A GENTECH Primer

1. Where to get it

You can get the GENTECH Genealogical Data Model specifications at www.gentech.org/gdm . You will want both the description document and the model and process diagrams. By following the sections below, you will gain a good working knowledge of the most important parts of the data model.

2. Personas: a slice of a timeline

In the data model, a Persona comes in two flavors:

- a "base" Persona, which is a person based strictly on one source document, and
- a "composite" Persona, which is a person based on other, "lower-level" base or composite Personas.

For example, suppose you came across a document which states that John Brown is of age 43, and another document which states that John Brown lived in New York in 1903. If you created a single Persona, "John Brown", who was of age 43 and lived in New York in 1903, then you have assumed that the John Brown in both documents are one and the same. The disadvantages of using a single Persona are many.

- If you later find out those John Browns are different, you will have a difficult time picking apart the Personas, since you made only one.
- A researcher looking at your work may not spot the assumption, and base their work on possibly erroneous reasoning.
- A reasercher looking at your work may spot the assumption, and wonder what other hidden assumptions you made.
- You will forget why you made that assumption.

The rule is:

one document = one Persona.

A Persona is given a name which you can consider a "working" name. It is not the "official" name of the person. Should the official name be the name a person was born with? The name a person called themselves? The name on their death certificate? The answer is, anything you want. A person has a name, but the name is not the person. Names change over time, just as age does. Just as you wouldn't uniquely identify a person by age, you wouldn't uniquely identify a person by name. So if one document gives a name as "John Braun" and another one gives it as "John Brown", you will have two Personas with different names, which you might conclude are one and the same person. A researcher searching through databases may not know one or the other name, and thus they will catch your data regardless of what name they know.

3. Assertions: the basic unit of genealogy

Assertions are facts. There are various types of facts that can be recorded in the data model, and they can be summarized in this table:

Column A	Column B
A Persona...	...is a Persona
An Event...	...played a role in an Event
A Characteristic...	...had a Characteristic
A Group...	...is part of a Group

Choose one from Column A and one from Column B. Certain combinations are not allowed. We will go through all possible 16 assertions.

3.1.1. A Persona is a Persona (disallowed)

This is not allowed. If you want to state that Persona A is the same as Persona B, then you must create a new Persona C which combines the two (see "A Group is a Persona" and "A Persona is part of a Group").

3.1.2. A Persona played a role in an Event

Here you are stating that something happened, and the Persona played a part in it. For example, an event could be a birth. The roles are things like father, mother, child, midwife, etc. When you say "John Smith was born to Abe Smith and Betty Tipper", you are stating three facts:

- John Smith was the child in the event "birth of John Smith",
- Abe Smith was the father in the event "birth of John Smith", and
- Betty Tipper was the mother in the event "birth of John Smith".

These assertions could come from different documents, which is why the assertions in that seemingly simple statement need to be kept separate.

3.1.3. A Persona had a Characteristic

Here you are asserting a characteristic of a Persona. Age, race, religion, hair color, height, and occupation are all examples of characteristics.

3.1.4. A Persona is part of a Group

Groups are logical aggregations of things of the same type. Personas can only be a part of a Group composed of other Personas. Persona Groups are used primarily to create higher-level Personas from lower-level ones.

However, one could also use Persona Groups to represent actual groupings of people. For example, suppose you wanted to represent the assertion that "John Smith was in Platoon X". You could:

- create a group called "People in Platoon X" and assert that the Persona "John Smith" was part of that group, or
- consider membership in Platoon X as a Characteristic and assert that the Persona "John Smith" had that characteristic.

If you really care who else was in that platoon, then you would create a Group. Otherwise it is just as reasonable to create a Characteristic.

3.2.1. An Event is a Persona (disallowed)

This doesn't make any sense.

3.2.2. An Event played a role in an Event

This may not seem to make sense. You could consider this statement "An Event happened relative to an Event". Thus, one Event could happen before another Event, and you would represent this by using the Event-Event Assertion. Whether you say Event A happened before Event B or Event B happened after Event A is irrelevant, since both mean the same thing, and the program should be expected to understand the equivalence.

3.2.3. An Event had a Characteristic (disallowed)

While on the surface this may seem to make sense, any possible Characteristic an Event might have -- namely date and place -- are saved within the Event itself.

3.2.4. An Event is part of a Group

Just as Personas can be optionally Grouped for membership, so too can Events. For example, a Group could be "things that happened to John Smith". This is not recommended, since which Persona is "John Smith"? The same information can be gained by starting with a Persona and then finding all the Events tied to that Persona and its lower-level Personas.

