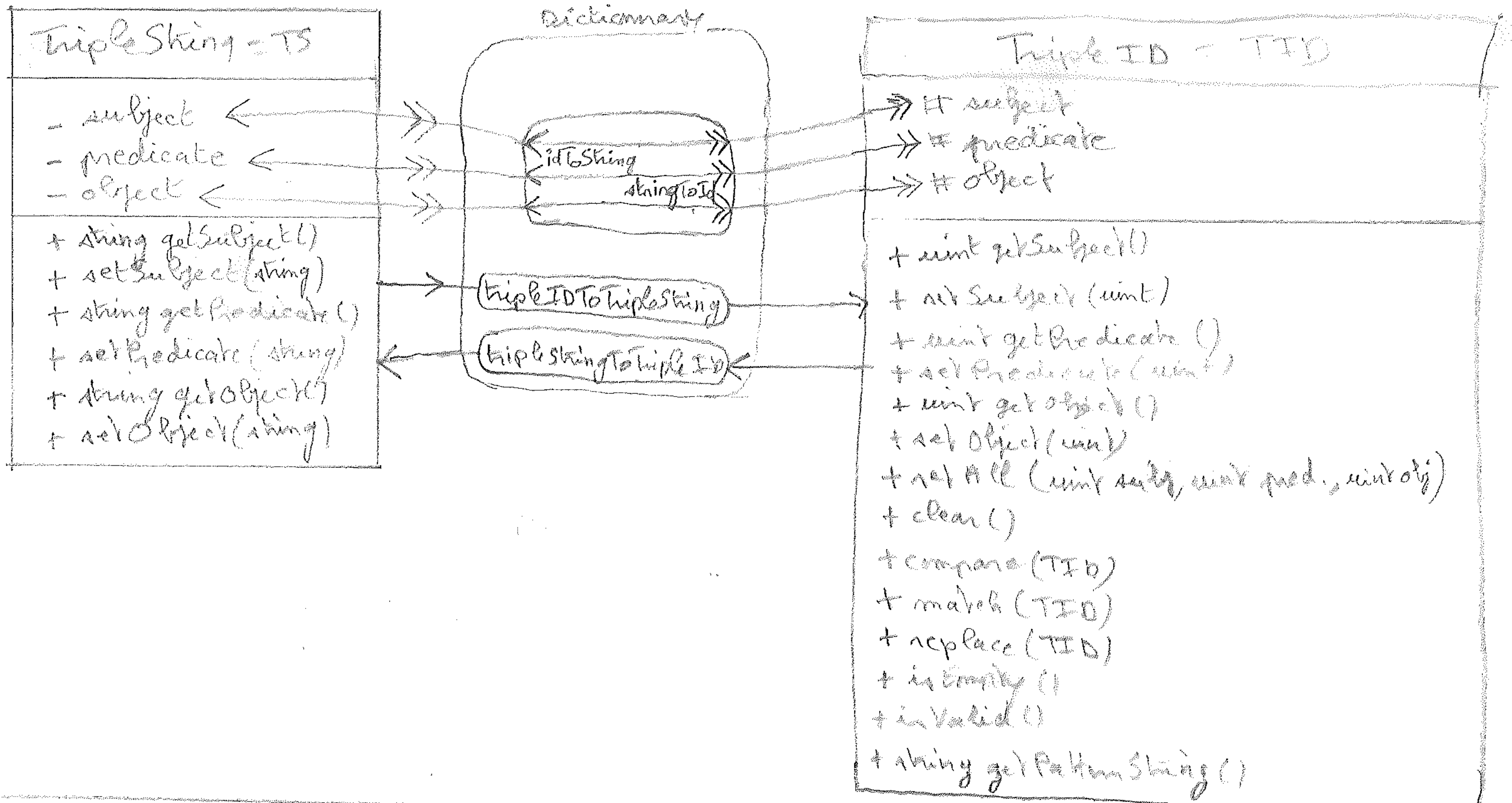
```
string URI = "http://dbpedia.org/resource/Barack_Obama"
HDT * hdt = HDTManager::mapHDT("fichier.rdf");                          main.cpp
ItTS * it = hdt->search( URI.c_str(), "", "");

    TS ts(URI.c_str(),"","")
    TID tid
    dictionary->tripleStringToTripleID(ts, tid);
    ItTID * iterID = htriples->search(tid)  //htriples = hdt->triples    BasicHDT::search

        swapComponentOrder(&tid, SPO, htriples->getOrder())             BitmapTriples::search
        string patternString = tid.getPatternString():

            patternString = "S??" // as only subject exist in ts        TripleID::getPatternString

        iterID = new BitmapTriplesSearchIterator(htriples, tid)

            iterID-> | triples = htriples
                     | pattern = tid
                     | adjY. | elements = htriples->arrayY               BitmapTSearchIt
                     |       | bitmap  = htriples->bitmapY               ::BitmapTSearchIt
                     |
                     | adjZ. | element = htriples->arrayZ
                     |       | bitmap  = htriples->bitmapZ

            swapComponentOrder(&ipattern, SPO, itriples->order) // ipattern  iterID.pattern
            iterID-> | patX = ipattern.getSubject()               // itriples = iterID.triples
                     | patY = ipattern.getPredicate()
                     | patZ = ipattern.getObject()
                     | findRange()
                     | goToStart()

    it = new TIDSIt(hdictionary, iterID)

        it -> | dict = hdictionary                                       TIDSIt::TIDSIt
              | iterator = iterID

while (it->hasNext())
{
  TS* tr = it->next();
  cout << tr.getSubject() << "  " << tr.getPredicate() << "  " << tr.getObject() << endl;
}
```



