenamiento de las notas previas, el pseudocódigo general es el siguiente:

```
foreach (measure in voiceA)
      foreach (noteA in measure)
            noteB = noteAt(time(noteA, voiceA), voiceB);
            regla (noteA, noteB);
```

  En ambos casos se aplica finalmente la comprobación de la regla.

Hay una excepción a este tratamiento general, la regla de duplicación de sensible. En esta regla es necesario saber la tonalidad de la música, lo que se realiza mediante un sistema heurístico que denominamos **Matchmaking**, desarrollado en este trabajo y que se describe a continuación.

4.2.4 Sistema de emparejamientos: sistema heurístico *Matchmaking*

Se denomina **sensible** al séptimo grado de una tonalidad cuando se encuentra a una distancia de semitono de la tónica, llamándose en caso de distar un tono subtónica. La sensible se encuentra de manera natural en las escalas mayores, pero dada su importancia a la hora de reafirmar la tonalidad se ha incorporado a las escalas menores, creando las escalas menores armónicas. La única diferencia entre las escalas menores naturales y las armónicas es este séptimo grado:

Distribución de tonos (T) y semitonos (S)								
	I	II	III	IV	V	VI	VII	I
Escala mayor	T	T	S	T	T	T	S	
Escala menor natural	T	S	T	T	S	T	T	
Escala menor armónica	T	S	T	T	S	TS	S	
Escala menor melódica	T	S	T	T	T	T	S	

Tabla 4.2-2 Distribución de tonos (T) y semitonos (S) entre los distintos grados de varias escalas

El problema de las escalas menores armónicas es que tienen una segunda aumentada (un tono y un semitono) entre el VI y VII grado. La solución a ello es la escala menor melódica, que eleva también el VI grado para que todas las segundas sean mayores (un tono) o menores (un semitono).

Por tanto, la regla que prohíbe la duplicación de la sensible, esto es, que la sensible no puede aparecer simultáneamente en dos voces, ya que crea una sensación de inestabilidad armónica, no es trivial, ya que puede aparecer en tonalidades menores pese a no ser parte de ellas de manera natural.

Teniendo en cuenta que la sensible (cuando aparece) siempre ocupa una posición fija en las tonalidades (VII grado), el primer paso para la detección de errores de duplicación de sensible es la detección de tonalidades.

La resolución de este problema ha llevado al desarrollo en este trabajo de un módulo que trabaje con escalas y tonalidades (como se recoge en el glosario, una tonalidad es la dupla de tónica, nota más importante de la misma, y modo, menor o mayor. Una escala es la denominación del conjunto de distancias entre notas, o grados, a partir de una tonalidad).

El módulo se basa en **la detección de las tonalidades a partir de acordes tríada** (conjuntos de 3 notas a distancia de 3ª y 5ª respecto a la más grave, la fundamental). Una vez detectada la tonalidad, el problema se reduce a la identificación del VII grado de la escala mayor sobre la tónica (primer grado de la escala) de la tonalidad encontrada (independientemente de su modo).

En algunos archivos MusicXML se especifica la tonalidad, resultando la resolución del problema trivial. Para el resto de casos se ha desarrollado en este proyecto una heurística de emparejamientos a la que nos referiremos a partir de ahora como *matchmaking*.

Teniendo en cuenta la mayor presencia de los grados I, IV y V (por las funciones de aporte de estabilidad que desempeñan son los más habituales en la armonía anterior al Romanticismo, incluyendo la armonía del Barroco) se intenta agrupar el mayor conjunto de notas posible que

pertenezcan a un mismo acorde. Para estimar si un conjunto de notas pueden pertenecer o no a un acorde se calcula un peso medio a partir de la siguiente heurística:

- Si el sonido es la fundamental del acorde se suma 5.
- Si el sonido es la tercera o la quinta del acorde se suma 3.
- Si el sonido es la séptima del acorde se suma 1.

Una vez sumados estos valores, los acordes que superen un umbral son añadidos a la lista de posibles acordes para el conjunto de notas examinado.

La inclusión de la séptima se debe a que pese a que la gran mayoría de acordes son tríadas (fundamental, tercera y quinta) a veces se añade en el V grado la séptima.

Por ejemplo, tras leer una nota esta puede ser (ordenadas de mayor a menor probabilidad) fundamental, tercera o quinta de un acorde. Si se quiere comprobar la siguiente nota, al leerla se pueden dar dos casos diferentes:

- La nota pertenece a alguno de los posibles acordes de la nota anterior. En este caso se volvería a generar la lista de posibles acordes. Si ambas notas son la fundamental y la quinta, solamente puede tratarse de un acorde, ya que la ponderación de la séptima al ser promediada no llega al umbral que el *matchmaking* tienen en cuenta para considerar un acorde viable. Sin embargo, si ambas notas están a una distancia de tercera (fundamental+tercera o tercera+quinta) hay dos posibles acordes. Y si las dos notas son el mismo sonido, continúan los mismos tres posibles acordes).

  Una vez generada la nueva lista de posibles acordes se intentaría extender el grupo a la siguiente nota, realizando de nuevo esta comprobación de pertenencia a los acordes.

- Es una nota extraña y no pertenece a ninguno de los acordes anteriores. Aquí se distinguen dos situaciones: si la nota está a una distancia de segunda de la anterior o si no lo está.
  - Si está a una segunda podría tratarse de una nota de paso o de un floreo. Por tanto se comprueba la siguiente nota para confirmar o desmentir esta hipótesis. Si se confirma se continuaría intentando extender el conjunto de notas.
  - Si no se trata de una nota de paso o un floreo o no se encuentra a distancia de segunda, la nota pertenece a otro acorde distinto. El grupo anterior se cierra, asignándole el acorde más probable y se empieza otro grupo con esta nota como primer elemento.

Una vez se ha determinado un conjunto de acordes, es posible hallar cualquier grado: teniendo en cuenta que los acordes con mayor aparición son los de dominante y tónica (V y I grados respectivamente), el procedimiento es similar a un criptoanálisis por análisis de frecuencias. Una vez aislados estos dos grados, su distinción es trivial (por los intervalos entre sus fundamentales: entre I y V hay una 5ª, pero entre V y I hay una 4ª). De este modo se obtiene la tónica de la tonalidad. A partir de esa tónica, la aplicación construye una tonalidad mayor, donde el séptimo grado es siempre sensible.

En una melodía a varias voces, si ninguna de las notas pertenece a un acorde (esto es, hay un valor heurístico 0 para esa unidad de tiempo), no se probará a añadir esa nota al acorde, dejando la media

a 0 para forzar un fin de ese itinerario, de modo que se prosiga con el resto de acordes de esa unidad de tiempo que superen el umbral, o si ninguno lo hace, se cierre el grupo de notas en la unidad de tiempo anterior.

Este sistema tiene además otras potenciales aplicaciones. En la posible ampliación de generación de voces (de la que se hablará detalladamente más adelante) este sistema tiene una importancia vital. La generación de voces aleatorias que cumplan las normas de la armonía barroca implica el uso de los acordes de maneras concretas: no todos los enlaces están permitidos (por ejemplo III y VI apenas se usan, II y IV suelen enlazarse con V...) por lo que al devolver una lista de posibles acordes y no sólo un elemento se podría comprobar qué enlaces viables son más probables.

Por otro lado la generación "correcta" de voces (a diferencia de la generación de casos de prueba) conlleva el uso de notas de los acordes en lugar de notas extrañas, problema que también resuelve este sistema heurístico.

En el anexo 2 se muestra un ejemplo de funcionamiento del sistema de *matchmaking* sobre una partitura a cuatro voces.

4.3 Interfaz gráfica

La interfaz gráfica de la aplicación está desarrollada a partir de la librería *Windows Forms*, diseñada especialmente para realizar aplicaciones de escritorio. Consta de tres pestañas con las distintas funcionalidades ofrecidas:

- Comprobación de armonía
- Generación de casos de prueba
- Ayuda

4.3.1 Comprobación de armonía

En esta pestaña el usuario podrá introducir el nombre del archivo que desee comprobar y el nombre del archivo donde se escribirá el fichero detallado de errores, así como seleccionar las reglas que quiera que se comprueben sobre este archivo.

Una vez introducidos estos datos y pulsado el botón “Comprobar”, una lista de los errores le es devuelta en el panel derecho de la pestaña y en un archivo log en el que se plasman con mayor detalle. También se genera un archivo <nombre>ERRORES.xml en el que aparece indicada la ubicación de los errores mediante etiquetas de texto. A continuación se muestra la vista de esta pestaña en la aplicación:

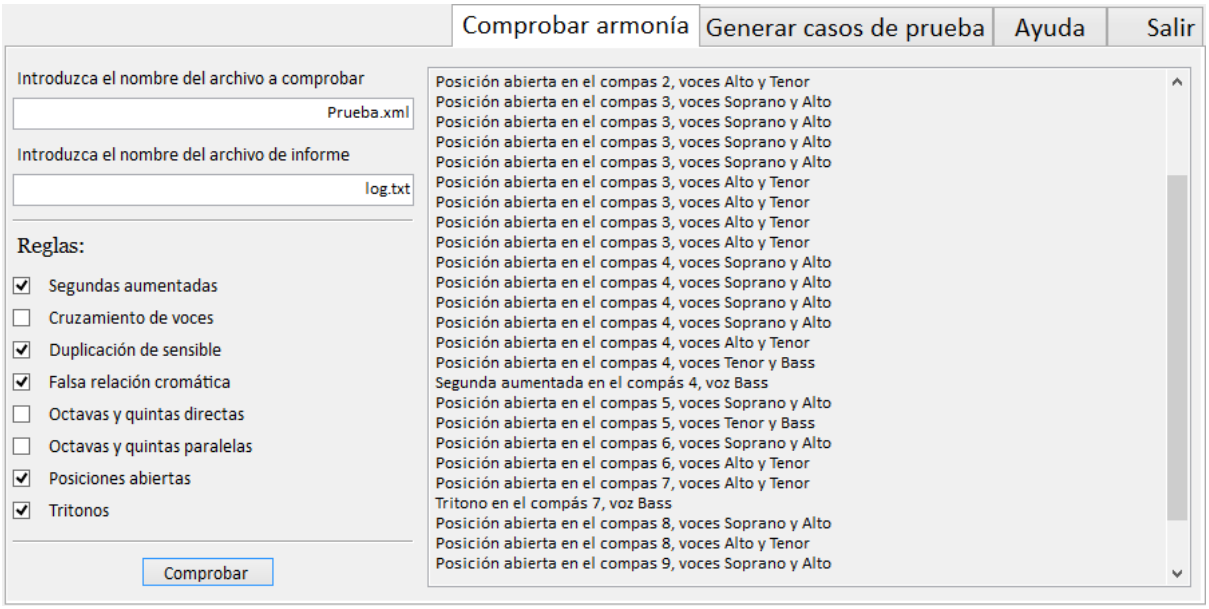


Figura 4.3-1 Vista de la aplicación: pestaña de comprobación de armonía

4.3.2 Generación de casos de prueba

En esta pestaña el usuario podrá introducir el nombre del archivo que quiera editar. Una vez introducido deberá pulsar el botón "cargar". Al realizar esta acción se mostrarán los posibles errores que se pueden generar en el documento en el panel derecho de la pestaña y se habilitarán las opciones de errores, cuyo número se puede seleccionar, tal como se muestra en la siguiente figura:

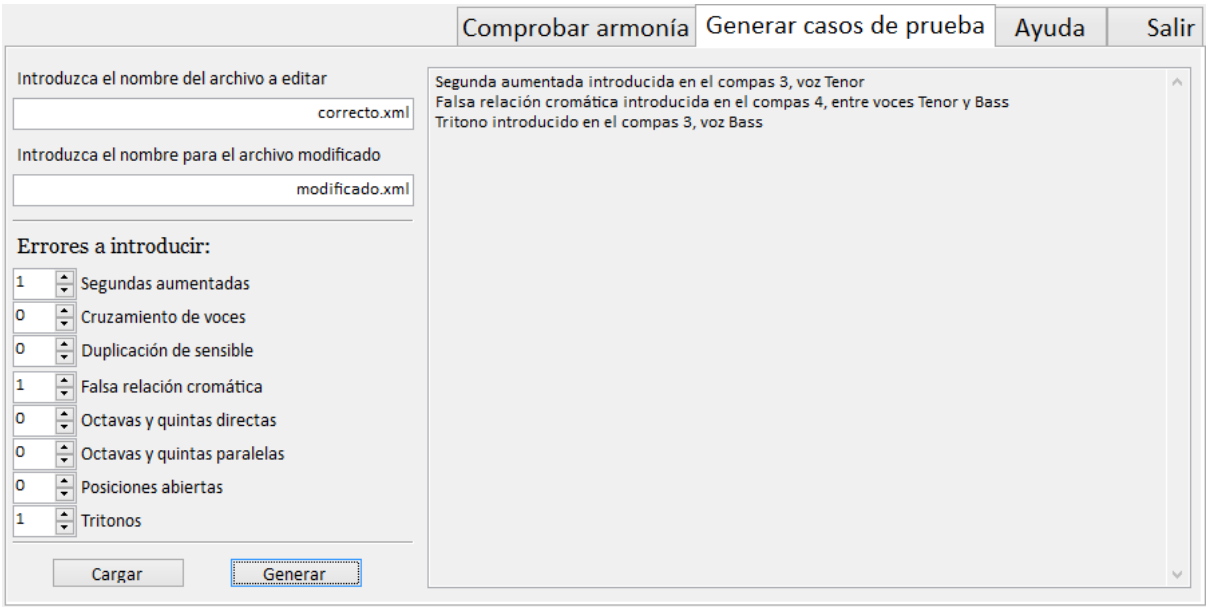


Figura 4.3-2 Vista de la aplicación: pestaña de generación de casos de prueba

A continuación podrá escoger los errores a introducir en la versión modificada del archivo, así como el nombre de este nuevo documento generado con los errores indicados. Cuando haya completado estos parámetros deberá pulsar el botón "Generar" con lo que se creará el archivo, apareciendo en el panel derecho de la pestaña la lista de errores introducidos.

