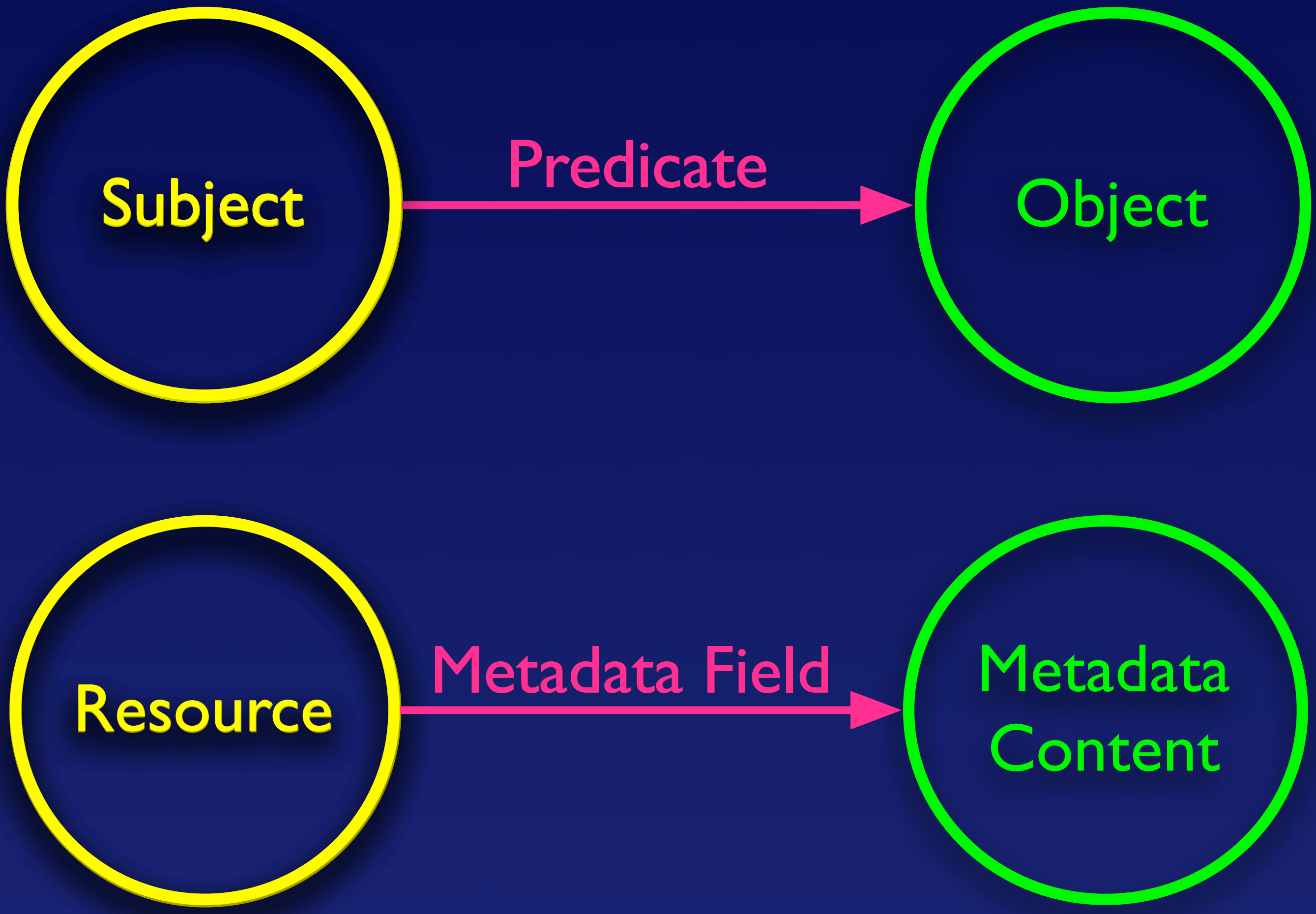
Using RDF in a linguistics archive

Jeff Good, MPI EVA (good@eva.mpg.de)

RDF Introduction

- What is RDF?
  - Gory detail: It's a W3C specification and one of the core technologies of the Semantic web
  - At heart it's a data *model* not a format
  - The model takes the form of *triples* which join together to form a *graph*
  - There are some additional restrictions to make things web-friendly

