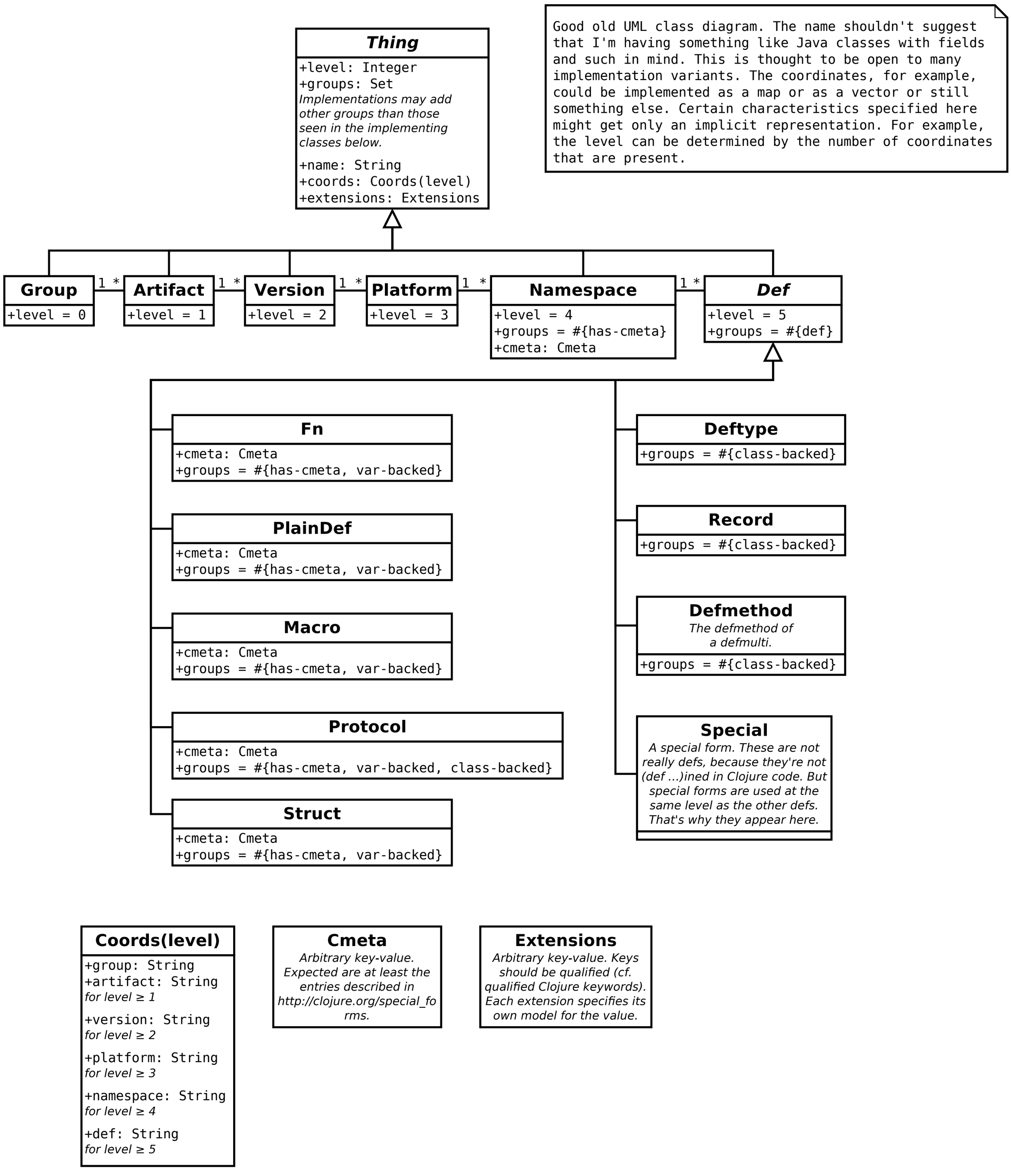
Thing
+level: Integer
+groups: Set
Implementations may add other groups than those seen in the implementing classes below.
+name: String
+coords: Coords(level)
+extensions: Extensions
Good old UML class diagram. The name shouldn't suggest that I'm having something like Java classes with fields and such in mind. This is thought to be open to many implementation variants. The coordinates, for example, could be implemented as a map or as a vector or still something else. Certain characteristics specified here might get only an implicit representation. For example, the level can be determined by the number of coordinates that are present.
Group
+level = 0
1 *
Artifact
+level = 1
1 *
Version
+level = 2
1 *
Platform
+level = 3
1 *
Namespace
+level = 4
+groups = #{has-cmeta}
+cmeta: Cmeta
1 *
Def
+level = 5
+groups = #{def}
Fn
+cmeta: Cmeta
+groups = #{has-cmeta, var-backed}
PlainDef
+cmeta: Cmeta
+groups = #{has-cmeta, var-backed}
Macro
+cmeta: Cmeta
+groups = #{has-cmeta, var-backed}
Protocol
+cmeta: Cmeta
+groups = #{has-cmeta, var-backed, class-backed}
Struct
+cmeta: Cmeta
+groups = #{has-cmeta, var-backed}
Deftype
+groups = #{class-backed}
Record
+groups = #{class-backed}
Defmethod
The defmethod of a defmulti.
+groups = #{class-backed}
Special
A special form. These are not really defs, because they're not (def ...)ined in Clojure code. But special forms are used at the same level as the other defs. That's why they appear here.
Coords(level)
+group: String
+artifact: String
for level ≥ 1
+version: String
for level ≥ 2
+platform: String
for level ≥ 3
+namespace: String
for level ≥ 4
+def: String
for level ≥ 5
Cmeta
Arbitrary key-value. Expected are at least the entries described in http://clojure.org/special_forms.
Extensions
Arbitrary key-value. Keys should be qualified (cf. qualified Clojure keywords). Each extension specifies its own model for the value.