4.3.3 Ayuda

En esta pestaña el usuario dispondrá de una serie de mensajes de ayuda, tanto acerca de la aplicación como sobre las reglas de la armonía barroca. Al pulsar en cualquiera de los títulos de ayuda, este se desplegará mostrando el texto completo referente a ese tema. Si el usuario vuelve a pulsar en un título ya desplegado, este se contraerá. A continuación se muestra una vista de esta pestaña con algunos títulos desplegados:

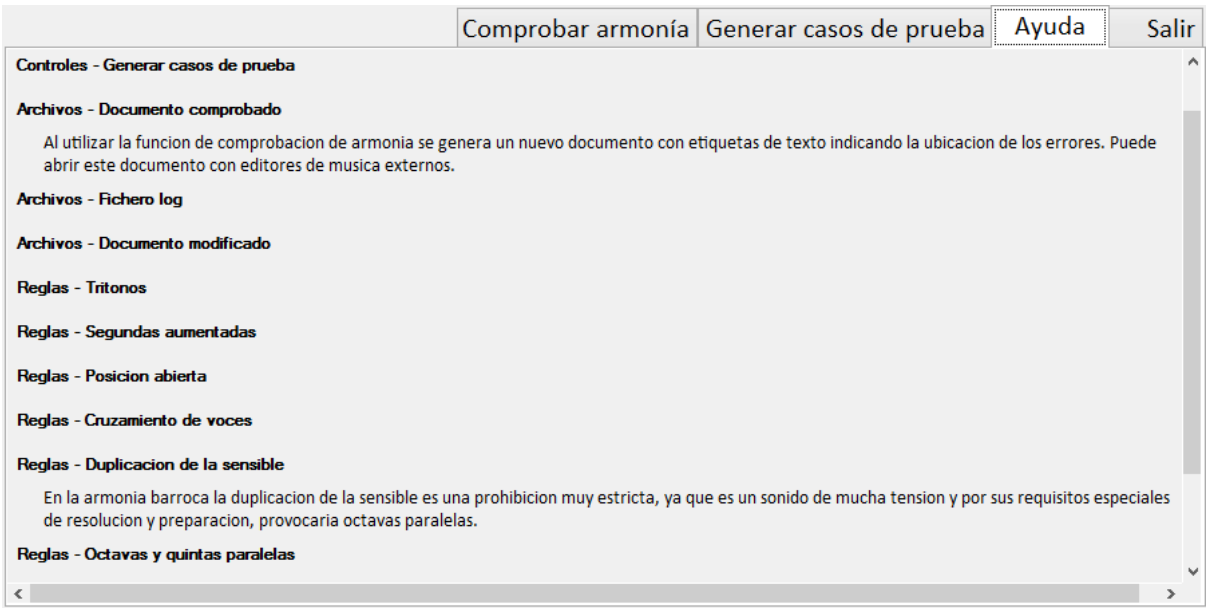


Figura 4.3-3 Vista de la aplicación: pestaña de ayuda

4.4 Codificación y pruebas

A continuación se presentan los conceptos tenidos en cuenta en el desarrollo de la aplicación y en la realización de pruebas sobre la misma.

4.4.1 Codificación

La codificación de la aplicación se ha llevado a cabo en C#. La elección del lenguaje se debe a que es un lenguaje orientado a objetos (facilitando el desarrollo de la interfaz gráfica) y para facilitar una posible futura adaptación del sistema a un formato web a través de .NET (ver sección 01).

Para estructurar el código se ha seguido un patrón modelo-vista-controlador, buscando la mayor independencia posible entre unos módulos y otros. En la figura 4.4.1 se muestra la estructuración del código:

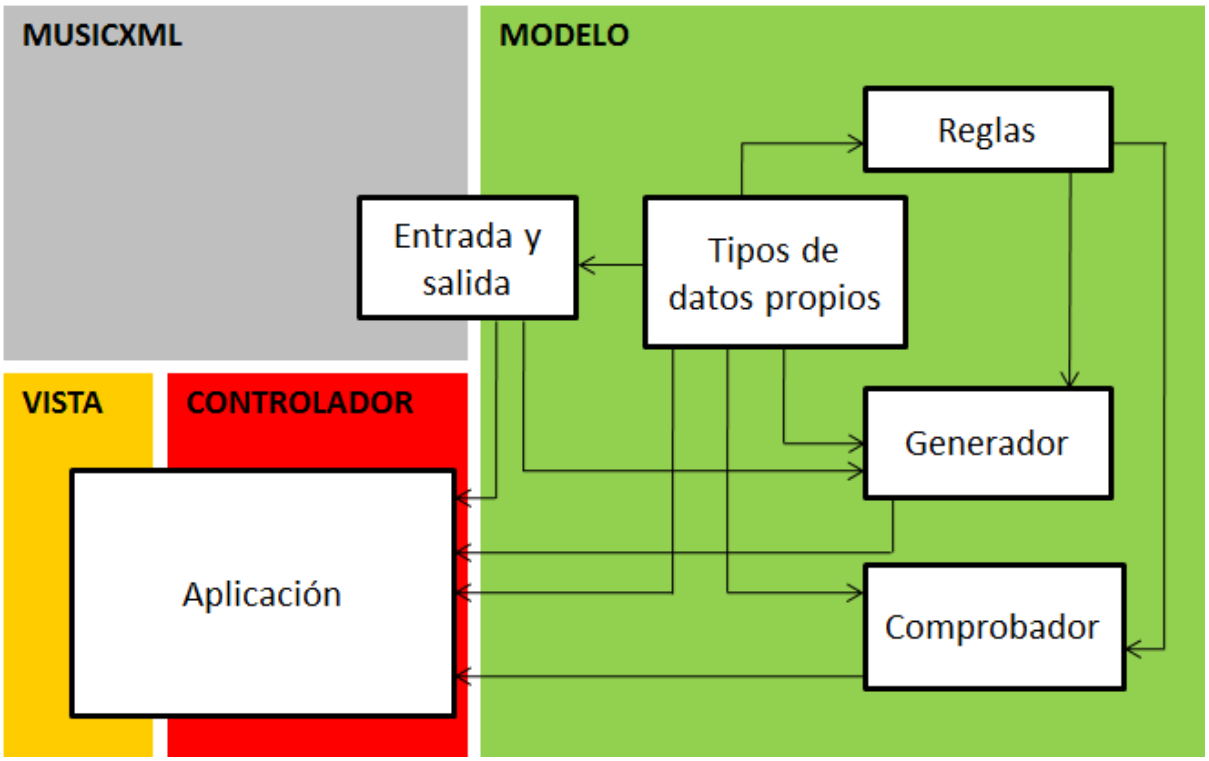
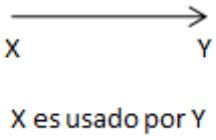


Figura 4.4-1 Diagrama de diseño de la aplicación

Leyenda:

- Cajas blancas: representan cada uno de los módulos en los que se ha dividido el código.
- Área verde: agrupa los módulos pertenecientes al modelo.
- Área amarilla: agrupa los módulos pertenecientes a la vista.
- Área roja: agrupa los módulos pertenecientes al controlador.
- Área gris: representa el formato MusicXML, los módulos que se encuentran en esta área trabajan directamente con este estándar.
- Flechas: representan una relación de uso de la siguiente manera:



Tanto la vista como el controlador se encuentran implementados en el mismo módulo; sin embargo se respeta el patrón, ya que se encuentran en archivos diferentes. Esto es así por la estructura de C#, en la que los archivos de interfaz están compuestos (entre otros) por un archivo *designer* que contiene la interfaz gráfica propiamente dicha (vista) y un archivo de formulario que enlaza esa interfaz con la lógica del programa (controlador).

En el modelo, el módulo básico es el que contiene los tipos de datos. A partir de éste se implementan las reglas y se leen y escriben los datos MusicXML. Tanto la parte de comprobación de reglas (Comprobador) como la de generación de casos de prueba (Generador, ver sección 4.5)

utilizan también los tipos de datos, así como las reglas. Además la generación de casos de prueba utiliza el módulo de entrada y salida para escribir el archivo de salida en MusicXML.

La interacción de los distintos módulos con las estructuras de datos (tipos de datos propios) se encuentra limitada a funciones públicas y nunca hay un acceso directo a las estructuras desde otros módulos. De este modo se garantiza la independencia entre módulos, ya que aunque se modifiquen las estructuras de datos y sus funciones, si se mantienen las cabeceras no es necesario cambiar el resto de módulos. También así se deja abierta la posibilidad a la futura generación de una API de la aplicación para otros programadores (ver subsección 0).

4.4.2 Pruebas

Las pruebas sobre la aplicación se han realizado comparando la salida de la aplicación con correcciones analógicas para los mismos documentos de entrada. Estos documentos han sido de varios tipos:

- **Documentos de errores específicos,** a modo de pruebas unitarias para cada tipo de error se han diseñado archivos sencillos con distintos casos de cada error (ver sección 3.2). Con estos documentos se prueba la funcionalidad de las distintas reglas de comprobación.
- **Ejercicios de armonía**, donde estos archivos permiten simular el funcionamiento para el que ha sido desarrollada la aplicación: su uso como herramienta de ayuda docente a los profesores de armonía. Los documentos de este tipo son transcripciones a MusicXML de ejercicios reales, por lo que el funcionamiento para estos ejercicios es el más fiel reflejando las situaciones a las que la aplicación tendrá que enfrentarse.
- **Documentos musicales**, esto es, obras musicales ya existentes transcritas a MusicXML. En este caso la finalidad no es tanto comprobar el funcionamiento de las reglas (ya se ha probado con los tipos de documentos anteriores) como comprobar la integración de la aplicación con el estándar MusicXML y su respuesta ante tags poco habituales que pueda encontrarse en los mismos: comentarios en las partituras, silencios, puntillos, indicaciones...
- **Archivos de salida del generador de casos de prueba**: como parte de la aplicación se ha desarrollado un generador de casos de prueba que modifica los archivos que recibe como entrada para introducir los errores que el usuario requiera (ver sección 4.5). Las pruebas sobre esta parte de la aplicación también sirven como verificación de la parte de comprobación de reglas de la armonía barroca, ya que al analizar el archivo de salida han de obtenerse los mismos errores que se ha pedido introducir a la aplicación.

4.5 Generador de casos de prueba

Para facilitar la comprobación del correcto funcionamiento de la aplicación, se ha desarrollado también un generador de casos de prueba que añada incumplimientos de las reglas a archivos MusicXML existentes. Este generador puede ser también usado por los profesores de armonía para la creación de ejercicios donde los estudiantes deban detectar manualmente los mismos.

4.5.1 Análisis de requisitos

Además de los requisitos de la aplicación en general (ver sección 4.1) el generador de casos de prueba deberá ser capaz de:

- **Req.FG01 - Aleatorizar un punto de error:** dada la información de un archivo en formato MusicXML, obtener un punto aleatorio donde se introducirá el error.
- **Req.FG02 - Generar casos de error:** modificar el archivo original de modo que se incumpla alguna o algunas de las reglas de la armonía barroca.
- **Req.NG03 - Integrar con la aplicación:** el generador se debe adaptar a la interfaz gráfica existente en la parte de comprobación para facilitar su uso.

4.5.2 Diseño

A las estructuras de datos de la aplicación principal se añade una ampliación del sistema de coordenadas presentado en *Scale* (ver subsección 4.2.1). El formato de este sistema es

<voz, compás, nota>.

La motivación de la ampliación es adaptar el sistema a un conjunto de datos completo (como lista de *Voice*), ya que en la parte de detección de errores el sistema de coordenadas funciona internamente en cada voz y por tanto no es necesario este parámetro. Con esta modificación del sistema, se han implementado funciones para aleatorizar estos tres parámetros. Además se han desarrollado funciones que devuelven dos valores aleatorios para el parámetro voz, tanto voces separadas como voces contiguas, en lugar de uno. De este modo se facilita el cálculo del punto de error para las reglas que afectan a varias voces.

El segundo elemento a tener en cuenta al generar los errores es el tipo de error. Para ello, el usuario escogerá mediante la interfaz gráfica el tipo de error a introducir. Hay una limitación en cuanto al número máximo de errores, ya que al introducir muchos podrían corregirse unos a otros. Además, la introducción de un error forzado puede generar otros no previstos, hecho del que se informa al usuario mediante un mensaje de la aplicación.