However, this kind of assertion can represent Events that didn't happen to Personas. For example, a Group could be "things that happened during World War I". This could serve as a useful reference tool.

3.3.1. A Characteristic is a Persona (disallowed)

This doesn't make any sense.

3.3.2. A Characteristic is an Event (disallowed)

This also doesn't make any sense.

3.3.3. A Characteristic is a Characteristic (disallowed)

This doesn't make sense, even in the sense of Event-Event assertions, since Characteristics are not relative to each other.

3.3.4. A Characteristic is part of a Group (not recommended)

This use is not recommended for Groups of Characterstics about Personas, since the same information can be gained by finding all the Characteristics tied to that Persona and its lower-level Personas.

3.4.1. A Group is a Persona

This is the second part of composing a higher-level Persona out of lower-level Personas. Once a Group of low-level Personas is created, a Group can be asserted to be equivalent to a new Persona. This only applies to Groups of Personas.

3.4.2. A Group is an Event (disallowed)

This doesn't make any sense.

3.4.3. A Group is a Characteristic (disallowed)

This also doesn't make any sense.

3.4.4. A Group is part of a Group

This can be used when sub-groups are grouped together.

3.5. All the combinations

Of the 16 combinations, fully half are disallowed. Actually, the GENTECH organization merely proposes their prohibition, so they might come up with a use for a disallowed combination.

3.6. Examples

These diagrams show how statements in documents are converted to Assertions in the data model.

3.6.1 John Smith was born in New York in 1887.

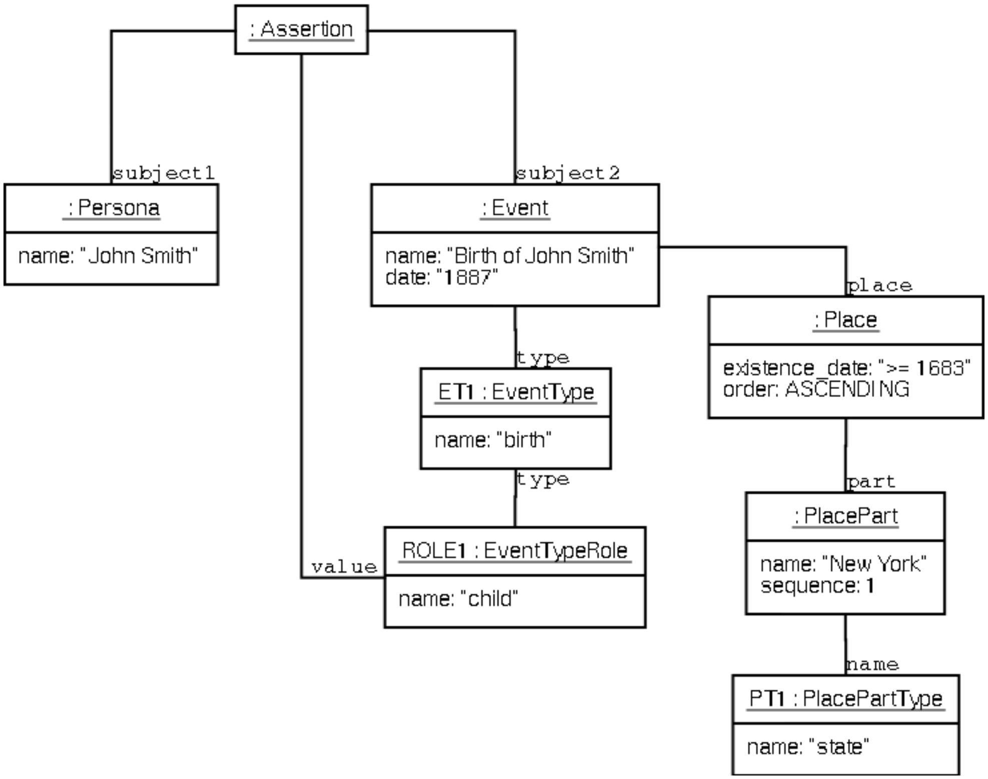

Diagram 3.6.1: An Assertion exemplifying 3.1.2.

The objects ET1, ROLE1, and PT1 are "standard" objects which can be reused. All other objects are created at the time of the assertion. Note that the program should be savvy enough to understand that a date of "1887" means any time in 1887. It would probably be useful to have some kind of starting library of Places, and perhaps a repository of Places accessible on the Internet (such as JewishGen 's "Shtetl Seeker ").

