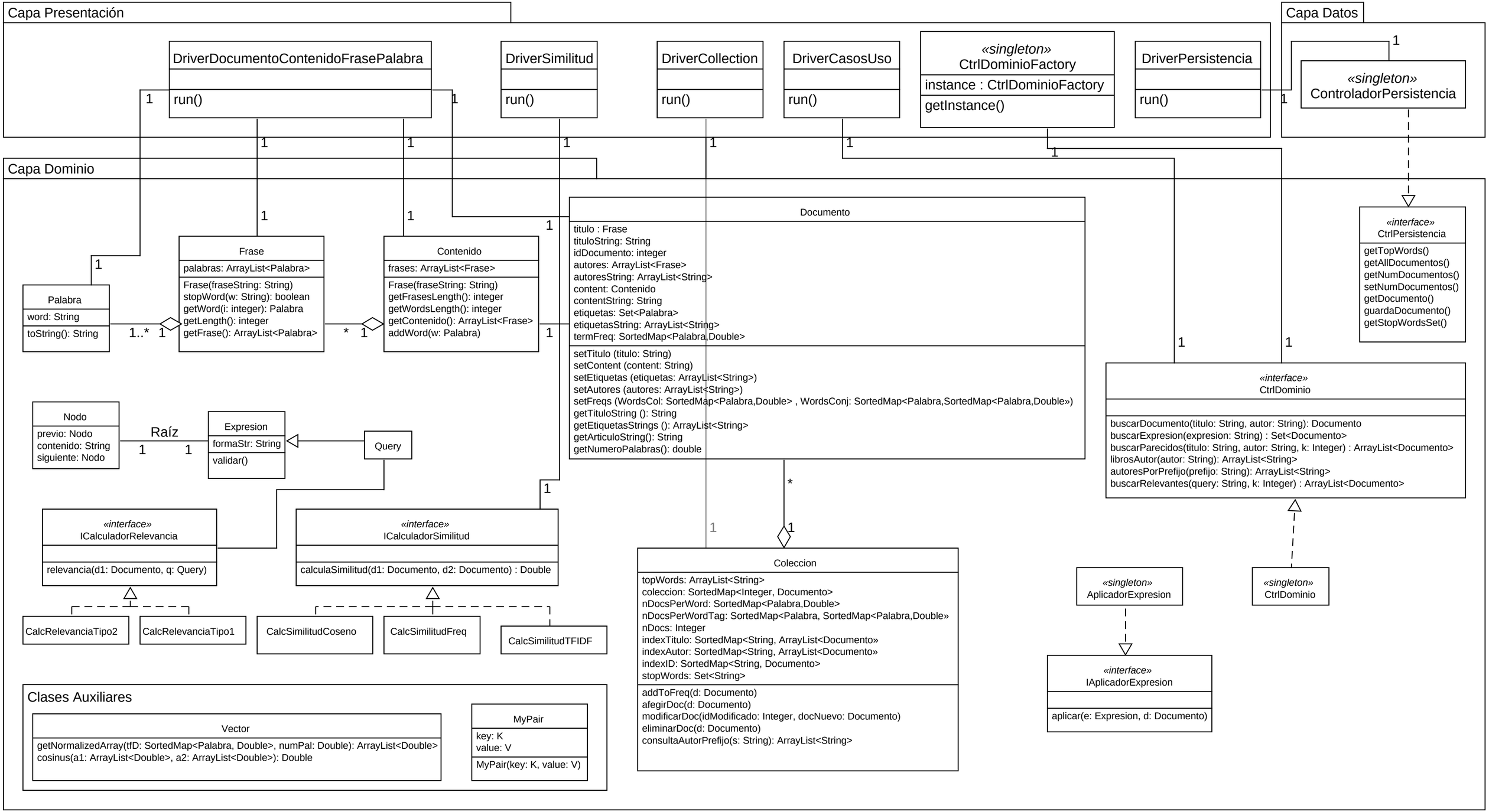
Capa Presentación
DriverDocumentoContenidoFrasePalabra
run()
DriverSimilitud
run()
DriverCollection
run()
DriverCasosUso
run()
«singleton»
CtrlDominioFactory
instance : CtrlDominioFactory
getInstance()
DriverPersistencia
run()
Capa Datos
«singleton»
ControladorPersistencia
Capa Dominio
Palabra
word: String
toString(): String
Frase
palabras: ArrayList<Palabra>
Frase(fraseString: String)
stopWord(w: String): boolean
getWord(i: integer): Palabra
getLength(): integer
getFrase(): ArrayList<Palabra>
Contenido
frases: ArrayList<Frase>
Frase(fraseString: String)
getFrasesLength(): integer
getWordsLength(): integer
getContenido(): ArrayList<Frase>
addWord(w: Palabra)
Documento
titulo : Frase
tituloString: String
idDocumento: integer
autores: ArrayList<Frase>
autoresString: ArrayList<String>
content: Contenido
contentString: String
etiquetas: Set<Palabra>
etiquetasString: ArrayList<String>
termFreq: SortedMap<Palabra,Double>
setTitulo (titulo: String)
setContent (content: String)
setEtiquetas (etiquetas: ArrayList<String>)
setAutores (autores: ArrayList<String>)
setFreqs (WordsCol: SortedMap<Palabra,Double> , WordsConj: SortedMap<Palabra,SortedMap<Palabra,Double»)
getTituloString (): String
getEtiquetasStrings (): ArrayList<String>
getArticuloString(): String
getNumeroPalabras(): double
«interface»
CtrlPersistencia
getTopWords()
getAllDocumentos()
getNumDocumentos()
setNumDocumentos()
getDocumento()
guardaDocumento()
getStopWordsSet()
«interface»
CtrlDominio
buscarDocumento(titulo: String, autor: String): Documento