Una vez seleccionado el punto de error y el tipo, la aplicación sustituye la nota en ese punto por otra que cumpla las condiciones del error. En este proceso se distinguen tres casos diferentes:

- Reglas que afectan a una voz: para las segundas aumentadas y los tritonos se calcula el punto de error y en ese punto se introduce un elemento erróneo aleatorizado (en el caso de las segundas aumentadas el parámetro aleatorizable es la dirección del intervalo y en el caso de los tritonos tanto la altura como el tipo del intervalo). Las nuevas notas a sustituir se calculan a partir de la anterior al punto de error.

```
coord = errPoint();
note = noteAt(voice, coord);
newNote = interval(noteAt(voice, coord - 1));
swap (note, newNote);
```

- Reglas que afectan a dos voces que no tienen en cuenta notas previas, esto es, las reglas sobreposiciones abiertas, cruzamientos y duplicación de sensible. En estos casos al calcular el punto de error se pide una voz extra; en el caso de posiciones abiertas y cruzamientos,esta voz debe ser la contigua inferior o superior (cuando sea posible). Una vez obtenidas las dos voces (y el resto de las coordenadas) se altera la primera voz a partir de la nota de la segunda.

```
coord = errPoint();
v2 = coord.SeconVoice(voces);
note = noteAt(voice, coord);
newNote = interval(noteAt(v2, coord));
swap (note, newNote);
```

- Reglas que afectan a dos voces y tienen en cuenta notas previas: octavas y quintas directas y paralelas y falsas relaciones cromáticas. Aquí ocurre una combinación de los dos casos anteriores, siendo necesario obtener una segunda voz aleatoria y la nota anterior a la del punto de error.

```
coord = errPoint();
v2 = coord.SeconVoice(voces);
note1 = noteAt(voice, coord);
note2 = noteAt(v2, coord);
newNote1 = interval(noteAt(voice, coord - 1));
newNote2 = interval(noteAt(v2, coord - 1));
swap (note1, newNote1);
swap (note2, newNote2);
```

  Este escenario es el más variado en cuanto al cálculo de las nuevas notas. En los casos más simples se soluciona (al igual que en las otras situaciones) con el cálculo de un intervalo respecto a la nota anterior; sin embargo, a veces es necesario calcular además de estas relaciones, las relaciones interválicas entre las dos voces.

Los parámetros adicionales que sean necesarios en cada regla concreta se seleccionan mediante listas, escogiendo una posición aleatoria de esa lista. Por ejemplo, en el caso de los tritonos, estos pueden ser tanto intervalos de 4ªA como de 5ªD, y para ambos casos tanto descendentes como ascendentes. Con ese concepto, se introducen en una lista los valores {-5, -4, 4, 5}, obteniendo el tipo del intervalo con el que se obtendrá la nueva nota.

Siguiendo el mismo ejemplo, el cálculo de la especie del intervalo no es necesario, ya que la función encargada de este proceso recibe como argumento la distancia total (en semitonos) y no la especie del intervalo, por lo que la llamada simplemente sería <valor lista, 6>.

4.5.3 Codificación

Tal como se explica en la subsección 4.4.1, gracias a la modularización del código la implementación del generador de casos de prueba es fácilmente integrable con el resto del código.

De manera similar al comprobador de reglas, esta nueva funcionalidad se codifica en un módulo separado que utilizará funciones de los módulos de **reglas** y de **tipos de datos**. Además, debido a que los casos de prueba generados deben ser devueltos en formato MusicXML, este módulo utilizará también las funciones de **entrada y salida**.

En cuanto a las reglas, se añaden al módulo **reglas** las funciones de aleatorización del punto de error y las funciones necesarias para forzar las reglas (siguen una filosofía similar a las reglas "comprobables" pero son independientes de ellas en código).

Las pruebas de este módulo se realizan mediante la parte de la aplicación de comprobación de reglas, ya que los nuevos errores introducidos son detectados por la aplicación base.

4.5.4 Ejemplo de tipo de error

A continuación se muestra el proceso que sigue el sistema para generar un error (Figura 4.5-1). En este caso el error a generar es un cruzamiento de voces. Para ello se debe disponer de dos voces contiguas (calculadas aleatoriamente). Una vez obtenidas esas dos voces, en el punto de error (compás y unidad de tiempo) se intercambian (si es necesario) las notas de ambas voces, de modo que la nota más aguda quede en la voz inferior y la nota más grave en la superior.

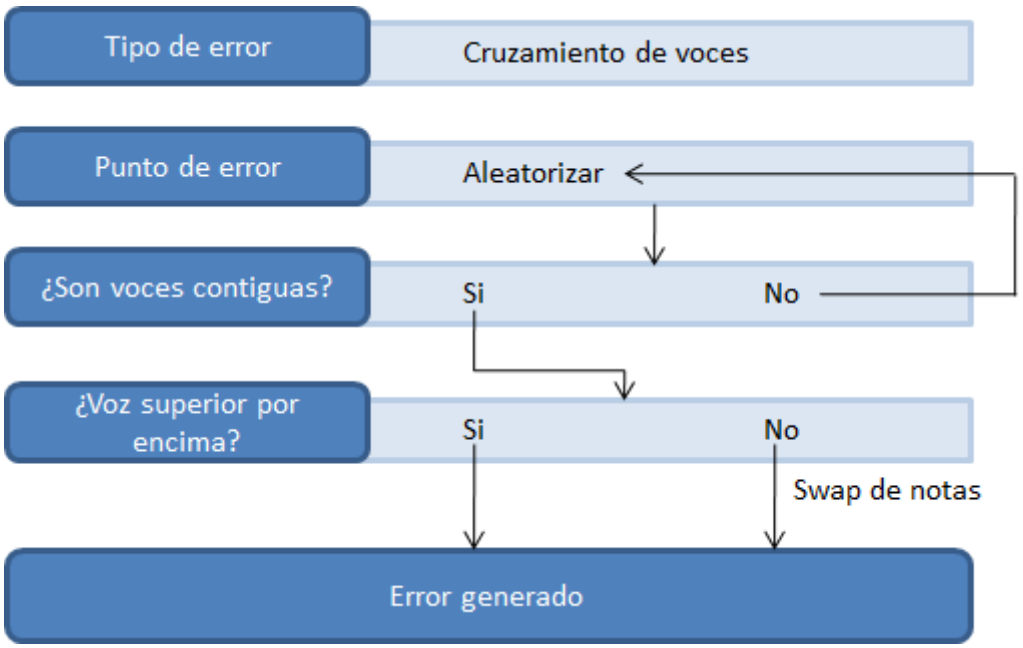


Figura 4.5-1 Ejemplo de generación de errores

5 Ejemplos de uso

En este capítulo se presentan como ejemplos del uso de la aplicación los resultados de la misma sobre distintas partituras. Los archivos base son:

- ejercicio.xml, ejercicio real de un alumno de armonía.
- bach_bwv2.xml, fragmento de una composición a cuatro voces de Johann Sebastian Bach.
- mahler_lieder.xml, sección de una obra para voz y piano de Gustav Mahler.

5.1 Ejercicios de armonía

En primer lugar se muestra un ejercicio real de las enseñanzas de armonía. Los errores más habituales en los alumnos de armonía son la presencia de octavas y quintas paralelas en los primeros cursos y la duplicación de la sensible y uso de tritonos y segundas aumentadas en los cursos superiores. Esto se debe a un enriquecimiento de los acordes usados (al principio sólo se usan acordes tríada pero luego se añaden acordes de otras tonalidades, acordes con séptima y acordes con novena), lo que acarrea una mayor posibilidad de cometer fallos.

Este ejercicio concreto se basa en una composición sencilla a cuatro voces donde la voz aguda corresponde al himno zarista ruso. El objetivo de esta actividad es escribir tres voces adicionales, utilizando una lista de acordes proporcionados por el docente. No busca hacer hincapié en ninguna norma concreta, sino que se trata de una revisión general de las mismas.

Al comprobar el ejercicio de 16 compases elaborado por un estudiante de segundo año de armonía utilizando la aplicación se obtienen los siguientes resultados:

- 5 octavas y quintas paralelas.
- 2 octavas y quintas directas.
- 1 tritono.

Por tanto aquí no hay errores debidos a posiciones abiertas, cruzamientos de voces, duplicaciones de sensible, falsas relaciones cromáticas o segundas aumentadas.

A continuación se muestra una parte del fragmento con uno de estos incumplimientos (octavas paralelas y directas) señalado:

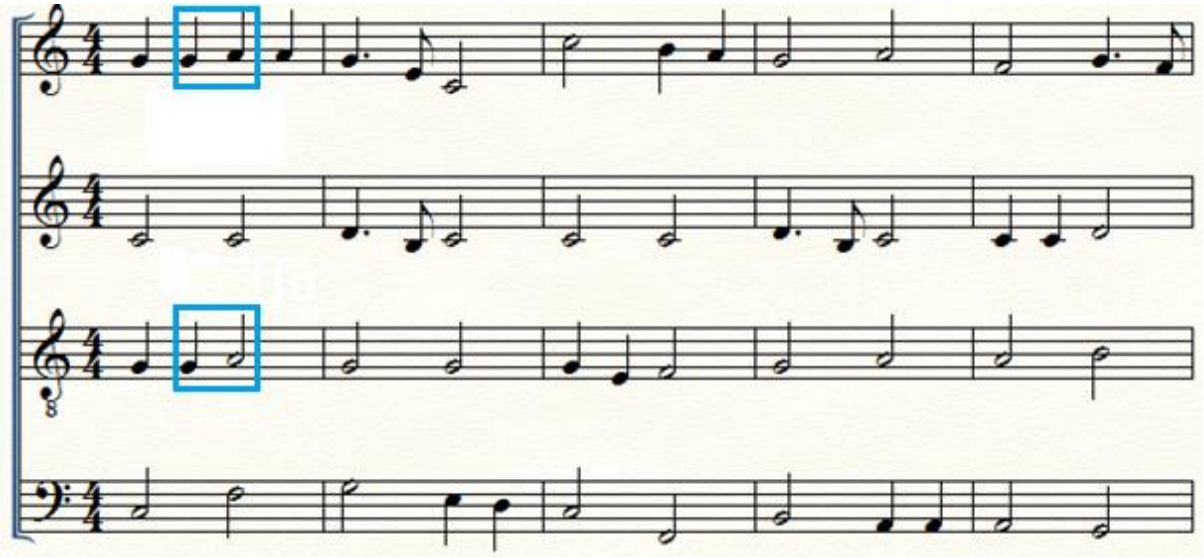


Figura 5.1-1 Ejercicio sobre el himno zarista, alumno de armonía

5.2 Ach Gott, vom Himmel sieh darein

La segunda obra de prueba es un fragmento de la cantata (basada en un preludio coral) *Ach Gott, vom Himmel sieh darein*, BWV 2 de Johann Sebastian Bach. Fue estrenada en 1724 y es una cantata coral, esto es una obra compuesta para una misa en varios movimientos y a cuatro voces (aunque no intervengan en todos los movimientos). Está basada en el himno homónimo de Lutero y su título significa "*Oh Dios, mira desde el cielo*".

Este fragmento ha sido escogido porque representa el estilo que se está trabajando con la aplicación. Pese a que cronológicamente J. S. Bach correspondería a un Barroco tardío, su estilo es más similar al de la música anterior a él, y es de hecho uno de los compositores más importantes del Barroco.

Al introducir este fragmento de 16 compases en la aplicación encontramos los siguientes incumplimientos de normas:

- 11 octavas y quintas directas.
- 1 tritono.

Por tanto aquí no hay errores debidos a octavas y quintas paralelas, posiciones abiertas, cruzamientos de voces, duplicaciones de sensible, falsas relaciones cromáticas o segundas aumentadas.

Pese a la presencia de estos incumplimientos de las normas, estos aparecen siempre en voces que en ese momento tienen menor importancia, por lo que auditivamente pasan desapercibidos.

A continuación se muestra una parte del fragmento con uno de estos incumplimientos (una quinta directa) señalado:

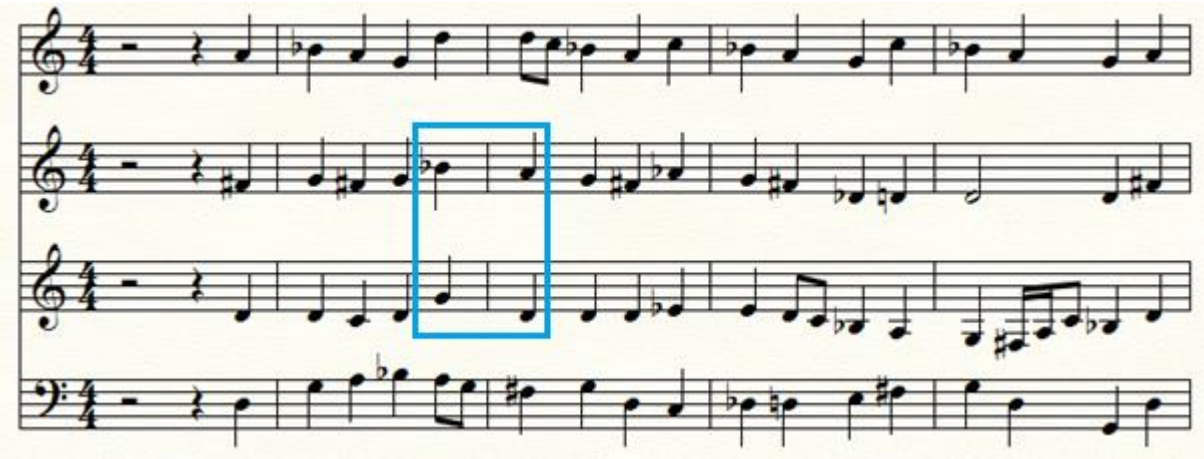


Figura 5.2-1 *Ach Gott, vom Himmel sieh dahrein*, J. S. Bach

5.3 Lieder eines fahrenden Gesellen

La última obra escogida es un fragmento del ciclo de canciones *Lieder eines fahrenden Gesellen* (Canciones de un compañero de viaje) de Gustav Mahler. Estas canciones fueron publicadas en 1897 y están escritas para voz y piano. La elección de este pequeño fragmento de 11 compases se debe a que es un lenguaje muy distinto al del Barroco, en el que por ejemplo los tritonos ya no son mal vistos. Dentro de la corriente tradicional de la música (sin entrar en la música contemporánea) es uno de los lenguajes más complejos y completos que se puede encontrar. Pese a no estar en el formato de cuatro voces como los otros ejemplos, puede ser igualmente analizado por la aplicación comparando la voz cantada y las dos voces piano, obteniendo los siguientes resultados:

- 1 octavas y quintas paralelas.
- 6 octavas y quintas directas.
- 8 tritonos.