3.6.2. Adam Smith was born 3 months before John Smith.

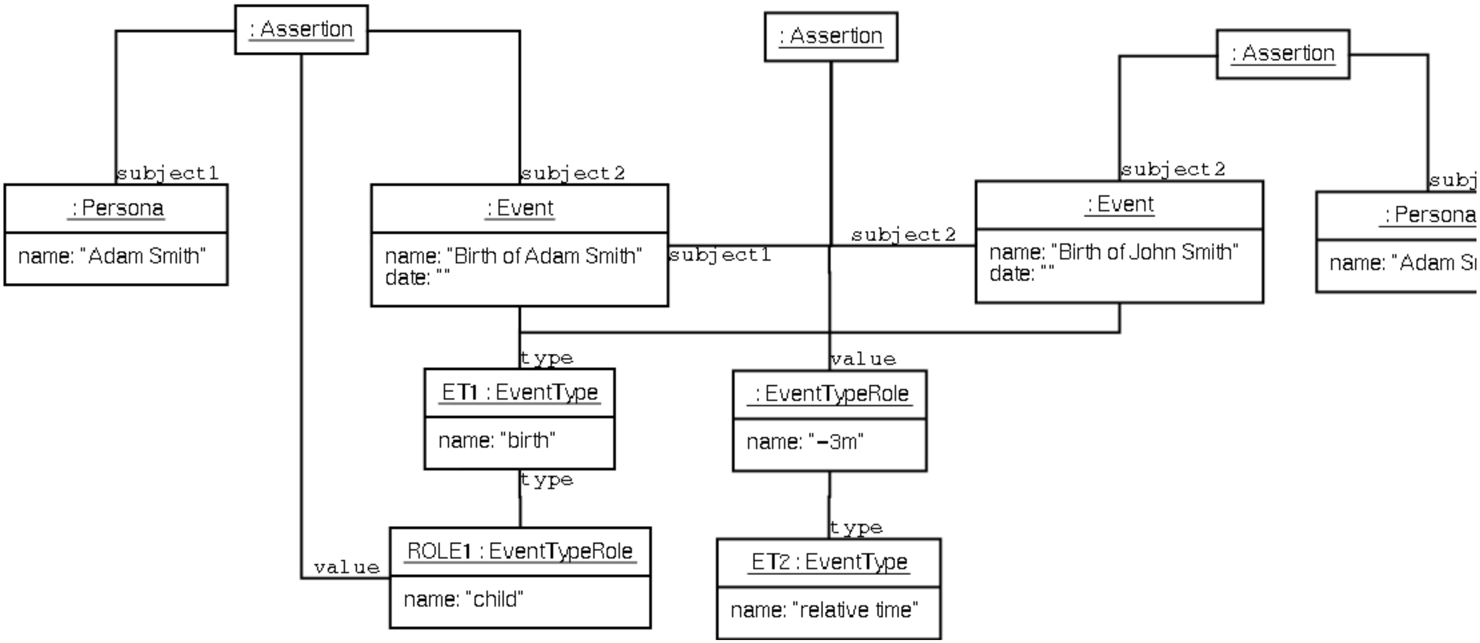

Diagram 3.6.2: An Assertion exemplifying 3.2.2.

The key feature of this diagram is the assertion that one event took place three months prior to another event. The program should know that the relative time could be anywhere from 3 months 0 days to 3 months 31 days.