TripleString = TS

- subject
- predicate
- object

+ string getSubject()
+ setSubject(string)
+ string getPredicate()
+ setPredicate(string)
+ string getObject()
+ setObject(string)

TripleID = TID

subject
predicate
object

+ uint getSubject()
+ setSubject(uint)
+ uint getPredicate()
+ setPredicate(uint)
+ uint getObject()
+ setObject(uint)
+ setAll(uint subj, uint pred, uint obj)
+ clear()
+ compare(TID)
+ match(TID)
+ replace(TID)
+ isEmpty()
+ isValid()
+ string getPatternString()

enum TripleComponentRole = TCompRole

SUBJECT
PREDICATE
OBJECT

enum TripleComponentOrder = TCompOrder

Unknown
SPO
SOP
PSO
POS
OSP
OPS

enum DictionarySection = DicSec

SHARED_SUBJECT
NOT_SHARED_SUBJECT
SHARED_OBJECT
NOT_SHARED_OBJECT
NOT_SHARED_PREDICATE

enum Mapping

MAPPING1
MAPPING2

enum ResultEstimationType = ResEstType

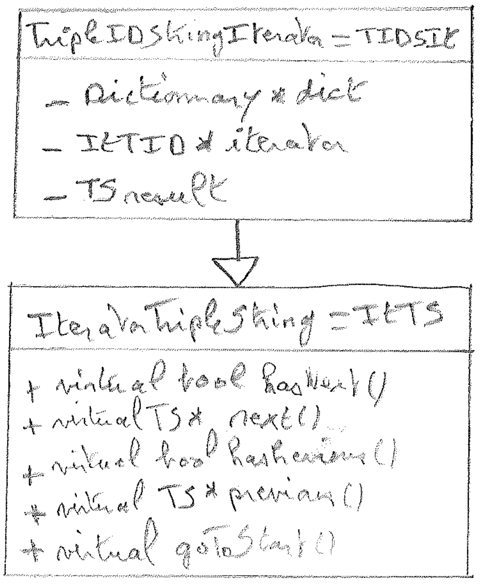
UNKNOWN
APPROXIMATE
UP_TO
EXACT

enum RDFNotation

XML
NTRIPLES
TURTLES
N3
NQUAD
JSON

enum HDTCost

CONSTANT
LOG_S
LOG_P
LOG_O
SEQUENTIAL_S
SEQUENTIAL_P
SEQUENTIAL_O
SEQUENTIAL_ALL

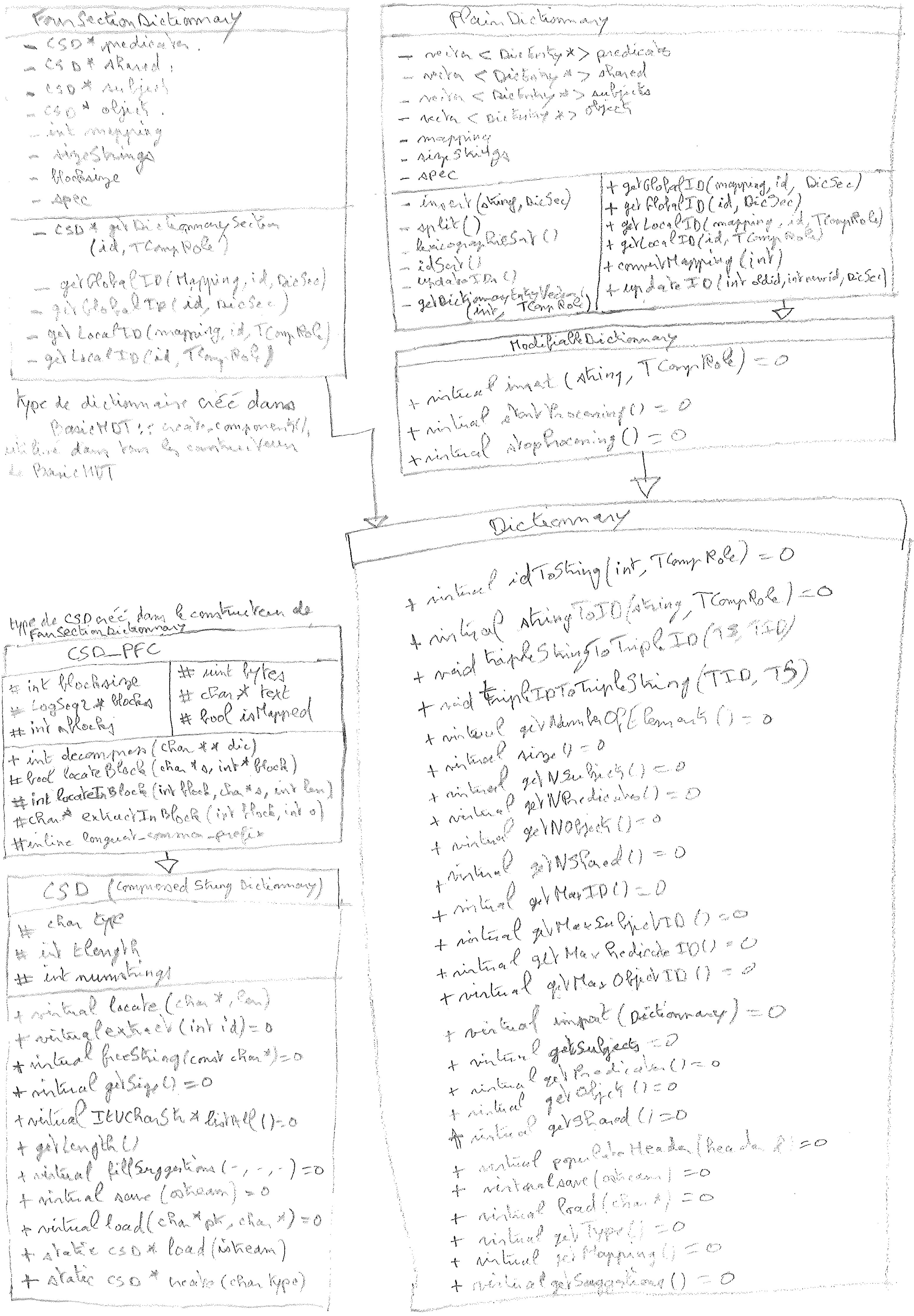


TripleIDStringIterator = TIDSIt

- Dictionary * dict
- ItTID * iterator
- TS result

IteratorTripleString = ItTS

+ virtual bool hasNext()
+ virtual TS* next()
+ virtual bool hasPrevious()
+ virtual TS* previous()
+ virtual goToStart()



FourSectionDictionary

```
- CSD* predicates
- CSD* shared
- CSD* subjects
- CSD* objects
- int mapping
- sizeStrings
- blocksize
- spec
```

```
- CSD* getDictionarySection
        (id, TCompRole)

- getGlobalID(Mapping, id, DicSec)
- getGlobalID(id, DicSec)
- getLocalID(mapping, id, TCompRole)
- getLocalID(id, TCompRole)
```

