PoolParty's prerequisites for imported SKOS data

PoolParty is an editor for SKOS vocabularies; it is not an editor for generic RDF. That means:

- Vocabularies created using the PoolParty editor use SKOS in a particular way.
- If you have existing RDF and wish to import it into a PoolParty project, that RDF must satisfy certain prerequisites.

The SWC web site has some background information on how PoolParty uses SKOS:

https://help.poolparty.biz/pp/poolparty-overview/poolparty-skos-rdf-and-uris

More specifically, here are some details about what must be included in a SKOS file that is to be imported into PoolParty:

https://help.poolparty.biz/pp/user-guide-for-knowledge-engineers/basic-features/import-export-and-reporting-with-poolparty/poolparty-rdf-import-export/prerequisites-for-importing-non-poolparty-skos-thesauri-or-zthes-thesauri

You have a couple of options for checking your vocabulary data, before or during the import process. Please use at least one of these:

- SWC provide a SKOS validity checker: http://qskos.poolparty.biz/.
- PoolParty has an Import Assistant, which is enabled by default. When you import vocabulary data with the Import Assistant enabled, if there are errors, the import does not go ahead immediately, and you have several options for proceeding. Depending on the type of problem, you may be given an option of fixing the problem interactively rather than having to cancel the import..

ARDC staff have experience in working with RDF imported into PoolParty. Here are some practical observations based on this experience:

- PoolParty relies on you following the SKOS integrity constraints as spelled out in the SKOS Reference (https://www.w3.org/TR/skos-reference/). If you violate the integrity constraints, all bets are off. You can break a project by importing bad SKOS. For example, if you import RDF that defines a resource as *both* a `skos:Collection` and a `skos:ConceptScheme` (contra SKOS integrity constraint S37), PoolParty will report an "internal server error" and the project will break.
- Use of the `skos:hasTopConcept` property is necessary and sufficient to link a concept into the hierarchy at the top level. That is, the `skos:topConceptOf` property plays no role in determining what PoolParty considers to be a top concept.
    - This has been tested by importing some RDF that defines a concept and specifies `skos:topConceptOf` the concept scheme. It *did not* appear in the hierarchy. Importing more RDF that specifies that the concept scheme `skos:hasTopConcept` the concept causes the concept to appear in the hierarchy.
    - PoolParty does not add the `skos:hasTopConcept` property for imported top concepts. If you wish an imported concept to have the `skos:hasTopConcept` property, this must also be included in the imported data.
    - PoolParty does not add the `skos:topConceptOf` property for imported top concepts. If you wish an imported concept to have the `skos:topConceptOf` property, this must also be included in the imported data.
    - A top concept created within the PoolParty user interface is linked to the concept scheme with *both* the `skos:hasTopConcept` and `skos:topConceptOf` properties.
- Use of a chain of `skos:narrower` properties from a top concept down to a concept is necessary and sufficient to link such a non-top concept into the hierarchy. That is, the `skos:broader` property plays no role in determining what PoolParty considers to be a broader concept.
    - This has been tested by importing some RDF that defines a concept and specifies `skos:broader` to a top concept that is already visible in the user interface. The new concept *did not* appear in the hierarchy. Importing more RDF that specifies that the existing top concept has `skos:narrower` the new concept causes the new concept to appear in the hierarchy.
    - In another test, some RDF was imported that defines a concept and specifies `skos:narrower` from a top concept that is already visible in the user interface, but does not include the corresponding `skos:broader` property. The new concept *did* appear in the hierarchy.
    - PoolParty does not add the `skos:narrower` property for imported concepts that are otherwise linked to another concept using `skos:broader`. If you wish an imported concept to have the `skos:narrower` property, this must also be included in the imported data.
    - PoolParty does not add the `skos:broader` property for imported concepts that are otherwise linked to another concept using `skos:narrower`. If you wish an imported concept to have the `skos:broader` property, this must also be included in the imported data.
    - A non-top concept created within the PoolParty user interface is linked to its parent concept with *both* the `skos:broader` and `skos:narrower` properties.
- SKOS Collections do work. In PoolParty, Collections have a `dcterms:title` property for their title, and need it so that they show up in the Collections list. So before importing, you need to add an extra triple for each Collection so that PoolParty can cope with it. If you import existing vocabulary data without having added these triples, the Collections list shows a table with blank titles, which makes it unusable.