3.6.3. The John Smith from 3.6.1 is the same as the John Smith from 3.6.2.

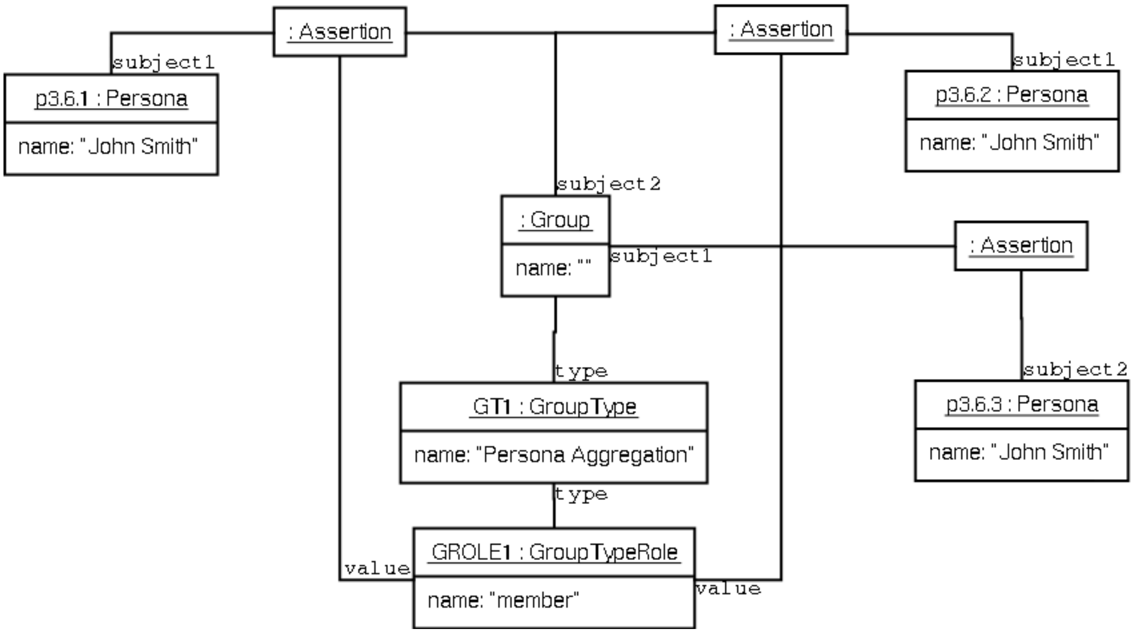

Diagram 3.6.3: An Assertion exemplifying 3.4.1.

The p3.6.1 and p3.6.2 Persona objects are the same objects from 3.6.1 and 3.6.2.

4. Evidence

As stated in section 2, one document = one Persona. The GENTECH data model provides for the representation of documents or, more generally, sources of information.

To show where a given Assertion came from, we can connect it to a Source. Sources also have hierarchies. For example, a book is composed of pages, and pages are composed of lines. If it is important to cite different pages in a book, or different lines in a page, then there can be a book-level Source describing the publication information of the book, page-level Sources referring to individual page numbers (or ranges of page numbers) within the book, and line-level Sources referring to individual lines (or ranges of lines) within a page.

If it isn't important to cite different pages or lines in a book, as when you will probably only pick one page or one line out of a given book, then there need be no hierarchy for that book. A single Source representation the publication information, page range, and line range, suffices.

Sources are not necessarily books. They could be personal interviews, images, or any source of information.

It is not only important to cite a source of information properly, but it is important to record where that source of information exists, so that other researchers can follow your path. Thus, Sources exist in Repositories. A Repository could be a library, or someone's personal possessions.

4.1. Example

An example should illustrate the above more clearly. Suppose the assertion from example 3.6.1 comes from a book called, "Smiths in America", by Hokey Pokey, published by Jenny Holojii Co., ISBN 0880090011, page 16. The book is owned by the researcher.