Type de dictionnaire créé dans BasicHDT :: createComponents(), utilisé dans tous les constructeurs de BasicHDT

type de CSD créé dans le constructeur de FourSectionDictionary

CSD_PFC

```
# int blocksize          # uint bytes
# LogSeq2 * blocks       # char* text
# int nblocks            # bool isMapped
```

```
+ int decompress(char ** dic)
# bool locateBlock(char *s, int* block)
# int locateInBlock(int block, char *s, int len)
# char* extractInBlock(int block, int o)
# inline longest_common_prefix
```

CSD (Compressed String Dictionary)

```
# char type
# int tlength
# int numstrings
```

```
+ virtual locate(char *, len)
+ virtual extract(int id) = 0
* virtual freeString(const char*) = 0
+ virtual getSize() = 0
+ virtual IteratorUCharString * listAll() = 0
+ getLength()
+ virtual fillSuggestions( - , - , - ) = 0
+ virtual save(ostream) = 0
+ virtual load(char* ptr, char*) = 0
+ static CSD* load(istream)
+ static CSD* create(char type)
```

PlainDictionary

```
- vector < DictEntry *> predicates
- vector < DictEntry *> shared
- vector < DictEntry *> subjects
- vector < DictEntry *> objects
- mapping
- sizeStrings
- spec
```

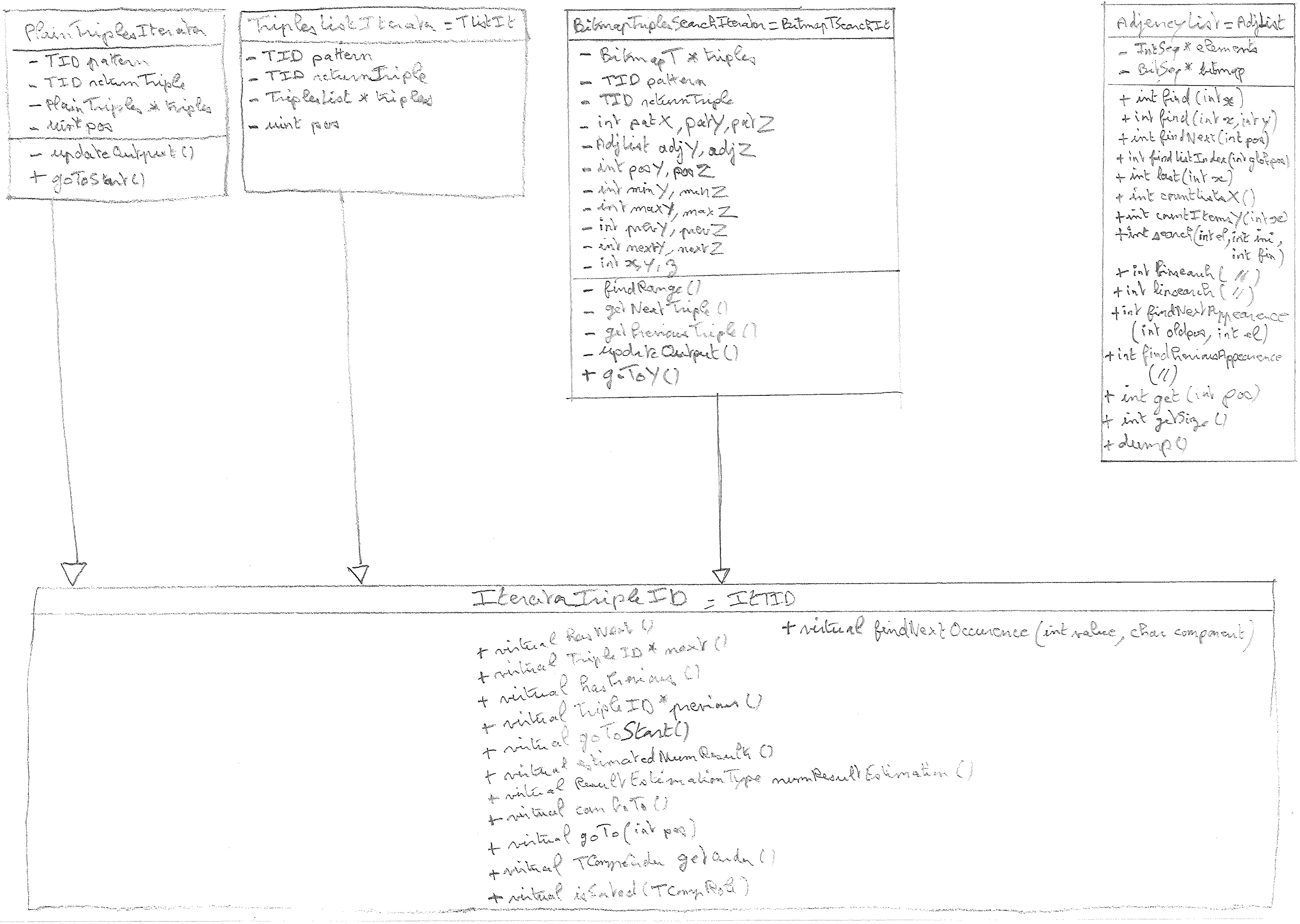
```
- insert(string, DicSec)                     + getGlobalID(mapping, id, DicSec)
- split()                                    + getGlobalID(id, DicSec)
- lexicographicSort()                        + getLocalID(mapping, id, TCompRole)
- idSort()                                   + getLocalID(id, TCompRole)
- updateIDs()                                + convertMapping(int)
- getDictionaryEntryVector(                  + updateID(int oldid, int newid, DicSec)
      int, TCompRole)
```

