ICOM/CIDOC

DATA MODEL WORKING GROUP

Graphic Data Model

This graphic model covers the big four entities: objects, events, people, and roles. It gives you an alternative way to access information about the CIDOC data model. Use it in concert with the data documentation provided. Each box on the graphic model should correspond to an LDG in the textual documentation. Not all LDGs are represented on the graphic model.

CIDOC Relational Data Model

A Guide

by Patricia Ann Reed

April 1995

Revised April 1995

CIDOC Relational Data Model

A Guide

Table of Contents

CIDOC Relational Data Model

A Guide

Introduction

The CIDOC[1] Data Model Working Group is creating a relational data model as a prerequisite to recommending a relational data structure for the interchange of museum information worldwide.

Advances in database technology and processing offer opportunities for using information flexibly and efficiently when data is organized and stored in relational structures.

This guide is for those who wish a better understanding of relational data modeling - its purpose, its nature, and the standards used in creating the CIDOC model. The examples used are found in the CIDOC model reports.

A relational data model defines what the data is rather than how it is used, because data is used in multiple applications to serve multiple functions. For example, data is collected about **Object**, not **Object-on-loan** or **Object-being-photographed** or **Object-acquired-from-donor. Loan, photograph,** and **acquire** are functional contexts - the settings in which **Object** information is used. In relational technology, each automated function uses the same **Object** data.

This is a sea change in thinking for many museum professionals responsible for the management of their collections. If data was automated in the past, it was stored in flat file structures where duplicating the data was the only way to automate multiple functions or activities. Today's technologies, supported by a well-defined relational data model, offer better solutions.

[1] The International Documentation Committee of the International Council of Museums.

I. Purpose of a Relational Data Model

Data is the raw material from which information is produced, and it can be stored on disk, on tape, or in a file drawer (or in a brain!). **Information** is data processed and presented in meaningful form and context.

Data is collected, modeled, and documented to serve functions. In other words, data must support what is done and provide the information needed to perform daily tasks and plan for the future.

Data separated into its smallest discrete parts and defined precisely can be organized in a structure which achieves the following objectives:

- Eliminate logical data redundancy, thereby reducing physical data redundancy.
- Ensure consistency of logical data names and definitions within and across systems and disciplines.
- Enable multiple use of physical databases.
- Enable greater flexibility of data usage.
- Enhance the capability to deliver decision support information.
- Provide data structures which enable data interchange across systems and disciplines.

It is the last objective which is the goal of the CIDOC Data Model Working Group.

II. Logical Data Model - What It Is, What It Isn't

At the highest level of abstraction, there are five big entities which can be defined and documented:

People **Places** **Things** **Events** **Concepts**

These five entities and the relationships among them can document anything in the entire spectrum of human (or inhuman) experience.

This highest-level model is sometimes called a **Conceptual Data Model.** It contains major entities, broadly defined and without attributes or details.

The task of a **Logical Data Model** is to particularize the Conceptual Data Model entities and relate them to each other, creating a data structure which supports the intellectual and physical worlds in which work is done.

A logical data model does not contain real data. Rather, it contains the infrastructure into which real data fits. This section describes the infrastructure and distinguishes it from the physical database structure.

A. Metadata

Data in a relational data model is called **metadata,** i.e., **data about data.**

Metadata provides

- a commonly understood body of data which can be used in multiple applications and
- common data structures which users from diverse process areas can populate with unique data values.

B. Principles for Creating Metadata

When defining metadata, the following principles apply:

- Logical data is defined in the abstract and without redundancy.

- Logical data is defined independent of, and outside the context of, functions, processes, and automated applications.

- Logical data is defined by users from diverse functional areas who need the same logical data.

- Logical data element names are consistent and meaningful; they are created according to naming standards. (See *Section III. Defining and Naming Logical Data)*

- Composite data is broken down into its smallest meaningful parts, each of which is defined separately.

C. Data Model and Database Schema

The logical data model contains the characteristics of real data, whereas a physical database contains real data. The following comparative table characterizes the differences between metadata in a relational data model and data descriptions (also called data schema or record layouts) for the contents of a physical database.

Relational Data Model	**Physical Database**
Logical, abstract in nature.	Physical in nature.
Contains metadata, i.e., data about data.	Contains real data.
Contains information about the attributes of data entities and the logical relationships among them.	Contains a body of data facts which are instances, or occurrences, of logical data entities.
Stable, reusable product; logical data definitions seldom change; relationships among data entities seldom change.	Technologies change; over time, changes in hardware and software force migrations to new information systems implementations.
Logical data is defined and documented independent of, and outside the context of, functions, processes, and automated applications.	Physical data is stored and used in the context or one or more automated or manual processes to satisfy a functional need.

Relational Data Model	Physical Database
Logical data is defined without redundancy.	Physical data may be stored redundantly to meet physical processing requirements and/or constraints.
Composite data is broken down and logically defined at the level of the smallest meaningful part.	Logically defined data elements may be combined in a physical database to form a data group which is commonly known as a business attribute or entity.