Por tanto aquí no hay errores debidos a posiciones abiertas, cruzamientos de voces, duplicaciones de sensible, falsas relaciones cromáticas, segundas aumentadas.

La presencia de tritonos en este lenguaje no es considerada error, y de hecho son un elemento completamente habitual desde finales del siglo XIX. Las octavas y quintas directas y paralelas son resultado del tipo de escritura (voz y piano), sirviendo como apoyo del piano a la voz, por lo que no se considerarían errores.

A continuación se muestra una parte del fragmento con uno de estos incumplimientos (octavas paralelas y directas) señalado:

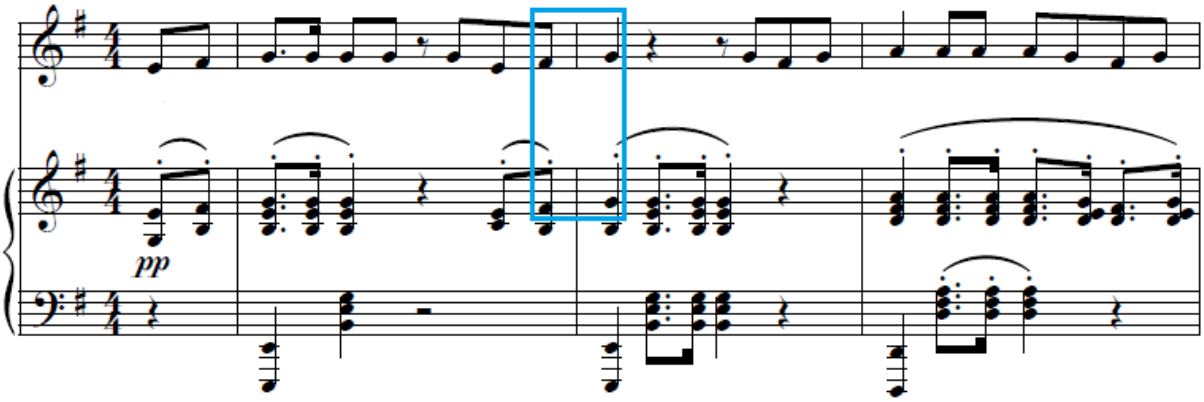


Figura 5.3-1 Lieder eines fahrenden Gesellen, G. Mahler

6 Resumen, conclusiones y posibles ampliaciones

6.1 Resumen

En este trabajo se presenta el desarrollo de una aplicación de ayuda docente para las enseñanzas musicales de armonía. La aplicación comprobará sobre un documento que se le proporcione (habitualmente composiciones a cuatro voces) una serie de normas utilizadas en la música barroca, estilo utilizado para el aprendizaje de la armonía. El resultado de esta comprobación se muestra tanto en la aplicación como en un fichero externo, el *log*, en el que se detallan los incumplimientos de dichas normas. También se devuelve una versión modificada del archivo de entrada en el que se han señalado los puntos de error mediante etiquetas de texto.

Además, la aplicación incluye una segunda funcionalidad que es la generación de errores a partir de un documento proporcionado. Esto es, modificar el archivo de entrada de modo que se fuercen incumplimientos de las normas de la armonía barroca. La cantidad de errores es elegida por el usuario y su ubicación en el documento es aleatorizada. Se devuelve un archivo con las modificaciones introducidas y etiquetas de texto que señalen el lugar donde se han producido.

Se considera, entre otras, una posible ampliación para que la aplicación genere a partir de un documento musical con una única voz tres voces adicionales que cumplan las normas de la armonía barroca.

La motivación de este trabajo es la ayuda a los docentes, que actualmente realizan la corrección del trabajo de sus alumnos a mano, con lo que se podría mejorar el tiempo de corrección y así dar más tiempo en las clases a explicaciones. Actualmente no existen aplicaciones que cubran esta funcionalidad.

6.2 Summary

This project exposes the development of an application intended for helping with musical education as a helping tool for teachers. The application will check the fulfillment of a set of rules used on baroque harmony (Baroque is the standard style when talking about teaching harmony) on a provided document (generally a four-voice composition). The results of this checking are shown both in the application's interface and in an extern file, log file, where non-fulfillments of rules are detailed. There is another output file, a modified version of input file with text labels pointing out the error points.

The application also includes a second functionality which is error generation from a provided document. That is, modifying input file so non-fulfillments of baroque-harmony rules are forcedly introduced. The amount of errors the application will introduce is chosen by user and their placement in the document is generated randomly by application. An output file is created by modifying the input file with the required modifications and text labels pointing out where those placements.

An improvement is considered, amongst many others, consisting in the generation of three fulfilling baroque harmony rules voices from a provided single-voiced document.

Motivation for this project is to provide musical education teachers with a digital tool to reduce the amount of time they put in correcting their students' homework, as currently they have to check each exercise manually. With this saving of time they would be able to focus on explanations. At the present time there isn't any application fulfilling this functionality.

6.3 Conclusiones

A la vista del presente trabajo se puede concluir que:

- Actualmente existen numerosas aplicaciones de informática musical, la mayoría de ellas usando el estándar MusicXML. Sin embargo no existe ninguna aplicación para comprobar las normas de la armonía barroca, siendo una herramienta así de gran ayuda para los docentes de las enseñanzas de armonía.

- Por esta razón se ha desarrollado una herramienta que cubra estas necesidades. Esta aplicación comprueba el cumplimiento de las normas de la armonía barroca sobre un documento musical en formato MusicXML que se le proporciones. Para la comprobación de una de estas reglas (duplicación de sensible) es necesario abordar un problema no trivial: la identificación de acordes en una partitura. Para resolver este problema se ha desarrollado en este trabajo un sistema heurístico al que se ha denominado *matchmaking*.

- Además de esta funcionalidad de comprobación de armonía, se ha incluido en la aplicación la capacidad de generar casos de error que ejemplifiquen los incumplimientos de las normas de la armonía barroca. Estos ejemplos se generan a partir de un documento musical en formato MusicXML proporcionado a la aplicación.

- Al utilizar un formato estándar en el tratamiento de documentos musicales, la aplicación desarrollada en este trabajo es compatible con la mayoría de aplicaciones de informática musical existentes en el mercado. Aprovechando esta compatibilidad se sugiere el uso de aplicaciones complementarias de dos tipos: editores de música, para poder ver las partituras comprobadas o generadas; y sintetizadores, para poder escuchar dichas partituras.

- Se proponen una serie de posibles futuras ampliaciones para la aplicación que integran estas dos funcionalidades (visionado y síntesis) en la herramienta desarrollada, así como otras ampliaciones que mejoren tanto su funcionalidad como su capacidad musical.

- También se propone una ampliación que aborda la generación de voces adicionales (que cumplan las normas de la armonía barroca) a partir de una voz dada. Esta funcionalidad si está cubierta por otras herramientas existentes en la actualidad, pero se plantea abordar el problema desde otra perspectiva, buscando crear nuevas voces con interés propio en lugar de un simple relleno armónico.

6.4 Posibles ampliaciones

En esta sección se presentan las posibles ampliaciones que podrían ser introducidas en la aplicación en versiones posteriores, añadiendo nuevas funcionalidades y mejorando su accesibilidad.

6.4.1 Mejoras funcionales

Integración de visionado

Esta ampliación consiste en integrar el visionado de partituras en la propia aplicación, en lugar de utilizar aplicaciones externas para ello, como ocurre actualmente.

Para ello sería necesario un sistema que represente gráficamente la información contenida en los documentos MusicXML. Esto se puede llevar a cabo con el estándar de codificación de caracteres **Unicode**, que dispone en el rango 1D100 – 1D1FF de una colección de símbolos musicales (de la música occidental, incluida la notación cuadrada medieval. Para otras notaciones como la bizantina o la griega existen otros conjuntos de caracteres en otros rangos). A continuación se muestran algunos de los caracteres incluidos en este conjunto:

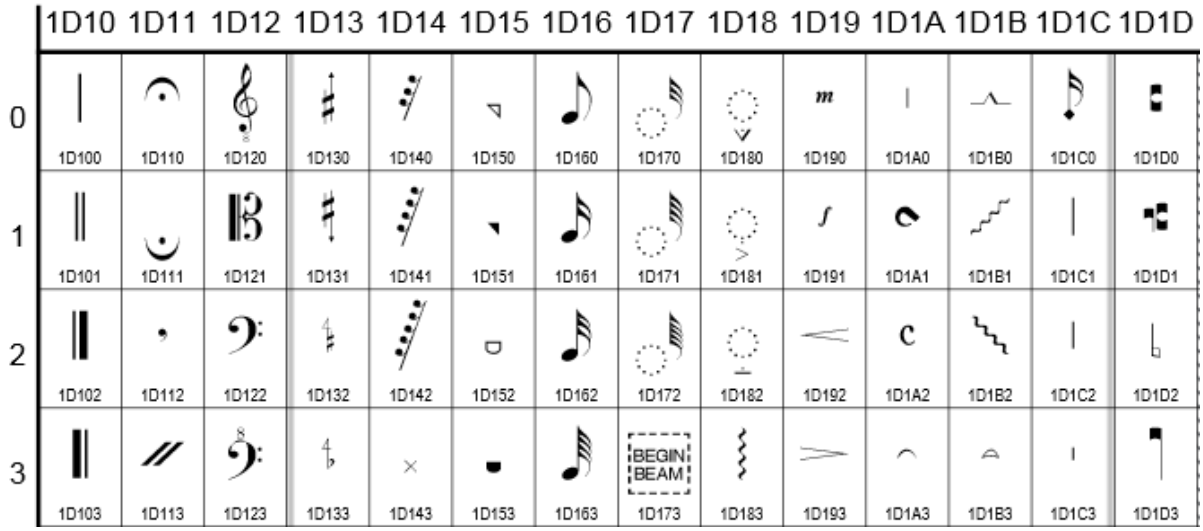


Figura 6.4-1 caracteres musicales Unicode

Integración de síntesis

Al igual que el visionado de los documentos como partituras, la escucha de los mismos también se realiza actualmente mediante aplicaciones externas. Esta funcionalidad puede igualmente incluirse en la propia aplicación.

La forma más sencilla de transferir la información de los documentos en MusicXML a un formato de audio (y por tanto la que utilizaría la aplicación) es mediante el formato MIDI. Esto es así gracias a que no se trata de un formato de audio propiamente dicho, sino que es de hecho más cercano a una representación de la partitura que el receptor interpreta para generar sonidos (ya sea un dispositivo o un programa). Esta comunicación se produce mediante instrucciones sobre las características y el tiempo de inicio de cada sonido.

Por tanto, el proceso incluiría la generación de un archivo MIDI a partir del documento MusicXML y su posterior reproducción en la aplicación, para lo que existen funciones propias de C# que se podrían aprovechar.

Adición de paquetes de idiomas

Actualmente la aplicación devuelve todos sus mensajes en español, pero esto podría ser mejorado haciendo que cargue un paquete de idioma a elección del usuario. Independizando los mensajes de la aplicación (por ejemplo en ficheros externos que cargue como *data*) podrían añadirse nuevos idiomas sin necesidad de modificar el código, e incluso permitir que los usuarios escriban sus propias traducciones proporcionándoles plantillas para estos ficheros.

En la versión de la aplicación del presente trabajo, los mensajes de la pestaña ayuda ya se cargan de un fichero externo en formato xml, presentando la idea inicial para esta posible ampliación.

Sin embargo, para que este sistema sea completamente funcional, será necesario que la aplicación cargue una lista con los posibles idiomas y permita al usuario que elija uno de ellos. Del mismo modo, será necesario que la aplicación cree un archivo que contenga las preferencias del usuario, para almacenar esta elección y evitar que cada vez que el usuario ejecute la aplicación tenga que elegir un paquete de idioma.

Migración a .NET

El sistema actual consta de una aplicación de escritorio. Al estar desarrollado en C# podría ser porteado a .NET Framework (.NET), convirtiéndolo en una aplicación web. De este modo se permitirá el acceso a través de un navegador web sin necesidad de disponer de la aplicación.

Para compatibilizar esta ampliación con las anteriores que abogan por la creación de reglas y paquetes de idioma por parte de los usuarios (ver subsecciones 0 y 0) sería necesario crear un repositorio online con los archivos de los usuarios y que ellos configuren un perfil online para cargar el contenido deseado desde ese repositorio.

6.4.2 Mejoras musicales

Creación de API de reglas

Esta ampliación consiste en desarrollar un formato con el que se puedan introducir nuevas reglas sin modificar la aplicación. Además de facilitar las actualizaciones, esto permitiría que los usuarios personalicen la aplicación con las normas que quieran.

Además, este formato puede ser preparado para que cualquier desarrollador pueda programar nuevas reglas, de modo que entre los propios usuarios se pueda crear un desarrollo e intercambio de reglas.

Para ello sería necesario el desarrollo de este formato, que se basaría en la interválica entre dos unidades de tiempo para dos voces. De este modo las reglas se pueden configurar como un conjunto de un máximo de 6 combinaciones de intervalos. A continuación se muestra como sería esta lista de

intervalos para algunas de las reglas de la armonía barroca (la nomenclatura utilizada para las notas es **V** voz y **N** nota, siendo V1 la voz inferior): segundas aumentadas (todos los posibles casos) y algunos ejemplos de quintas directas:

Distancia/regla	Segunda aumentada				Quinta directa	
V1N1 – V1N2	2ªAu	-2ªAu			2ªM	-4ªD
V2N1 – V2N2			2ªAu	-2ªAu	2ªm	-3ªm
V1N1 – V2N1						
V1N2 – V2N2					5ªJ	5ªJ
V1N1 – V2N2						
V1N2 – V2N1						

Tabla 6.4-1 Ejemplos de listas de intervalos para la API de reglas

Los campos que no tienen ningún valor específico pueden tener cualquier valor. Con este sistema se podrían codificar las reglas, aunque todos los casos tendrían que ser plasmados.