buscarExpresion(expresion: String) : Set<Documento>
buscarParecidos(titulo: String, autor: String, k: Integer) : ArrayList<Documento>
librosAutor(autor: String): ArrayList<String>
autoresPorPrefijo(prefijo: String): ArrayList<String>
buscarRelevantes(query: String, k: Integer) : ArrayList<Documento>
Nodo
previo: Nodo
contenido: String
siguiente: Nodo
Raíz
Expresion
formaStr: String
validar()
Query
«interface»
ICalculadorRelevancia
relevancia(d1: Documento, q: Query)
«interface»
ICalculadorSimilitud
calculaSimilitud(d1: Documento, d2: Documento) : Double
CalcRelevanciaTipo2
CalcRelevanciaTipo1
CalcSimilitudCoseno
CalcSimilitudFreq
CalcSimilitudTFIDF
Coleccion
topWords: ArrayList<String>
coleccion: SortedMap<Integer, Documento>
nDocsPerWord: SortedMap<Palabra,Double>
nDocsPerWordTag: SortedMap<Palabra, SortedMap<Palabra,Double»
nDocs: Integer
indexTitulo: SortedMap<String, ArrayList<Documento»
indexAutor: SortedMap<String, ArrayList<Documento»
indexID: SortedMap<String, Documento>
stopWords: Set<String>
addToFreq(d: Documento)
afegirDoc(d: Documento)
modificarDoc(idModificado: Integer, docNuevo: Documento)
eliminarDoc(d: Documento)
consultaAutorPrefijo(s: String): ArrayList<String>
«singleton»
AplicadorExpresion
«interface»
IAplicadorExpresion
aplicar(e: Expresion, d: Documento)
«singleton»
CtrlDominio
Clases Auxiliares
Vector
getNormalizedArray(tfD: SortedMap<Palabra, Double>, numPal: Double): ArrayList<Double>
cosinus(a1: ArrayList<Double>, a2: ArrayList<Double>): Double
MyPair
key: K
value: V
MyPair(key: K, value: V)