D. Logical Data Groups (LDGs) and Logical Data Elements (LDEs)

The logical data model contains information about two levels of data: Logical Data Group (LDG) and Logical Data Element (LDE). In this discussion, the terms "LDG" and "Element" are used. LDGs are groups of Elements. Elements are the discrete pieces of data which describe and define entities.

1. LDGs

LDGs are logical groups of data which define and describe entities. They can be equated roughly to a physical data record, database schema, or relational table.

In the CIDOC model, LDGs are designated as **primary, repetition, recursion, type,** or **intersection** in the "LDG TYPE" category.

A **primary entity** is something which is important to an organization's work, in this case museum work. There are two questions to ask in determining whether an entity is primary: "Can it stand alone, or is it merely an attribute?" and "If it can stand alone, do we want to define its attributes and document it as a separate entity?"

Some primary entities originally were thought to be attributes of another entity. These former attributes became primary entities because they were not intrinsic to the entity itself, and because users wanted to keep detailed information about them. An example is STYLE, which originally was considered an attribute of OBJECT. However, STYLE is not dependent on OBJECT for its existence - it can stand alone, has attributes of its own, and users want to describe it in more detail. New technologies make possible this discrete separation of entities.

Primary entities in the current CIDOC model are ALPHABET, AWARD, CALENDAR, CLASSIFICATION, COLOR, CONCEPT, EVENT, LANGUAGE, MATERIAL, METHOD, OBJECT, OCCUPATION, OPUS, PEOPLE-GROUP, PEOPLE-PERSON, PLACE, ROLE, STYLE, AND TIME-SPAN.

A **repetition entity** is created when an attribute can occur more than one time for any given occurrence of an entity. An example is OBJECT MARK LDG. MARK is an attribute of OBJECT. Because more than one mark may appear on any given OBJECT, MARK is removed from the OBJECT LDG and becomes a repetition entity. OBJECT MARK LDG has its own repetition entity called OBJECT MARK TRANSCRIPTION LDG because there can be more than one TRANSCRIPTION for any given MARK. OBJECT MARK TRANSCRIPTION LDG has its own repetition entity called OBJECT MARK TRSCRPTN TRANSLN LDG because there can be more than one TRANSLATION of any given TRANSCRIPTION.

A **recursion entity** is an entity which is related to itself. It is indicated by the term "RELATED" in the LDG name. PEOPLE RELATED LDG is an example of a recursion entity, where two instances of PEOPLE LDG are associated. In PEOPLE RELATED LDG, there are two occurrences of the Elements PEOPLE OCC IDN and ROLE OCC IDN which represent either two persons, two groups of persons, or a person and a group of persons; an Element called PEOPLE PEOPLE RELATIONSHIP NAM which documents the nature of the association between the two PEOPLE; and Elements documenting the time during which the relationship occurred.

An **intersection entity** is created by linking together two or more primary, repetition, or type entities. Intersection entities are indicated in the CIDOC model by an ampersand (&). An example is OBJECT & EVENT LDG, where an OBJECT is associated with an EVENT. The intersection entity contains Elements which document the association of the OBJECT and the EVENT, i.e., the relationship between them and the time during which the relationship occurred.

A **type entity** is a subset of a primary entity. It has special attributes which set it apart from the larger entity.

2. Elements

Although "Element" and "attribute" sometimes are used interchangeably, in the context of this document there is a difference: "Element" is a data fact logically defined and contained within an LDG. "Attribute" is an intrinsic characteristic of an entity.

Elements define the attributes of entities, answering the question "What is it?" They can be equated roughly to the data fields in a flat file or the columns in a relational table.

Elements comprise the contents of LDGs. An Element is dependent on an entity - it cannot exist apart it. In the CIDOC Model, for example, "OBJECT LDG" contains the Elements "OBJECT OCC IDN", "OBJECT CNT", and "OBJECT MEDIUM SUPPORT DISPLAY," which describe OBJECT and cannot exist apart from OBJECT.

Elements defining many of the attributes of entities are documented in repetition LDGs. For example, MARK is an attribute of OBJECT, although no Elements describing MARK appear in the OBJECT LDG. The Elements describing MARK appear in the repetition entity OBJECT MARK LDG because there can be more than one MARK for any given OBJECT.

III. Standards for Defining and Naming Logical Data

Using standards to define and name LDGs and Elements assures consistency and reliability in metadata retrieval and usage. These standards are for logical, not physical, data. Standards do not preclude the use of traditional, familiar data names in data entry screens, forms, reports, and the like.

A. Defining Logical Data

- **Standard: Logical data is defined without reference to and outside the context of process, function, or physical information system.**

Relational	Non-Relational
OBJECT & EVENT LDG	OBJECT LOANED
OBJECT & EVENT LDG	OBJECT ACQUIRED
OBJECT & EVENT LDG	OBJECT CATALOGUED

In the non-relational example above, the words LOANED, ACQUIRED, and CATALOGUED describe the context in which an OBJECT was used, and they do not describe intrinsically the OBJECT itself. They are EVENTs in which an OBJECT participated.

In the relational example, the OBJECT is stored once in an information system, each EVENT is stored once, and OBJECTs and EVENTs are linked together when appropriate.