Con este formato, la aplicación comprobaría las distancias interválicas existentes con las presentes en todas las reglas que tenga introducidas y a partir de ahí generaría la lista de errores. La información específica del error, como los mensajes para el log, deberán ser introducidos por el desarrollador junto a la lista de intervalos.

6.4.3 Generación de voces para una melodía dada

Esta ampliación consiste en generar un documento musical de cuatro voces a partir de uno con una única voz proporcionado como entrada. Esta generación respeta las normas de la armonía barroca, por lo que se basa en la comprobación de las reglas, como ya hace la aplicación.

Para poder generar las voces adicionales es necesario conocer los grados de la escala por los que va pasando la melodía en cada momento. Esto sería posible gracias al sistema de *matchmaking* (ver subsección 4.2.4) ya desarrollado, al que habría que añadir un segundo sistema que con las distintas listas de posibles grados (para cada intervalo temporal) devolviera los enlaces entre estos acordes.

Esto consiste en elegir de la lista de posibles acordes el más probable, no en cuanto a la aparición de notas (como se utiliza el sistema de *matchmaking* en las funcionalidades actuales de la aplicación) si no en cuanto a las funciones armónicas, esto es tónica, subdominante y dominante. De este modo encontramos que, por ejemplo, teniendo las listas {III, V} y {I, VI}, el enlace V-I es más probable que el III-I o el III-VI.

Para hacer estos nuevos cálculos sería necesario el desarrollo de un nuevo sistema de emparejamientos, sólo que esta vez en lugar de emparejar notas emparejaría acordes.

Una vez conocidos los grados para cada unidad de tiempo de la melodía, la generación de voces más sencilla pasa por completar el resto de voces con las notas del acorde (respetando las normas de la armonía barroca en nuestra aplicación). Estos sonidos actúan como relleno y se mantienen en lo que dure el acorde. A continuación se muestra un ejemplo de asignación de grados para una melodía con su correspondiente relleno armónico en una segunda voz.

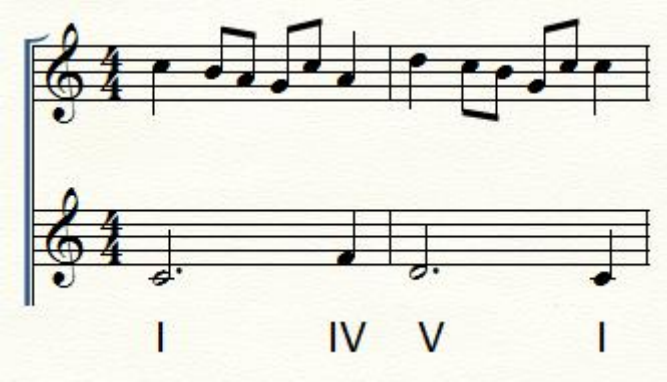


Figura 6.4-2 Análisis de acordes en una melodía

La melodía está en Do Mayor por lo que los grados utilizados constan de las siguientes notas:

- I {do, mi sol}
- IV {fa, la, do}
- V {sol, si, re}

Sin embargo, es posible añadir mayor complejidad a estas voces generadas haciendo que la duración de estos sonidos sea menor a la del acorde. Para ello se dispone de tres procedimientos:

- **Cambios de posición**: se trata del procedimiento más sencillo musicalmente, pero no así en cuanto a la lógica de la aplicación. Consiste en cambiar la nota del acorde que se está usando, por ejemplo para una blanca utilizar la tercera del acorde en la primera negra y la quinta en la segunda. La complejidad viene dada de que estos cambios de posición pueden generar nuevos errores, habitualmente quintas y octavas directas y/o paralelas. A continuación se muestra el ejemplo anterior con la introducción de un cambio de posición para el tercer pulso del primer acorde, donde se cambia la fundamental por la tercera:

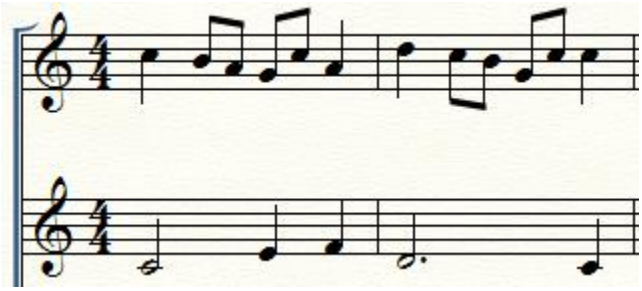


Figura 6.4-3 Ejemplo de cambio de posición

- **Adornos**: los adornos sirven para embellecer la melodía sin desvirtuar las funciones armónicas. Entre los adornos más habituales encontramos floreos y notas de paso:

    - Los floreos son batidas a la nota inmediatamente inferior o superior (y vuelta a la nota original). Aleatorizando "puntos de floreo" en la melodía (como se hace actualmente con los puntos de error) se puede dividir en tres secciones la duración de una nota y mantener la primera y la última en la nota original y florear la de en

medio. A continuación se muestra una variante del ejemplo inicial en la que se ha introducido un floreo en el primer acorde del segundo compás:

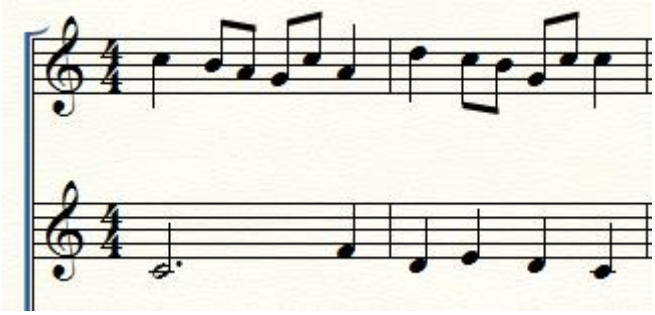


Figura 6.4-4 Ejemplo de floreo

- Las notas de paso consisten en utilizar las notas intermedias de un intervalo, aunque no pertenezcan a los acordes en uso. Se pueden utilizar dentro de un acorde con cambio de posición, de un modo similar a los floreos; o entre acordes, pasando por una o dos notas intermedias. A continuación se muestra una variante del ejemplo inicial en la que se han introducido unas notas de paso en el tercer pulso del primer acorde:

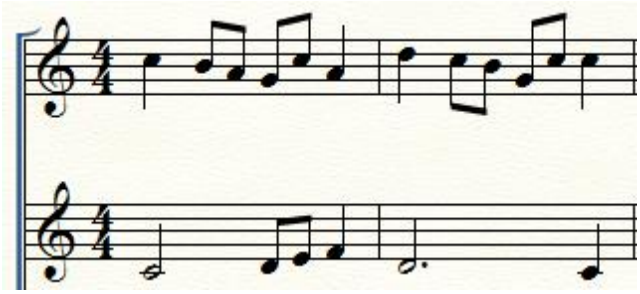


Figura 6.4-5 Ejemplo de nota de paso

- **Contrapunto**: aunque existen distintos procedimientos contrapuntísticos (aumentación, disminución, inversión, retrogradación, contrapunto imitativo y contrapunto reversible) el procedimiento por el que se optaría en la aplicación es el del contrapunto libre. Este procedimiento consiste en realizar imitaciones rítmicas entre las voces (a diferencia del resto que incluyen también imitaciones melódicas). Para introducir esta funcionalidad en la generación de voces habría que realizar un análisis estadístico de los grupos rítmicos más utilizados en la melodía original, y con estos datos introducir los grupos en el resto de voces, rellenándolos con repeticiones de la misma nota o con los procedimientos mencionados anteriormente (adornos y cambios de posición). A continuación se muestran dos ejemplos de contrapunto sobre el ejemplo inicial. En el primero se utiliza un contrapunto por inversión en el segundo compás, tomando el motivo del primer compás para la melodía y cambiando la dirección de los intervalos:

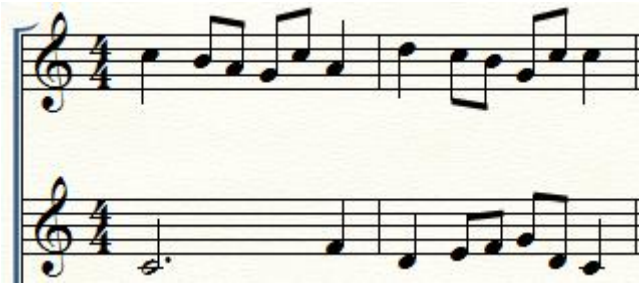


Figura 6.4-6 Ejemplo de contrapunto por inversión

Este ejemplo podría ser realizado por la aplicación pero como fruto aleatorio de combinación de cambios de posición, notas de paso y contrapunto libre. El segundo ejemplo se trata de una variación del ejemplo de cambios de posición en el que se aplica el patrón ritmo habitual en la melodía (negra, 4 corcheas, negra) sobre el primer compás de la segunda voz:

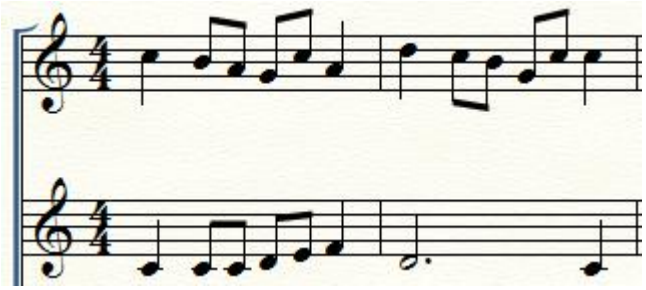


Figura 6.4-7 Ejemplo de contrapunto libre

Glosario

Acorde: conjunto de sonidos que suenan a la vez. Los acordes más comunes en lenguaje clásico son los acordes tríada, seguidos por los que incorporan la séptima y en menor medida, la novena. A partir del Romanticismo y en otros lenguajes como el jazz, la gama de acordes se amplía considerablemente.

Clave: signo gráfico que determina la altura de las notas, se escribe al principio de los pentagramas. La denominación de una clave viene dada por una nota y un número, hace referencia al nombre que tiene el sonido escrito en esa línea del pentagrama. Las claves son: fa en 3ª y 4ª, do en 1ª, 2ª, 3ª y 4ª y sol en 2ª.

Dominante: función armónica que crea inestabilidad. Habitualmente se designa dominante al V grado por ser el que desempeña esta función más a menudo, aunque también puede ser desempeñada por el VII grado.

Escala: sucesión de notas con unas distancias determinadas entre ellas. Las escalas abarcan los modos gregorianos, diatónicas (mayores y menores), pentatónicas, escalas de tonos enteros, modos de transposición limitada...

Especie (intervalo): en un intervalo, parte de su denominación que determina la distancia en tonos y semitonos entre las dos notas que conforman dicho intervalo. Las especies pueden ser: disminuido, menor, mayor, justo y aumentado. No todas las especies existen para todos los tipos.

Floreo: adorno entre dos notas reales con la misma altura que consiste en un giro hacia una nota contigua (normalmente la superior).

Función armónica: en una tonalidad, función que desempeña un acorde, pudiendo ser tónica si da sensación de reposo, contribuyendo a consolidar la tonalidad (I grado o III grado invertido); dominante si crea sensación de inestabilidad, buscando una tónica o cambiar a otra tonalidad (V y VII grado); o subdominante si sirve como unión entre una dominante y una tónica, proporcionando estabilidad tonal pero no sensación de reposo (II, IV y VI grado).

Fundamental: en un acorde sin invertir, la nota más grave.

Grado: en las escalas mayores y menores, cada una de las notas de la misma según la función que representen. Para designarlos numéricamente se utilizan números romanos. Los grados de la escala son: I tónica, II supertónica, III mediante, IV subdominante, V dominante, VI superdominante, VII subtónica o sensible.

Intervalo: un intervalo es la distancia entre dos notas. Pueden ser ascendentes, descendentes o unísonos (misma nota). Están compuestos de un número (distancia) y una letra (especie) que indica la distancia real en tonos entre los dos sonidos (tipo):

Especie/Tipo	2ª	3ª	4ª	5ª	6ª	7ª	8ª
D (disminuido)	0	1	2	3	3.5	4.5	5.5
m (menor)	0.5	1.5	-	-	4	5	-
M (mayor)	1	2	-	-	4.5	5.5	-
J (justo)	-	-	2.5	3.5	-	-	6

A/Au (aumentado)	1.5	2.5	3	4	5	6	6.5

Tabla G-6.4-1 Distancias de tonos y semitonos en los intervalos

Intervalo directo: dicho de un intervalo armónico (con las dos notas sonando a la vez) aquel a cuyas notas se llega por movimiento directo (a las dos de manera ascendente o a las dos de manera descendente). Los otros tipos de movimiento posibles son perpendicular (una voz asciende y otra desciende) y oblicuo (una voz se mantiene y la otra asciende o desciende).

Intervalo simplificado: en los intervalos mayores de la octava, intervalo del mismo tipo (D, m, M, J o Au) obtenido al bajar o subir por octavas uno de los sonidos hasta que quede en rango de la tabla anterior. Por ejemplo: 9ª=>2ª, 13ª=>6ª. En general se usan las versiones simplificadas de los acordes, las versiones completas se utilizan para formular acordes de manera menos disonante.