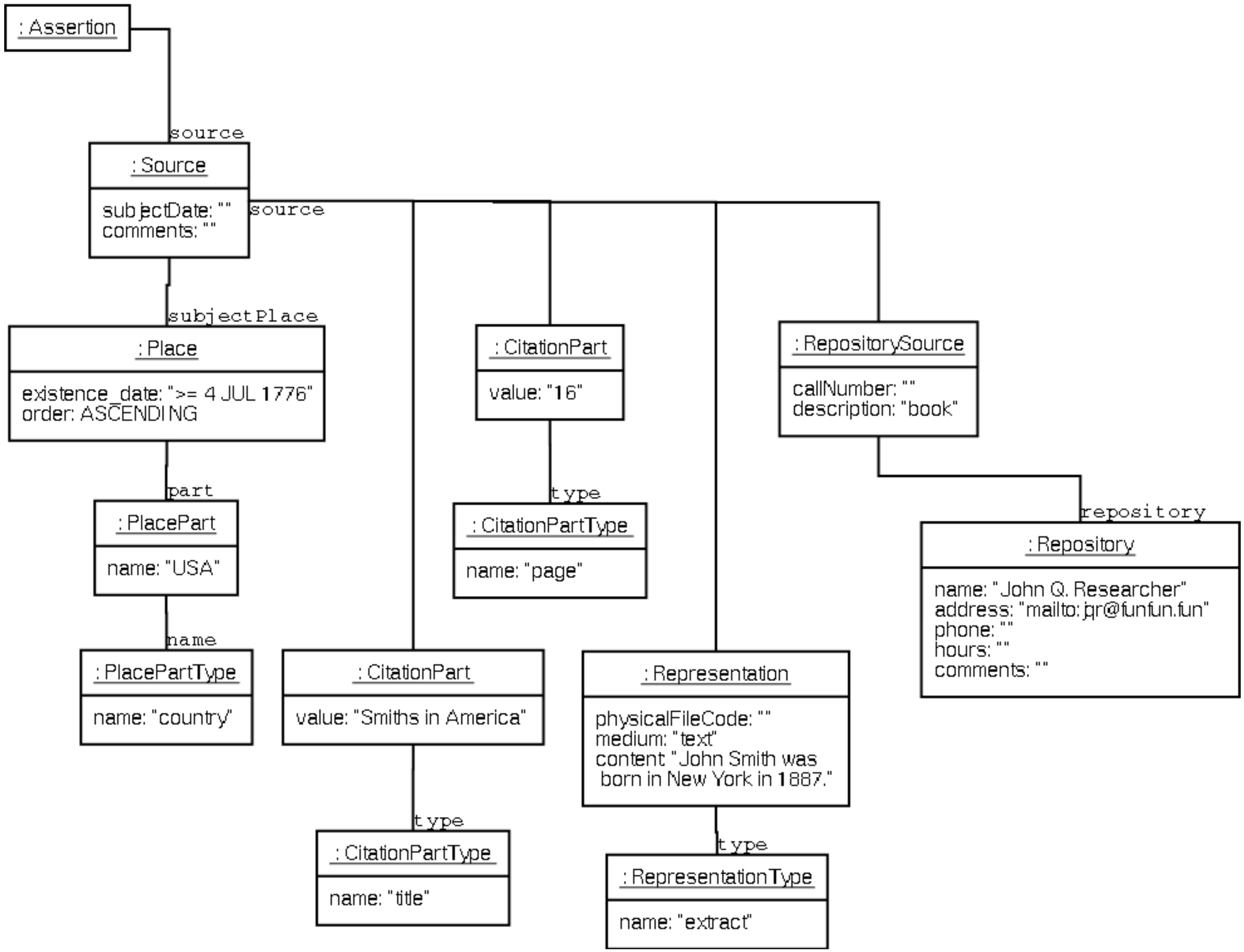

Diagram 4.1: A Source and some of its "decorations"

I haven't included all of the CitationParts in this diagram, otherwise it would get too cluttered. The other parts would be things like publisher, year of publication, ISBN number, author, and so on.

The Source is owned by John Q. Researcher, as shown by the RepositorySource link. In the case of an Internet-accessible repository (which should normally be backed up by a non-Internet repository, given the transient nature of content on the Internet) the Repository's address would be the top-level URL, and the RepositorySource's callNumber would be the URL relative to the top-level URL.

The Representation is where you can attach files, use extracts, or refer to physical files in your desk drawer (does anyone really still use physical files? JOKE!).

5. Reasoning

Assertions have a field called "rationale", in which you can put your reasons for making the Assertion. Generally speaking, the lowest-level Assertions don't nead a rationale, since they are simply representations of information from a Source.

However, when you combine Assertions into a new Assertion, or an Assertion doesn't come from a Source, you should provide your reasoning.

5.1 Assertions without Sources

For example, the Assertions in 3.6.3 are something the researcher asserted. The Assertion adding p3.6.1 probably wouldn't need a rationale, since it could be the first Persona added to the Group. However, the Assertion adding p3.6.2 could use an explanation of the researcher's reasoning for thinking that p3.6.2 is the same as p3.6.1. The Assertion equating a Persona to the Group probably wouldn't need a rationale, since it is an "administrative" function. The reason we would want to put rationales in the Persona-to-Group Assertions and not the Group-to-Persona Assertion is that any one of the Persona-to-Group Assertions could be proven wrong in the future, but the rest of the Persona-to-Group Assertions would not be affected.

5.2 Conclusional Assertions

By building an Assertion upon other Assertions, we can draw conclusions. For example, if one Assertion says Adam Smith was born 3 months before John Smith, and another Assertion says John Smith was born in 1887, we can conclude (via a new Assertion) that Adam Smith was born between 1 SEP 1886 and 30 SEP 1887. The rationale for this would be something like, "date range calculation".

5.3 Contradictory Assertions

Suppose we find, after concluding that Adam Smith was born between 1 SEP 1886 and 30 SEP 1887, a birth certificate for Adam Smith, which states he was born on 20 AUG 1886. That's pretty close to our range, but nevertheless out of our range. Although our conclusion was correct given the evidence, one or both pieces of evidence must be incorrect -- either Adam Smith was not born 3 months before John Smith, or John Smith was not born in 1887, or both. Nevertheless, it is not a good idea to erase incorrect evidence, because that evidence may still be useful, and will prevent future researchers from going down the wrong path.

Assertions have a "disproved" field, which signals whether an Assertion has been shown to be incorrect. We would first mark our conclusion as disproved. This is a signal that one or more of its supporting Assertions are incorrect, but we don't know yet which ones. Also, we would have to add the birth certificate Assertion to the list of lower-level Assertions maintained by the conclusional Assertion, and add to the rationale that we really, really believe the birth certificate.