- **Standard: Differences between data element types and data element values are resolved.**

Relational	Non-Relational
PEOPLE PERSON LDG	CALLIGRAPHER
ROLE LDG	PAINTER
	PRINTER
	DONOR

The non-relational examples above are typical of data defined in a flat-file OBJECT record. In the non-relational examples four pieces of data are defined as roles, and

each will be populated with a person's name. Conceivably, the same person's name could populate all four of the non-relational data definitions. In addition, that same person may be logically related to additional objects.

Relational modeling and technology solve both these anomalies by separating a person from a role he plays and creating a data group for each. Once information about a person is stored in a database, it can be linked to many roles related to the same object, and the combination of person and role can be linked to many different objects.

Another benefit occurs when a new ROLE is desired: Instead of defining a new piece of data, one only need add a new value to the ROLE database.

- **Standard: An Element appears in one, and only one, LDG. The exception is a foreign key[2], which may appear in multiple intersection LDGs.**

Relational	Non-Relational
OBJECT LDG	MARK1
OBJECT MARK LDG	MARK2
	SIGNATURE

This example was taken from a flat-file OBJECT record. These three data elements appeared in every OBJECT record, whether they were populated or not. Accepting that SIGNATURE is a kind of MARK, there are three MARK data elements in the flat-file OBJECT record.

By removing the MARKs from the OBJECT record and creating a Repetition Entity called OBJECT MARK LDG, it is now possible to document an unlimited number of MARKs without defining additional data elements. Data elements within the OBJECT MARK LDG describe the MARK fully, eliminating the need for the SIGNATURE data element in the flat-file structure.

B. Naming Logical Data

Data dictionary names reflect the abstract, process-independent nature of a relational data model. The following standards for naming logical data impose a structure which facilitates understanding a complex set of data requirements.

[2] A foreign key is the primary key of another entity; a primary key is the Element or group of Elements which uniquely defines an occurrence of an entity.

- **Standard: Nouns are used in singular form.**

Relational	Non-Relational
OBJECT LDG	OBJECTS LDG
EVENT ACTION LDG	EVENT ACTIONS LDG
OBJECT MARK LDG	OBJECT MARKS LDG

- **Standard: Logical data names are ordered by facet, or segment, according to the following formula:**

  PRIMEWORD MODIFIER(S) CLASSWORD/SUFFIX

  The facets are separated by a space.

  CLASSWORD applies only to Elements, and **SUFFIX** applies to LDGs.

  The purpose of using CLASSWORD and SUFFIX is to indicate at-a-glance what kind of dictionary entry one sees. The dictionary can be expanded to document other kinds of information such as Users, Applications, Systems, and Modules, for which one might choose suffixes of USE, APP, SYS, and MOD.

Following are standards for each facet of a logical name:

- **Standard: PRIMEWORD represents the name of a primary entity to which a LDG or Element belongs. It must be the first facet in a name.**

Relational	Non-Relational
OBJECT LDG	LDG OBJECT
OBJECT CONDITION NAM	NAME CONDITION OBJECT
OBJECT MEASURE LDG	MEASURE OBJECT LDG
OBJECT MARK OCC IDN	IDN OCC OBJECT MARK

- **Standard: MODIFIER qualifies and further defines a LDG or an Element emanating from a major entity. Ordering of multiple modifiers is left to right from general to specific.**

  Example: OBJECT LDG
  OBJECT MARK LDG

OBJECT MARK TRANSCRIPTION LDG
OBJECT MARK TRSCRPTN TRANSLN LDG
(TRANSCRIPTION and TRANSLATION abbreviated because of software length constraints)

In the above example the placement of modifiers is left to right from general to specific. OBJECT MARK LDG indicates that MARK is an attribute of OBJECT; OBJECT MARK TRANSCRIPTION LDG indicates that TRANSCRIPTION is an attribute of a MARK on an OBJECT; and OBJECT MARK TRSCRPTN TRANSLN LDG indicates that TRANSLATION is an attribute of a TRANSCRIPTION of a MARK on an OBJECT.

The LDGs above are examples of the **Repetition Entity.**

- **Standard: The key identifier of an LDG is indicated by an Element containing the standard modifier "OCC". The modifier "OCC" precedes immediately the Element CLASSWORD "IDN" (see CLASSWORDs below).**

Key Identifier in this context is defined as the unique identifier by which a computer recognizes a unique occurrence of a data group. The identifier may be machine-generated to guarantee uniqueness.

Examples: EVENT OCC IDN
CLASSIFICATION TERM OCC IDN
PLACE ADDRESS OCC IDN

- **Standard: CLASSWORD defines the intrinsic or inherent nature of an Element. It is the last facet of an Element name.**

The following CLASSWORDs are mutually exclusive categories which define the nature of an Element and answer the question "What is it?"

Class Word	**Meaning**
AMT	Amount (numeric) Indicates a monetary amount. (How much?)
CDE	Code (alphanumeric) Predefined values which represent specific names or terms and are formulated by the systematic use of symbols, letters, or numbers. Ex: Codes for country names, i.e., UK is a code for United Kingdom, FR

for France, etc. Codes may be standard, universal, or specific to a local system. Multiple code sets may exist for the same entity, as is the case for country names.