Modalidad, modo: en el contexto de escalas diatónicas, calificativo que se emplea para distinguir su distribución de tonos y semitonos: mayores o menores. En otros contextos puede referirse a los modos griegos, los modos gregorianos (escalas que se utilizaban durante la Edad Media) o a los modos de transposición limitada (escalas desarrolladas por el compositor Olivier Messiaen).

Música sin compasear: música en la que se ha prescindido de las líneas divisorias (que delimitan los compases), ya sea en toda la obra o en fragmentos determinados. Se suele emplear para dar mayor libertad al intérprete.

Nota de paso: adorno entre dos notas reales a una distancia de 3ª (y en casos excepcionales de 4ª) que consiste en la introducción de las notas entre ambas en una melodía. Por ejemplo, teniendo las notas reales mi y sol, se podría introducir una nota de paso, obteniendo mi-fa-sol, sin sacar las notas reales del contexto de su acorde.

Nota real: dicho de una nota, que pertenece al acorde sobre el que se construye el fragmento de melodía al que pertenece.

Octava: intervalo con una distancia de ocho notas. En el contexto de la altura, hace referencia al conjunto de notas (octava) en el que se encuentra un sonido, designadas numéricamente según la notación internacional de modo que la nota la con una frecuencia de 440Hz corresponde al la_4.

Referencia: en interpretación, sonido a partir del cual se busca otro, ya sea en afinación o en altura absoluta. Las referencias son necesarias cuando varios instrumentos o varias voces participan en la misma interpretación.

Resolver: dicho de un sonido, tener otro que cumpla unos requisitos preestablecidos a continuación. Por ejemplo “sensible resuelve en tónica” quiere decir que el séptimo grado siempre va seguido del primero. Las notas alteradas de manera accidental suelen resolver en un sonido próximo, normalmente a un semitono descendente o ascendente.

Sensible: séptimo grado de una escala cuando se encuentra a un semitono de la tónica (2ªm). Si se encuentra a una distancia de tono (2ªM) se denomina subtónica.

Subdominante: función armónica de transición entre dominante y tónica. Actúa como un elemento con características de las otras funciones: da estabilidad tonal (no busca cambiar de tonalidad) pero no da sensación de reposo, siempre busca resolver en otro acorde.

Tipo (intervalo): en un intervalo, parte numérica de su designación que designa la distancia entre las dos notas que lo componen sin tener en cuenta tonos y semitonos. De este modo, dos notas correlativas estarán a una distancia de 2ª, dos notas con otro sonido en medio (por ejemplo mi y do) estarán a una distancia de 3ª…

Tonalidad: aplicado a las escalas mayores y menores, identificación de las mismas mediante su tónica y su modo.

Tónica: en una tonalidad, primer grado, nota que da nombre a la escala. Por ejemplo "do" en Do Mayor.

Tríada: acorde compuesto por tres notas: la más grave se denomina fundamental y las otras dos están a distancia de tercera y quinta respectivamente. Dependiendo de las especie de estos intervalos pueden ser:

Tipo/Intervalos	3ª	5ª
Mayores	M	J
Menores	m	J
Aumentados	M	Au
Disminuidos	m	D

Tabla G-2 Tipos de acordes tríada

Tritono: intervalo con una distancia de 3 tonos. Puede tratarse de una 4ª aumentada o de una 5ª disminuida. Es un intervalo con una sonoridad muy tensa, por lo que desde la Edad Media fue designado por la Iglesia como algo maligno, intentando evitar su uso.

Voz contigua: tomando la distribución de voces soprano, contralto, tenor y bajo, voces contiguas son aquellas que están seguidas en esta lista. Por ejemplo: soprano-contralto o contralto-tenor.

Voz extrema: las voces extremas son la más aguda y la más grave, en la distribución soprano-contralto-tenor-bajo son la soprano y el bajo.

Referencias

Piston, Walter. *Armonía (Principles of Harmonic Analysis).* Mundimúsica, 2009.

de la Motte, Diether. *Armonía (Harmonielehre)*. Mundimúsica, 2008.

Schoenberg, Arnold. *Tratado de Armonía (Harmonielehre)*. Real Musical.

Hindemith, Paul. *Armonía tradicional: primera y segunda parte (A Concentrated Course In Traditional Harmony Book)*. Ricordi Americana, 2009.

Zamacois, Joaquin. *Tratado de Armonía*. Idea Books, 2002.

Pajares Alonso, Roberto L. *Historia de la música en 6 bloques. Bloque 1. Músicos y contexto*. Vision Net, 2011.

The Unicode Consortium, *Musical Symbols* http://www.unicode.org/charts/PDF/U1D100.pdf

Cheng, Heng-Tze; Yang, Yi-Hsuan; Lin, Yu-Ching; Liao, I-Bin; Chen, Homer H. (National Taiwan University), *Automatic Chord Recognition for Music Classification and Retrieval* http://users.ece.cmu.edu/~hengtzec/papers/icme08_chord.pdf

Music in MusicXML Format, http://www.musicxml.com/music-in-musicxml/

MusicXML Software, http://www.musicxml.com/software/

HarmonyWiz, http://www.wizdommusic.com/products/harmonywiz.html

Capella Software, http://www.capella.de/us/index.cfm/products/tonica-fugata/info-tonica-fugata/

Finale, http://www.finalemusic.com/products/finale/

MuseScore, https://musescore.org/

Cubase, http://www.steinberg.net/en/products/cubase

JSBChorales.net (repositorio de corales de J. S. Bach), http://www.jsbchorales.net/sets.shtml

J. S. Bach "*Ach Gott, vom Himmel sieh darein*", BWV 2, http://en.wikipedia.org/wiki/Ach_Gott,_vom_Himmel_sieh_darein,_BWV_2

G. Mahler "*Lieder eines fahrenden Gesellen*", http://en.wikipedia.org/wiki/Lieder_eines_fahrenden_Gesellen

Microsoft C# Reference, https://msdn.microsoft.com/en-us/library/618ayhy6.aspx

Anexo 1: Estructura del código

A continuación se presenta la estructura del código. La aplicación ha sido desarrollada en C#. En esta estructura se distinguen tres tipos de elementos distintos:

- Carpetas: contienen clases, funcionan de manera similar a los paquetes en otros lenguajes. Las clases contenidas en una carpeta tienen acceso al resto de clases de la misma carpeta (salvo que su visibilidad sea privada). Para acceder desde un archivo a funciones en archivos de otras carpetas se deberá hacer una importación utilizando *using*.
  En el diseño de la aplicación se han utilizado las carpetas para modularizar la aplicación, utilizándose en esta memoria el término módulo como equivalente a carpeta.

- Archivos de clases: son los archivos de código. Cada archivo puede tener una o más clases. Estas clases se engloban dentro de un *namespace* (que comparte nombre con el archivo). Cada clase tiene sus propias funciones.

- Formularios: contienen la información sobre la interfaz gráfica. Están compuestos de varios archivos, de los cuales los principales son el archivo *designer,* que hace las funciones de vista; y el propio archivo *form*, que actúa como controlador, uniendo los eventos del *designer* con la lógica del programa. Partiendo de esta estructura propia de C# se ha seguido un patrón modelo-vista-controlador.

A continuación se expone una relación de las carpetas y archivos de clases:

- **Check**: módulo encargado de gestionar todas las actividades relacionadas con la comprobación de las reglas.
  - Checker.cs

- **Generate**: módulo encargado de gestionar las actividades de generación de casos de prueba.
  - Generator.cs

- **MusicCode**: módulo que contiene los tipos de datos de la aplicación y las funciones que afectan a los mismos.
  - Chord.cs
  - Measure.cs
  - MusicFunctions.cs
  - Note.cs
  - Scale.cs
  - Voice.cs

- **Ruleset**: módulo de reglas, tanto para comprobación (*Ruleset*) como para generación (*ErrRules*).
  - ErrRules.cs
  - Rules.cs

- **XMLio**: módulo que gestiona la entrada y salida de texto de la aplicación.
    - LogWriter.cs
    - XMLRead.cs
    - XMLWrite.cs

- **Form**: interfaz gráfica de la aplicación.

Anexo 2: Ejemplo de *matchmaking*

A continuación se presenta un ejemplo del sistema heurístico *matchmaking* con un valor umbral de 10 y haciendo los cálculos para la tonalidad de Do Mayor (por la información obtenida del fichero MusicXML sabemos que por su armadura el documento estará en Do Mayor o La Menor) sobre la siguiente partitura a cuatro voces:

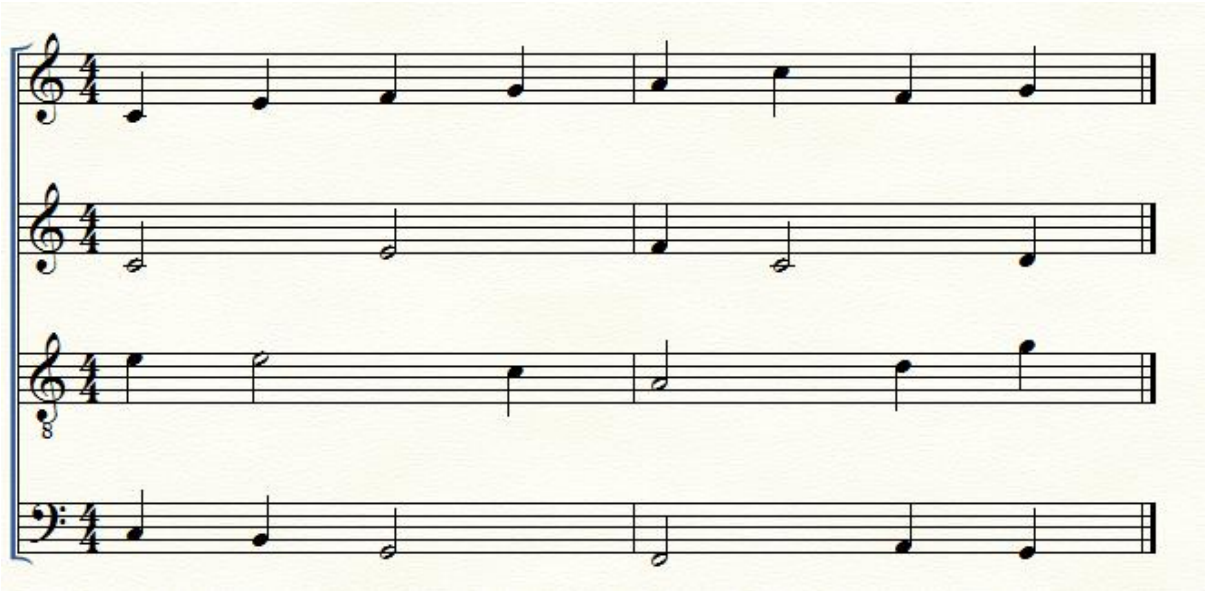


Figura A2-1 Partitura a cuatro voces

Se recuerda que los acordes están asociados a una pareja grado-escala. Dado que la escala es una información disponible en el archivo, el análisis se centra en los grados, pues una vez fijada la escala, los acordes posibles quedan asociados a los distintos grados. En primer lugar el sistema toma todas las notas en la primera unidad de tiempo (en este caso las unidades de tiempo son negras, ya que es la figura más breve del documento). Con las notas de esa unidad de tiempo, se calcula el valor heurístico para todos los grados de la escala:

- Si el sonido es la fundamental del acorde: se suma 5 al grado asociado ahora a dicho acorde.
- Si el sonido es la tercera o la quinta del acorde: se suma 3 al grado asociado ahora al acorde.
- Si el sonido es la séptima del acorde: se suma 1 al grado asociado ahora a dicho acorde.

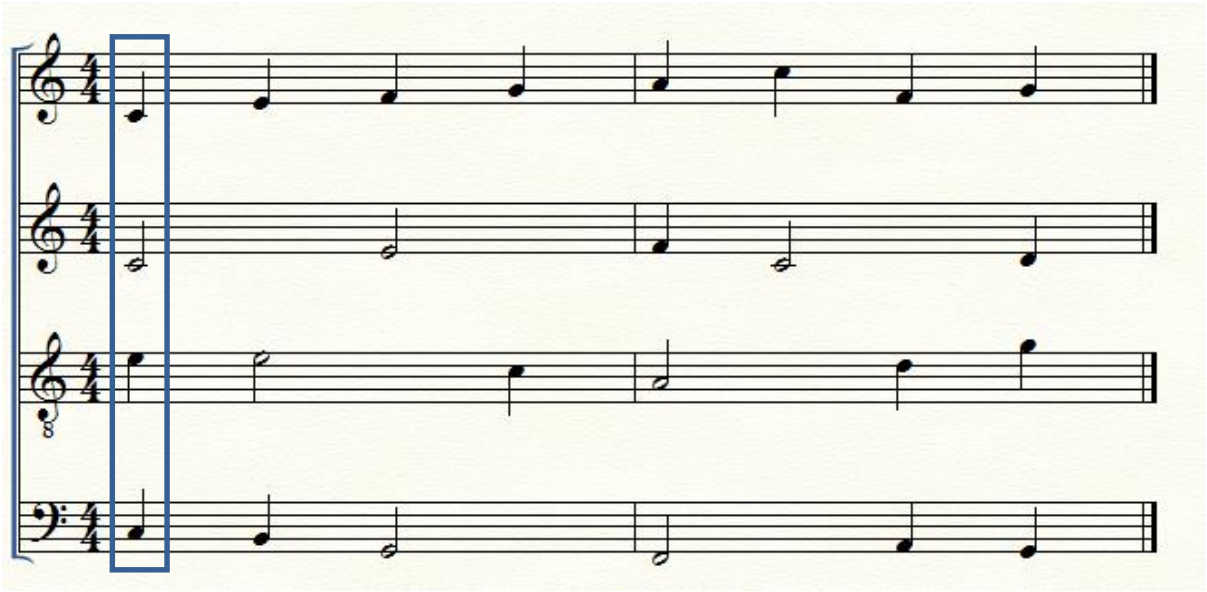


Figura6.4-2 Ejemplo de *matchmaking*: primera iteración

Grado	Soprano	Contralto	Tenor	Bajo	Total
I	5	5	3	5	18
II	1	1		1	3
III			5		5
IV	3	3	1	3	10
V					0
VI	3	3	3	3	12
VII					0

Tabla A2-1 Valores heurísticos por grados: primera iteración

Para los grados que superen el valor umbral (I, IV y VI) se intenta añadir la siguiente unidad de tiempo, calculando su valor individual y el valor medio entre la primera y la segunda unidad de tiempo. Este valor medio será el que se compare con el umbral para confirmar o desmentir la posible asignación a un acorde.