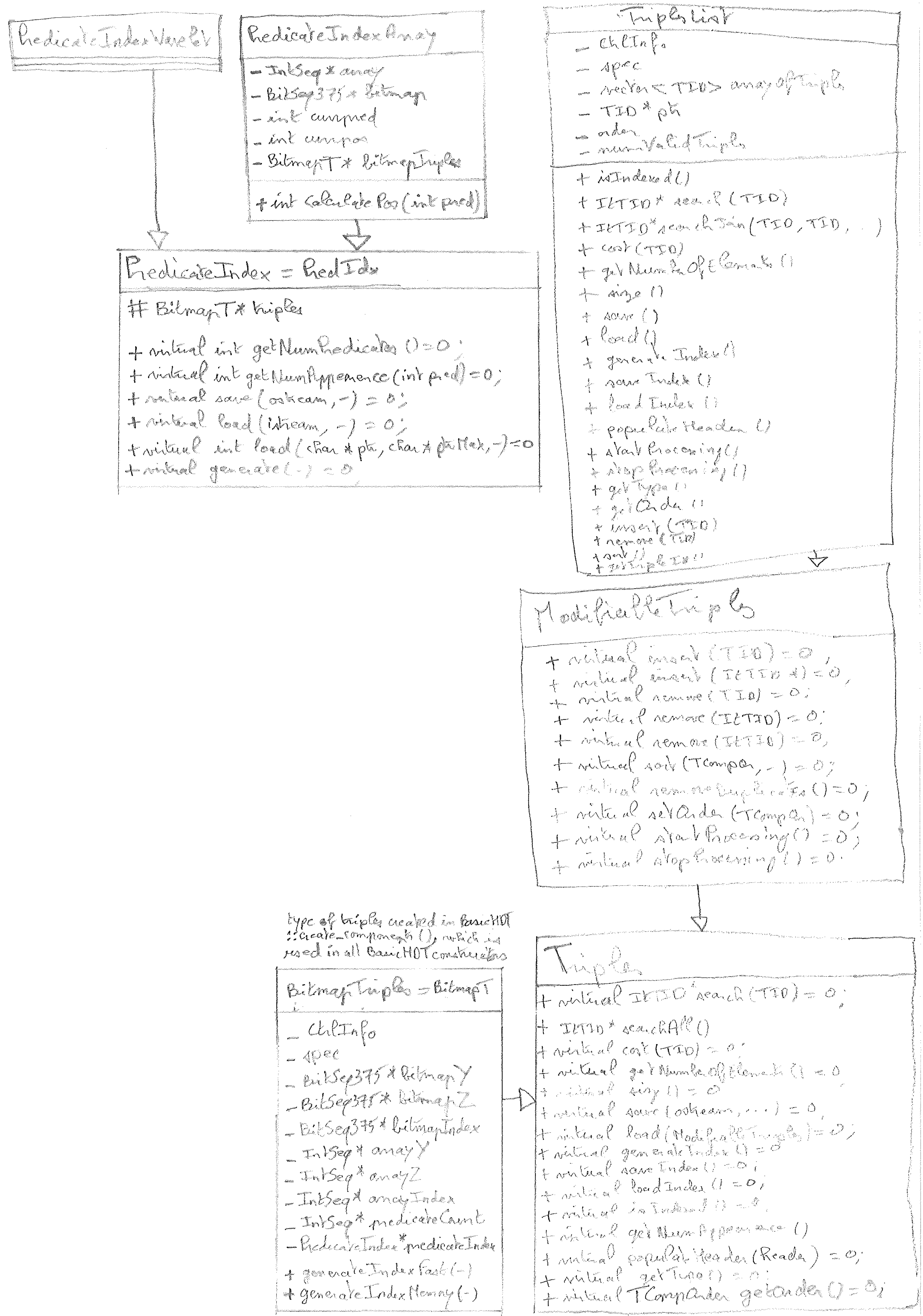
ModifiableDictionary

```
+ virtual insert(string, TCompRole) = 0
+ virtual startProcessing() = 0
+ virtual stopProcessing() = 0
```