CNT Count (numeric)
Indicates a non-monetary numeric quantity or accumulation. (How many?)

FLG Flag (alphanumeric)
Indicates a binary state or condition where only two opposite values are possible, and where the values have no function other than to indicate a described state or condition. (YES or NO, ON or OFF, IS or IS NOT)

IDN Identifier (alphanumeric)
Non-coded data which identifies an entity; not necessarily unique. (Ex: Museum catalog number, donor catalog number, exhibition catalog number, specimen tag number, and employee number cannot be **guaranteed** to be unique within a database.)

NAM Name (alphanumeric)
Alphanumeric data which documents an appellation, or name, given to a person or organization, place, thing, event, or concept. May be a single word or a short phrase; different in nature from "TXT".

TME Time (alphanumeric)
Identifies a duration or period of time, including dates, or a specific instant in which something occurs. (When?)
Format is standard ISO (International Organization for Standardization) format:

YYYYMMDDHHMMSS.SS

YYYY	year
MM	month
DD	day
HH	hour
MM	minute
SS	second
.SS	tenths, hundredths of second

TXT Text (alphanumeric)
Textual data which is imprecisely defined, has an unpredictable structure, and does not fit into one of the above classifications. Typically consists of notes, remarks, descriptions, and comments.

The following examples illustrate how CLASSWORD is used in naming a data element:

Relational	Non-Relational
OBJECT PART CNT	NUMBER OF OBJECT PARTS
CALENDAR NAM	NAME OF CALENDAR
CONCEPT APPELLATION NAM	NAME GIVEN TO CONCEPT
PLACE ADDRESS BUILDING IDN	BUILDING NUMBER

- **Standard: The standard SUFFIX for LDGs is "LDG".**

  Examples: OBJECT LDG
  OBJECT MARK LDG

- **Standard: The ampersand - "&" - is the standard character for documenting the linking of one LDG with another, indicating relationships among entities.**

  Examples: OBJECT & EVENT LDG
  OBJECT NOTE & PEOPLE PERSON LDG
  OBJECT & PEOPLE & ROLE LDG

- **Standard: Each facet in a logical data name is spelled in full. Abbreviations are used when needed to accommodate the 32-character length limit imposed by the current software which documents the model.**

  If abbreviations are necessary, begin with the MODIFIER facets, from specific to general (right to left), when possible. **CLASSWORD** and **SUFFIX** are not abbreviated.

C. Adapting Standards to Local Environments

While reviewing the standards in this document, there are considerations to keep in mind, especially if information will be stored in a commercial software package such as a data dictionary or a CASE (computer assisted software engineering) tool. A few of these considerations are listed below:

- Some software does not permit spaces to be used between facets of a name; a dash or underscore may be required.

Example: OBJECT & EVENT LDG
OBJECT-&-EVENT-LDG
OBJECT_&_EVENT_LDG

- The software which produced the CIDOC Data Model documentation accommodates use of the ampersand (&) to link one LDG to another. Other software products preclude the use of special characters. Another single character may be substituted, or the linking character may be omitted altogether.

  Example: OBJECT & EVENT LDG
  OBJECT A EVENT LDG
  OBJECT N EVENT LDG
  OBJECT EVENT LDG

- Some software packages allow only upper case or only mixed case alphabetic characters in a dictionary name, while others allow a choice of upper case, lower case, mixed case, and special characters including spaces.

- A dictionary name may be limited in length to a specific number of characters. The software used in the accompanying reports allows a maximum of 32 characters, thus forcing abbreviations in complex names. The abbreviations are predetermined to assure consistency.

- Become familiar with all the features of a software package before setting standards for its use.

- If multiple software packages are used, consider compatibility.

IV. Data Dictionary Reports

The term **data dictionary** is used to describe 1) a repository for the definition of logical metadata and 2) a DBMS-specific description of a schema, or record layout, for storing physical data. It is the first definition which documents the CIDOC data model.

There are three reports comprising the documentation package: LIST OF ENTITIES BY TYPE, ENTITY CONTENTS REPORT, and USED-BY DIRECTLY.

The LIST OF ENTITIES alphabetically lists first the Elements and then the LDGs.

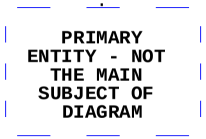
The ENTITY CONTENTS REPORT contains a full description of Elements and LDGs, entries appearing together in alphabetical order. The VALUES attribute[3] (field) in an Element entry is intended to further define logically the Element by providing examples of real data values which might appear in a physical implementation. The CONTAINS attribute (field) in an LDG entry lists the Elements which comprise the LDG. Other fields are self-explanatory.

The USED-BY DIRECTLY lists alphabetically each Element along with the LDGs in which it is found.

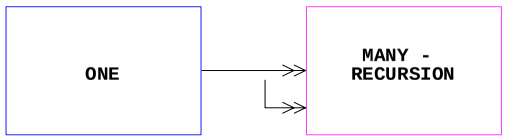
[3] A semantic consideration - "attribute" is the term PC Dictionary uses for the fields in an entity entry.