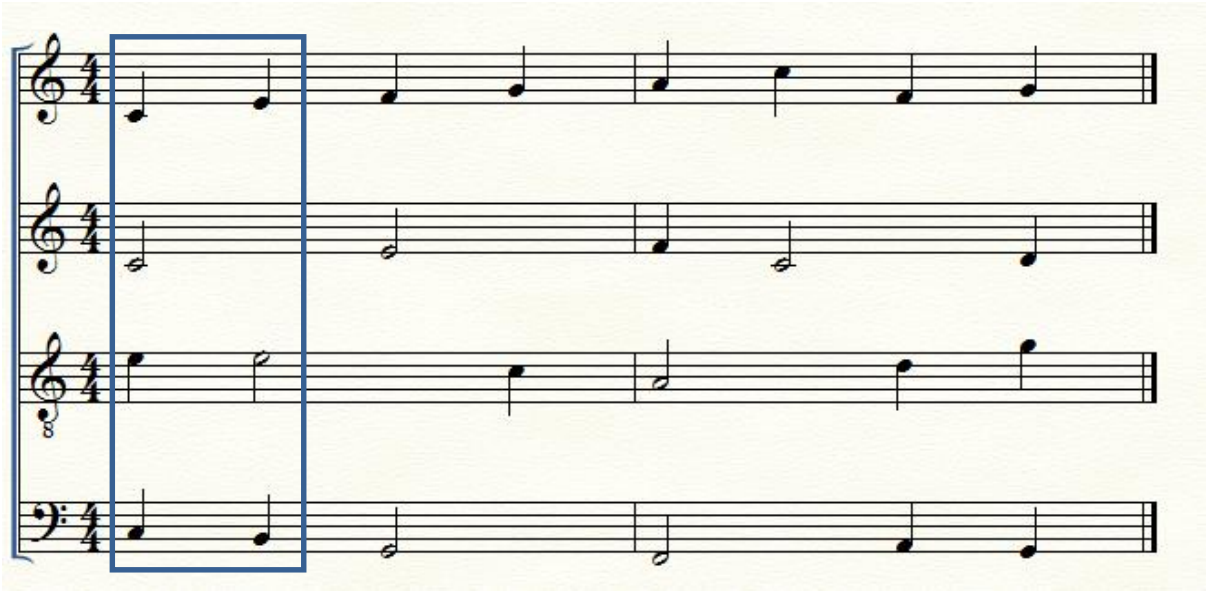


Figura A2-3 Ejemplo de *matchmaking*: segunda iteración

Grado	Soprano	Contralto	Tenor	Bajo	Total	Media
I	3	5	3	5	16	17
IV	1	3	1	3	8	9
VI	3	3	3	3	12	12

Tabla A2-2 Valores heurísticos por grados: segunda iteración

Una vez eliminado el IV grado por no superar el umbral, se intenta añadir la siguiente unidad de tiempo. La media se calcula mediante divisiones enteras, por ejemplo para el I grado y sobre los dos primeros pasos:

$$((18+16)/2 =17$$

Para este mismo grado y en el paso siguiente se tendría

$$(18+16+9)/3=14$$

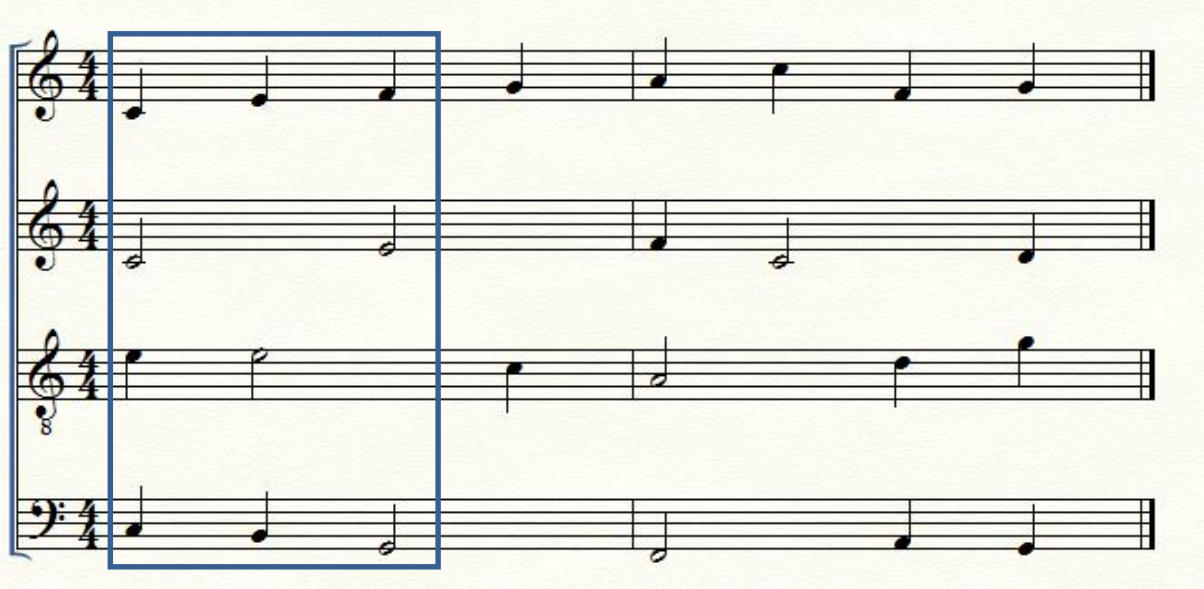


Figura A2-4 Ejemplo de *matchmaking*: tercera iteración

Grado	Soprano	Contralto	Tenor	Bajo	Total	Media
I		3	3	3	9	14
VI		3	3	1	7	10

Tabla A2-3 Valores heurísticos por grados: tercera iteración

Para ambos grados se supera el valor umbral, por lo que se intenta añadir la siguiente unidad de tiempo.

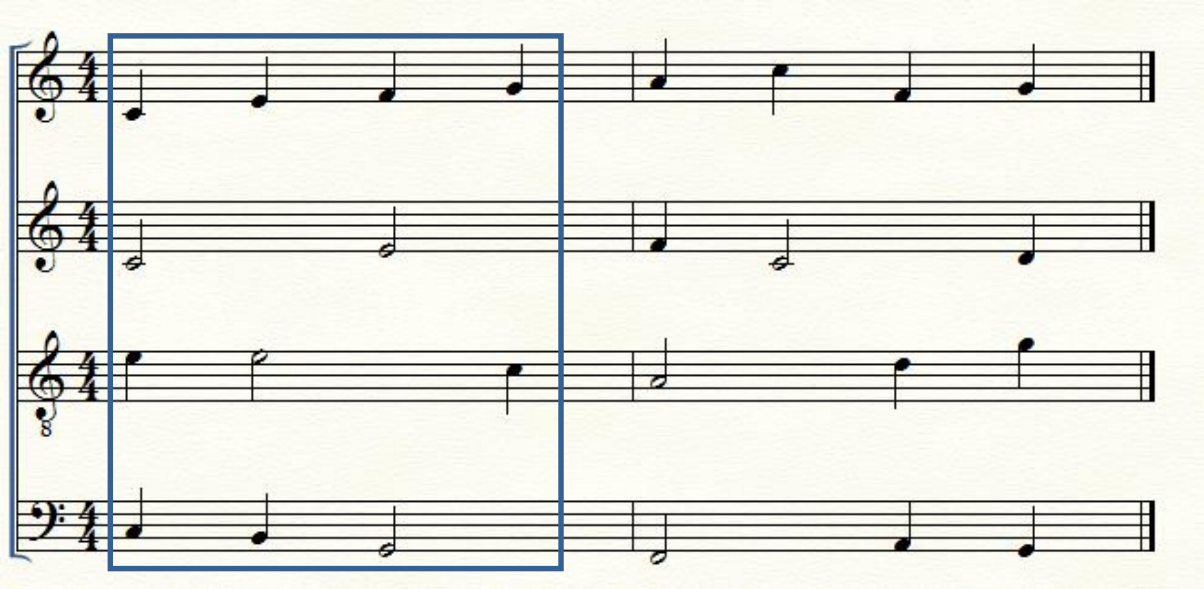


Figura A2-5 Ejemplo de *matchmaking*: cuarta iteración

Grado	Soprano	Contralto	Tenor	Bajo	Total	Media
I	3	3	5	5	16	14
VI	1	3	3	1	8	9

Tabla A2-4 Valores heurísticos por grados: cuarta iteración

Sólo el I grado supera el umbral, pero al intentar añadir la siguiente unidad de tiempo, su peso en este acorde es 0, por lo que no se intenta continuar (se trata de un documento a varias voces) y se devuelven los elementos de la lista anterior que superen el umbral ordenados de mayor a menor peso: {I}.

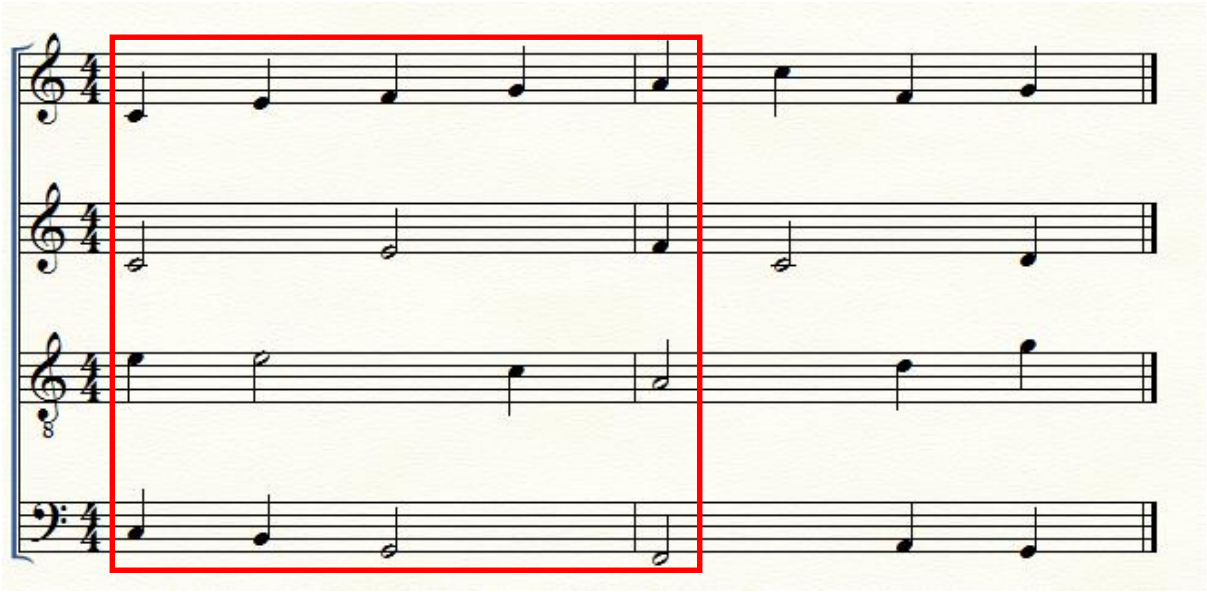


Figura A2-6 Ejemplo de *matchmaking*: quinta iteración con valores 0

Grado	Soprano	Contralto	Tenor	Bajo	Total
I					0

Tabla A2-5 Valores heurísticos por grados: quinta iteración con valores 0

Al haber cerrado una lista de acordes, se vuelve a empezar desde el primer paso con esta nueva unidad de tiempo: se calcula el valor heurístico de la misma para todos los grados de la escala:

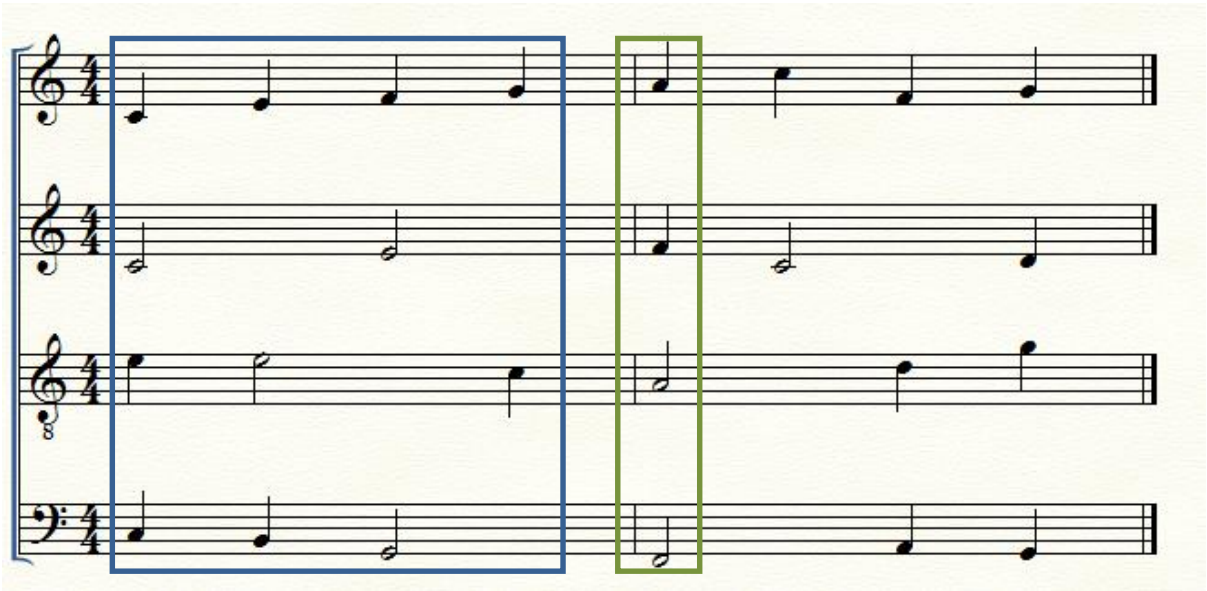


Figura A2-7 Ejemplo de *matchmaking*: quinta iteración con valores válidos

Grado	Soprano	Contralto	Tenor	Bajo	Total

I					0
II	3	3	3	3	12
III					0
IV	3	5	3	5	16
V		1		1	2
VI	5		5		10
VII	1	3	1	3	8

Tabla A2-6 Valores heurísticos por grados: quinta iteración con valores válidos

Los grados que superan el valor umbral son II, IV y VI. Para estos grados se intenta añadir la siguiente unidad temporal:

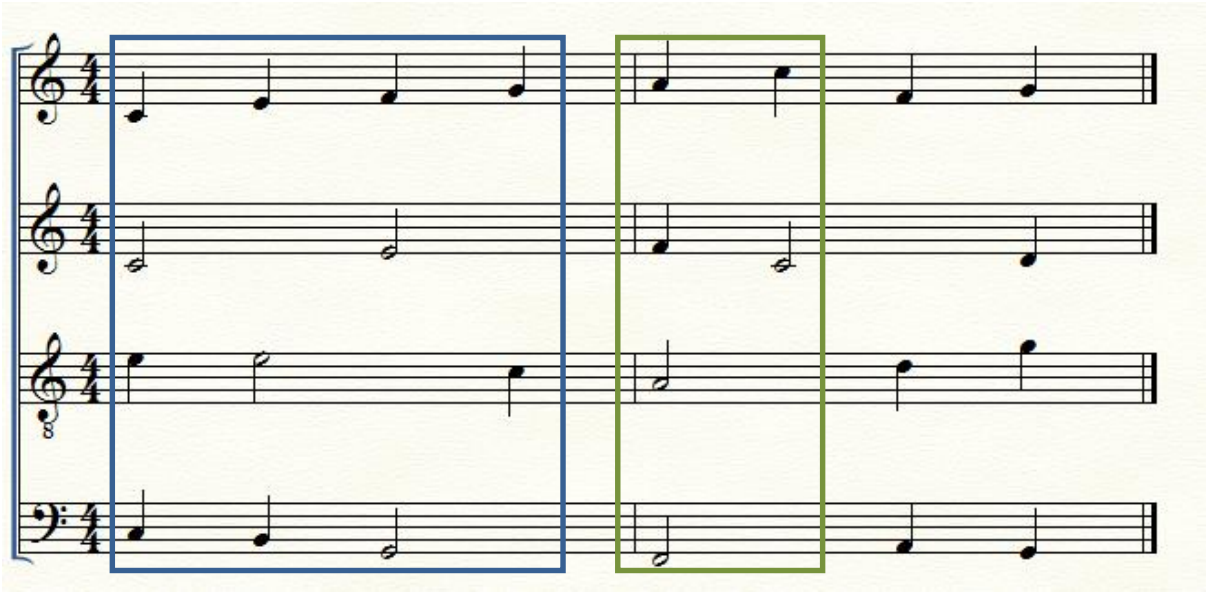


Figura A2-8 Ejemplo de *matchmaking*: sexta iteración

Grado	Soprano	Contralto	Tenor	Bajo	Total	Media
II	1	1	3	3	8	10
IV	3	3	3	5	14	15
VI	3	3	5		11	10

Tabla A2-7 Valores heurísticos por grados: sexta iteración

Dado que los tres grados siguen superando el valor, se continúa intentando ampliar el intervalo temporal para los tres:

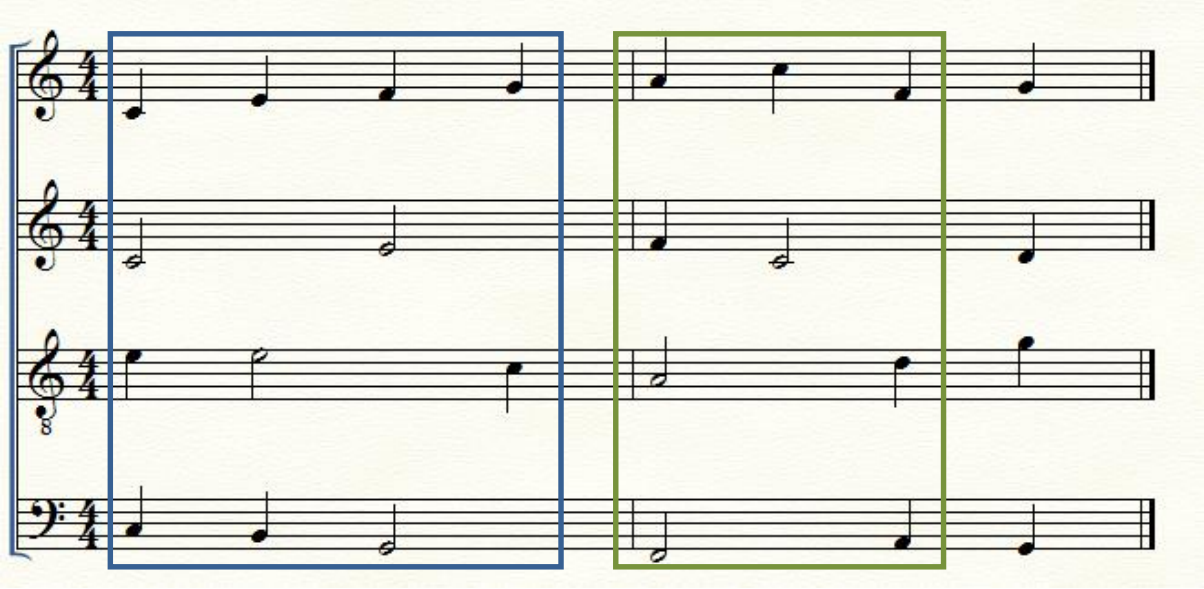


Figura A2-9 Ejemplo de *matchmaking*: séptima iteración

Grado	Soprano	Contralto	Tenor	Bajo	Total	Media
II	3	1	5	3	12	10
IV	5	3		5	13	14
VI		3			3	7

Tabla A2-8 Valores heurísticos por grados: séptima iteración

El VI grado ya no supera el valor umbral, pero al intentar añadir la siguiente unidad temporal el II grado tampoco supera el umbral y para el IV grado todo son notas extrañas, por lo que se devuelven los valores que superan el umbral de la lista anterior como posibles acordes para este intervalo temporal: {IV, II}.

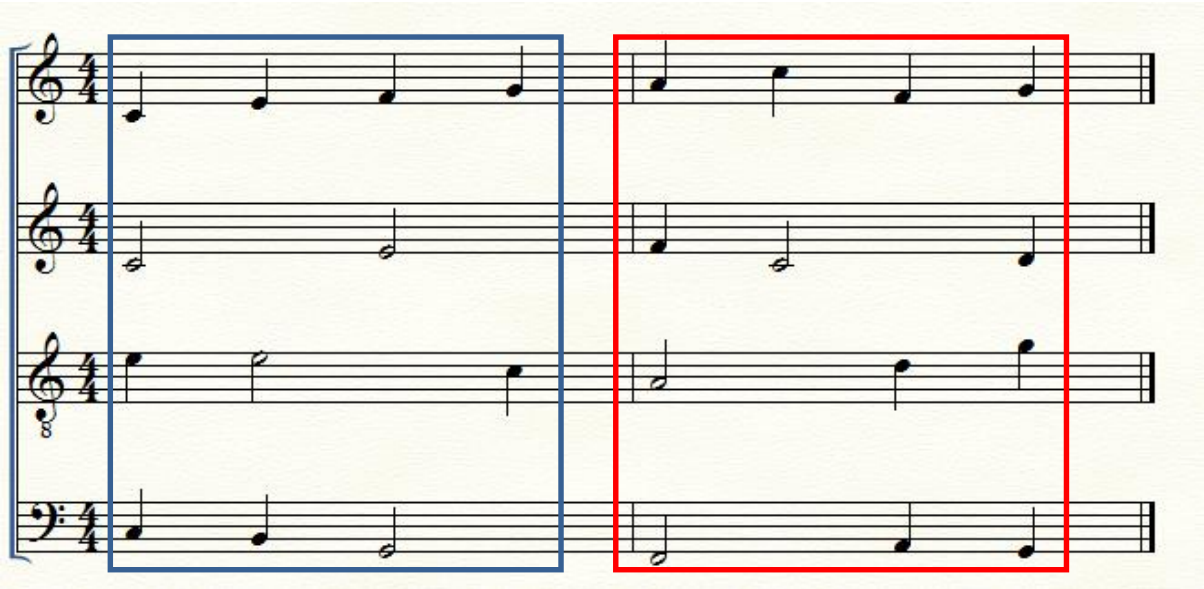


Figura A2-10 Ejemplo de *matchmaking*: octava iteración con valores 0

Grado	Soprano	Contralto	Tenor	Bajo	Total	Media
II		5			5	8

IV		0	-

Tabla A2-9 Valores heurísticos por grados: octava iteración con valores 0

La unidad temporal restante, es a la vez última y primera del intervalo (ya que el documento acaba y no puede ampliarse el intervalo), por lo que se calculan los valores heurísticos para todos los grados de la escala y se devuelve una lista con aquellos que superan el valor umbral ordenados de mayor a menor valor heurístico: {V, III}.

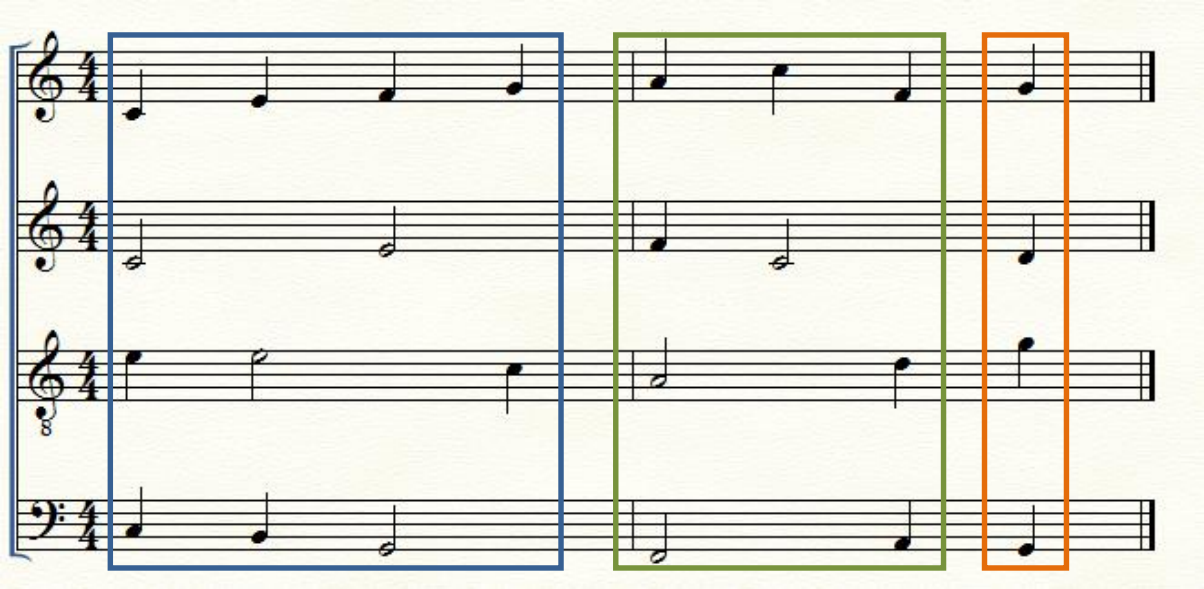


Figura A2-11 Ejemplo de *matchmaking*: octava iteración con valores válidos

Grado	Soprano	Contralto	Tenor	Bajo	Total
I	3		1	3	7
II		5			5
III	3	1	3	3	10
IV					
V	5	3	3	5	16
VI	1			1	2
VII		3	5		8

Tabla A2-10 Valores heurísticos por grados: octava iteración con valores válidos

Por tanto, utilizando el sistema de *matchmaking* se obtiene que este documento está dividido en tres secciones armónicas:

- La primera, azul, será uno de los acordes de la siguiente lista (ordenados de mayor a menor probabilidad): {I}
- La segunda, verde, será uno de los acordes de la siguiente lista (ordenados de mayor a menor probabilidad): {IV, II}
- La tercera, naranja, será uno de los acordes de la siguiente lista (ordenados de mayor a menor probabilidad): {V, III}

La lista de grados más probables por tanto será {I IV V} para Do Mayor. Para La Menor sería {VI II III}. Al ser los grados de más probable aparición I, IV y V el sistema deduce que la tonalidad es Do Mayor, y por tanto la sensible es "si".