Dictionary

```
+ virtual idToString(int, TCompRole) = 0
+ virtual stringToID(string, TCompRole) = 0
+ void tripleStringToTripleID(TS, TID)
+ void tripleIDToTripleString(TID, TS)
+ virtual getNumberOfElements() = 0
+ virtual size() = 0
+ virtual getNSubjects() = 0
+ virtual getNPredicates() = 0
+ virtual getNObjects() = 0
+ virtual getNShared() = 0
+ virtual getMaxID() = 0
+ virtual getMaxSubjectID() = 0
+ virtual getMaxPredicateID() = 0
+ virtual getMaxObjectID() = 0
+ virtual import(Dictionary) = 0
+ virtual getSubjects = 0
+ virtual getPredicates() = 0
+ virtual getObjects() = 0
+ virtual getShared() = 0
+ virtual populateHeader(Header &) = 0
+ virtual save(ostream) = 0
+ virtual load(char *) = 0
+ virtual getType() = 0
+ virtual getMapping() = 0
+ virtual getSuggestions() = 0
```


PlainTriplesIterator
- TID pattern
- TID returnTriple
- PlainTriples * triples
- uint pos
- updateOutput()
+ goToStart()
TriplesListIterator = TListIt
- TID pattern
- TID returnTriple
- TriplesList * triples
- uint pos
BitmapTriplesSearchIterator = BitmapTSearchIt
- BitmapT * triples
- TID pattern
- TID returnTriple
- int patX, patY, patZ
- AdjList adjY, adjZ
- int posY, posZ
- int minY, minZ
- int maxY, maxZ
- int prevY, prevZ
- int nextY, nextZ
- int x, y, z
- findRange()
- getNextTriple()
- getPreviousTriple()
- updateOutput()
+ goToY()
AdjencyList = AdjList
- IntSeq * elements
- BitSeq * bitmap
+ int find(int x)
+ int find(int x, int y)
+ int findNext(int pos)
+ int findListIndex(int globalpos)
+ int last(int x)
+ int countListsX()
+ int countItemsY(int x)
+ int search(int el, int ini, int fin)
+ int binsearch(//)
+ int linsearch(//)
+ int findNextAppearance (int oldpos, int el)
+ int findPreviousAppearence (//)
+ int get(int pos)
+ int getSize()
+ dump()
IteratorTripleID = ItTID
+ virtual hasNext()
+ virtual TripleID * next()
+ virtual hasPrevious()
+ virtual TripleID * previous()
+ virtual goToStart()
+ virtual estimatedNumResults()
+ virtual ResultEstimationType numResultEstimation()
+ virtual canGoTo()
+ virtual goTo(int pos)
+ virtual TComponentOrder getOrder()
+ virtual isSorted(TCompRole)
+ virtual findNextOccurence(int value, char component)


type of triples created in BasicHDT ::create_components(), which is used in all BasicHDT constructors

objet retourné par HDTManager :: mapHDT