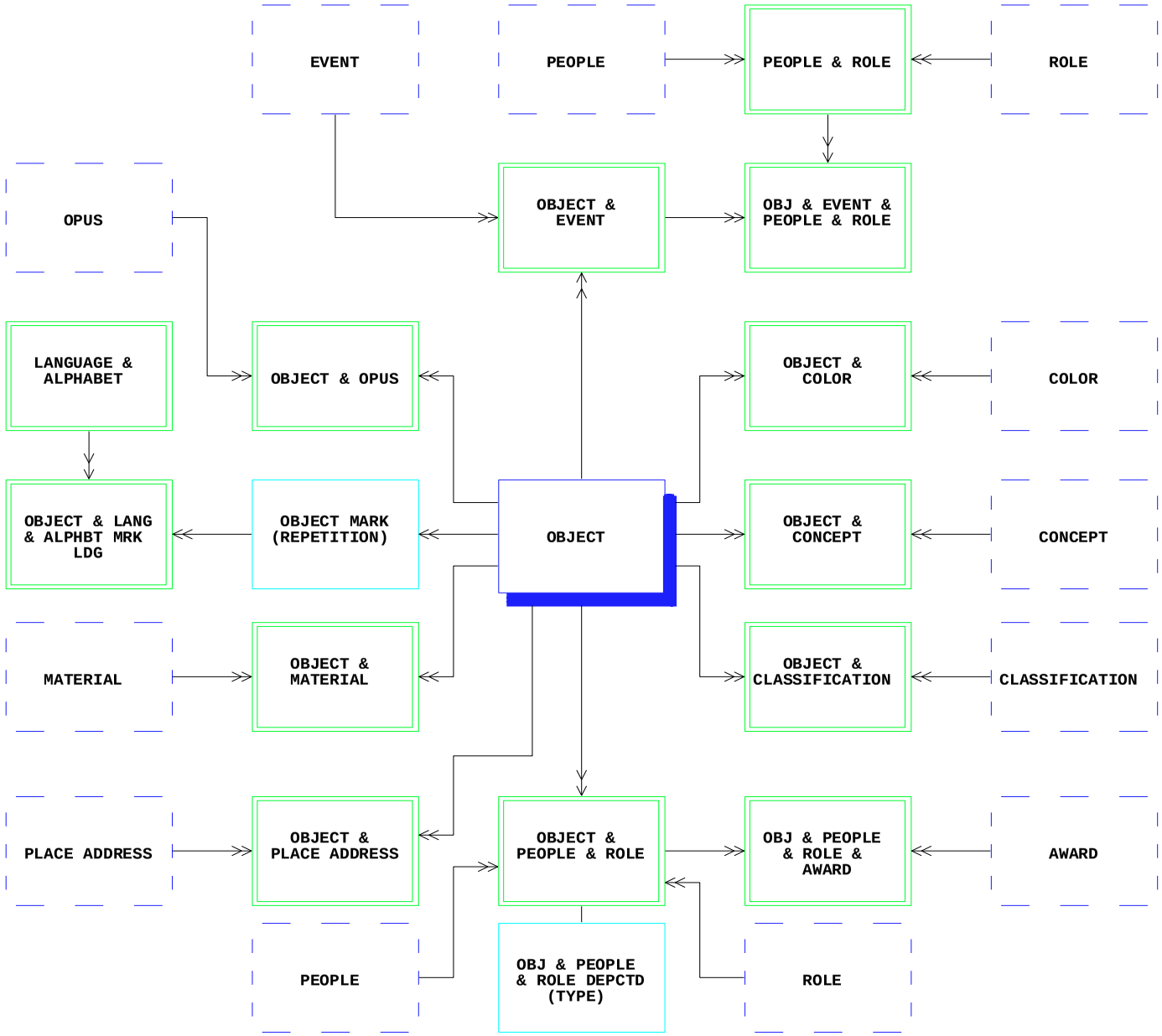
SYMBOLS USED IN CIDOC MODEL

BOXES REPRESENT DIFFERENT KINDS OF ENTITIES:

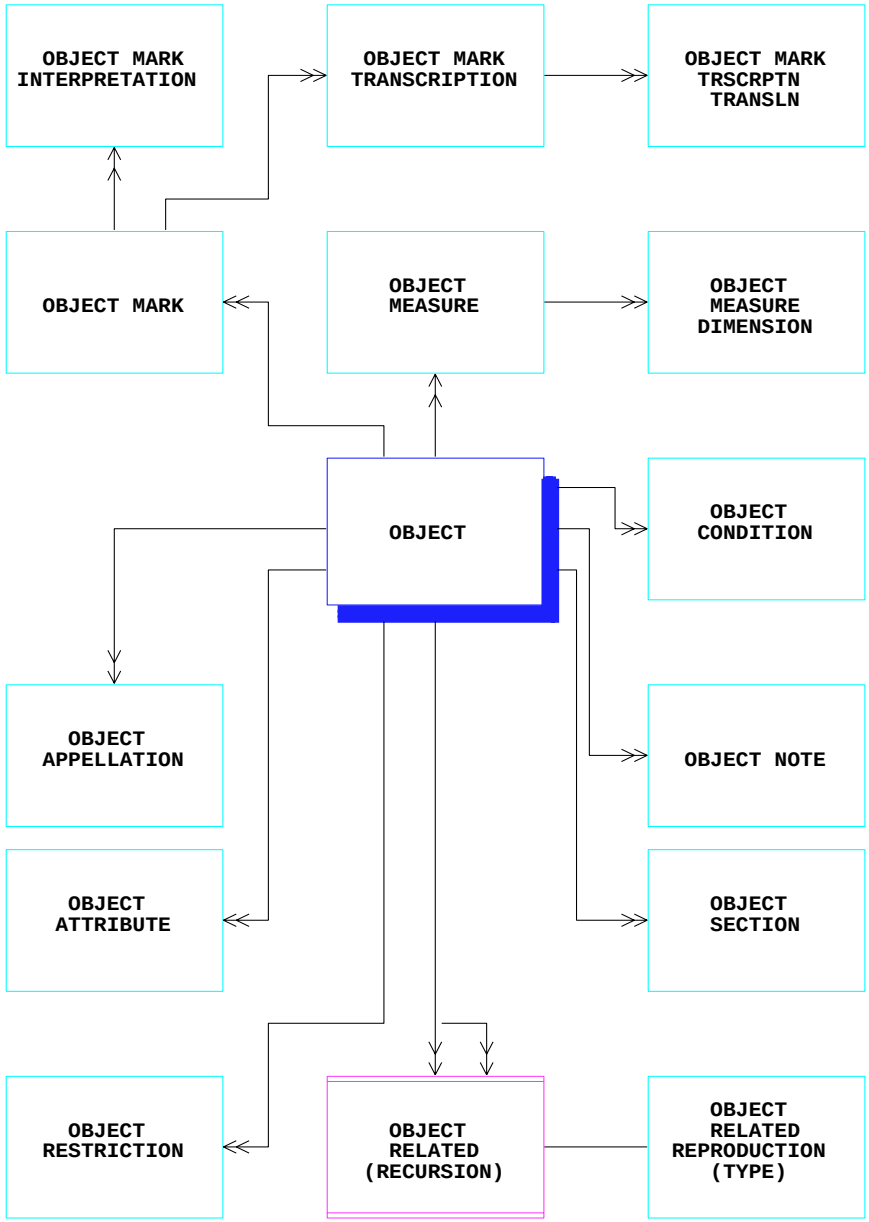


RELATIONSHIPS REPRESENTED BY CONNECTING LINES:

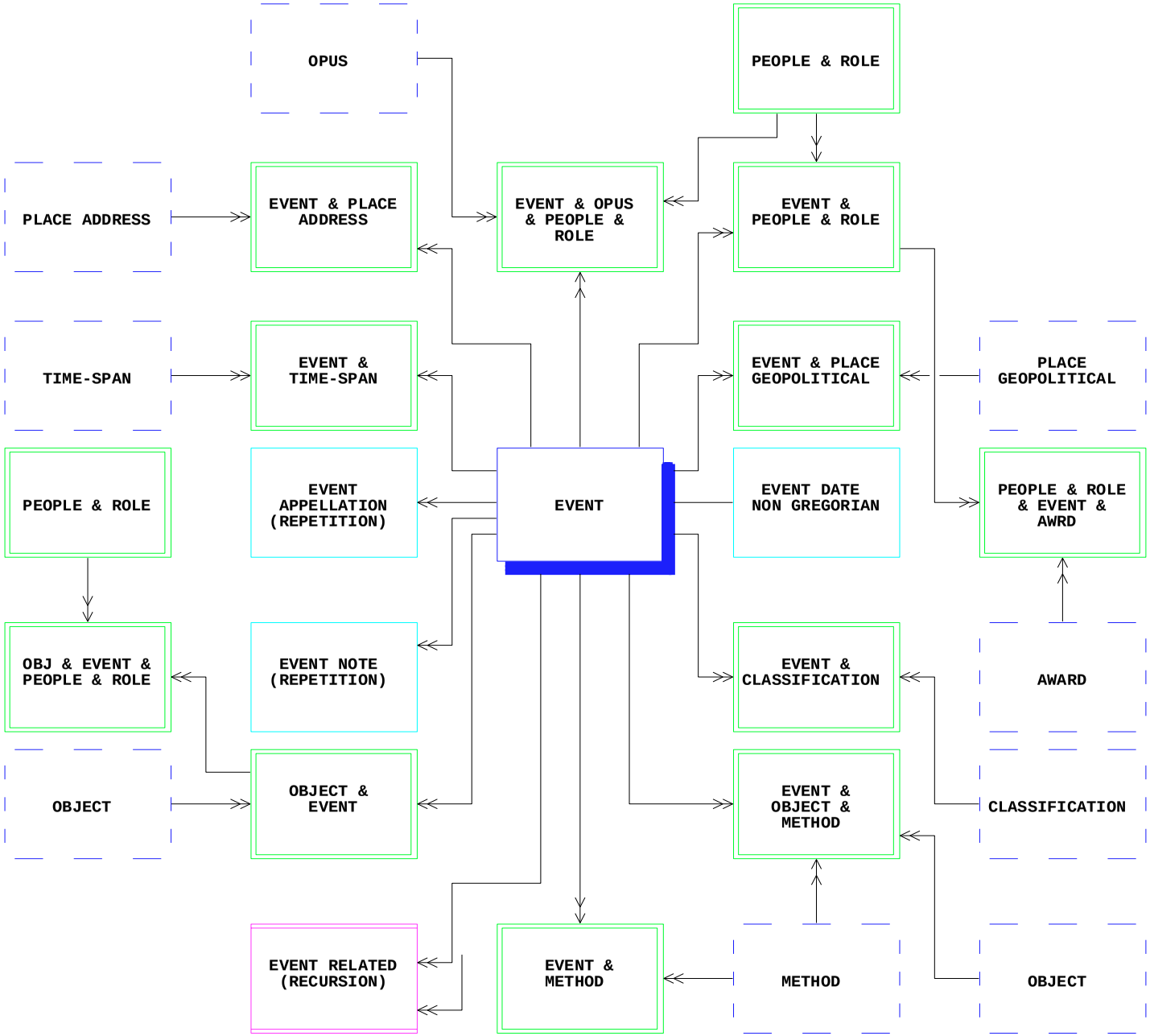




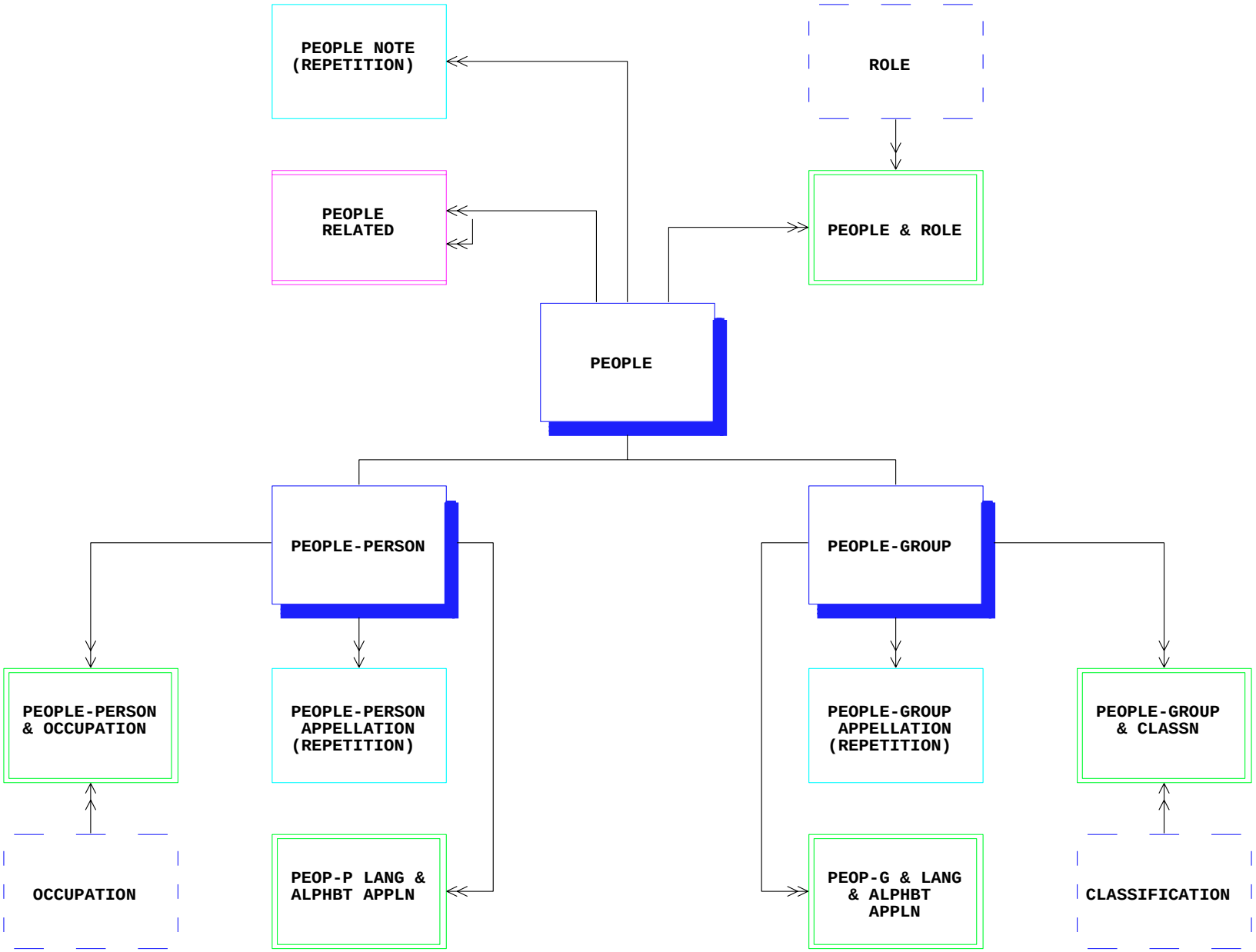
INTERSECTION ENTITIES RELATED TO "OBJECT". THE ENTITIES IN BROKEN-LINE BOXES SHOW A CONNECTION TO OTHER PARTS OF THE CIDOC DATA MODEL. YOU CAN TRACE THE ENTITY YOU ARE INTERESTED IN TO THE PRIMARY ENTITIES RELATED TO IT.



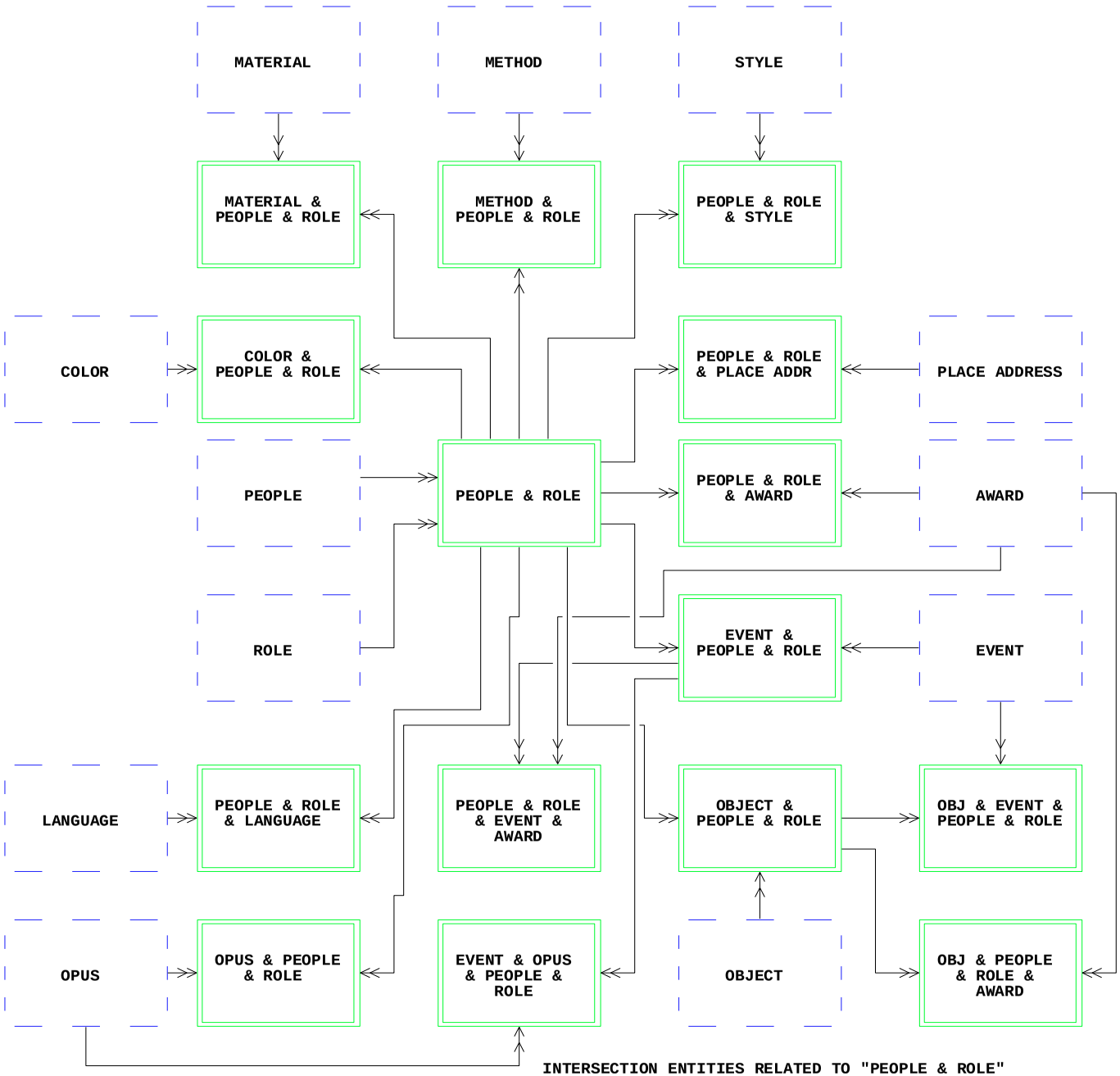
REPETITION ENTITIES OF "OBJECT"; ONE RECURSION ENTITY OF "OBJECT; A TYPE ENTITY OF THE RECURSION ENTITY.



ALL ENTITIES RELATED TO THE PRIMARY ENTITY "EVENT".



ENTITIES COMMON TO ALL 'PEOPLE', ENTITIES UNIQUE TO 'PEOPLE-PERSON', ENTITIES UNIQUE TO 'PEOPLE-GROUP'



INTERSECTION ENTITIES RELATED TO "PEOPLE & ROLE"