RDF Triples


Subject
Predicate
Object
Resource
Metadata Field
Metadata Content

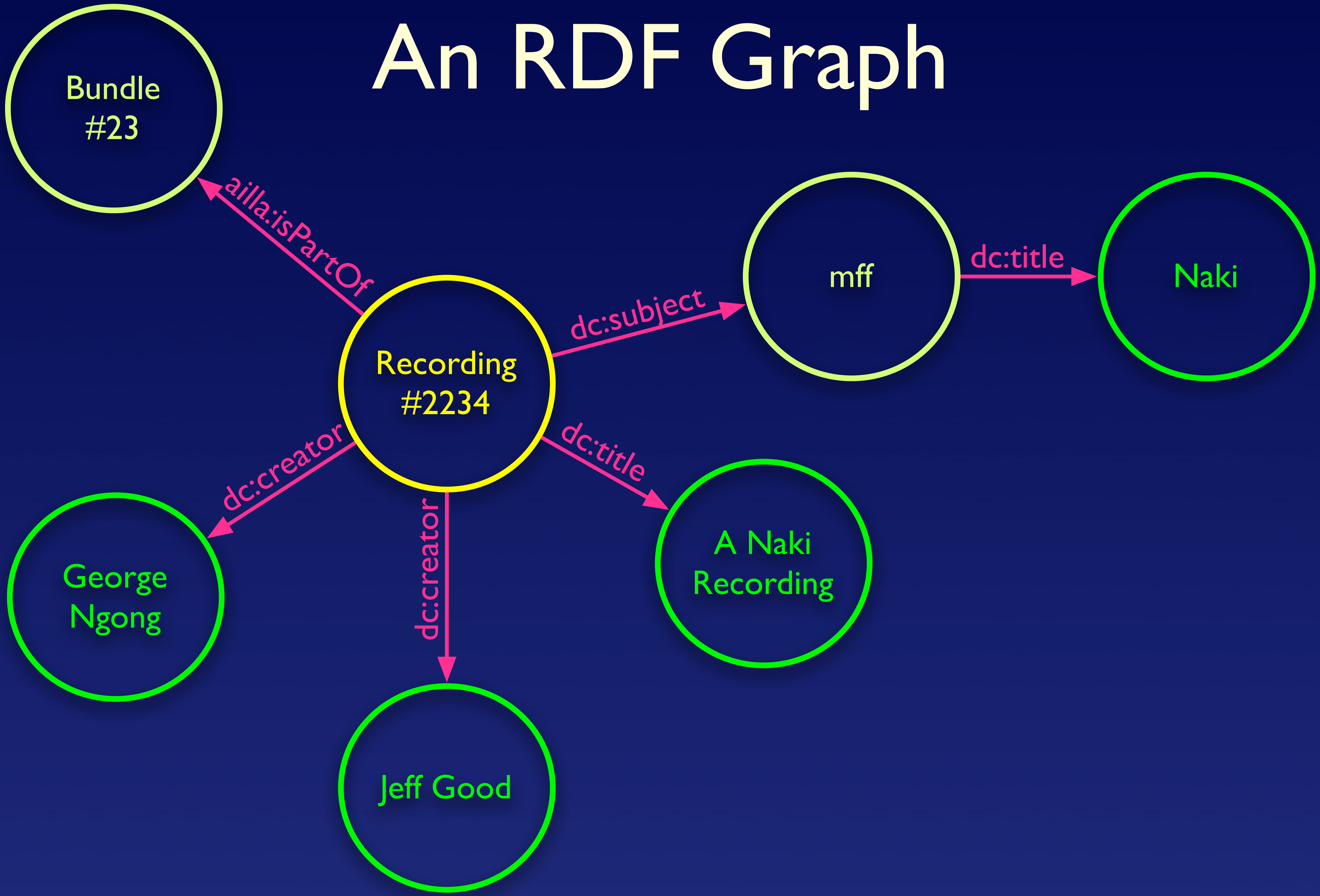
RDF Triples

- There shouldn’t be anything conceptually new to triples
- Most metadata schemes are already based on conceptual models which use triples or something like them
- What’s important about RDF is it explicitly uses triples as its model—as opposed to, say, vanilla Dublin Core which does not

RDF Triples

- Obviously, one triple on its own isn't very interesting
- But, triples can become a fairly powerful way to encode information when combined together

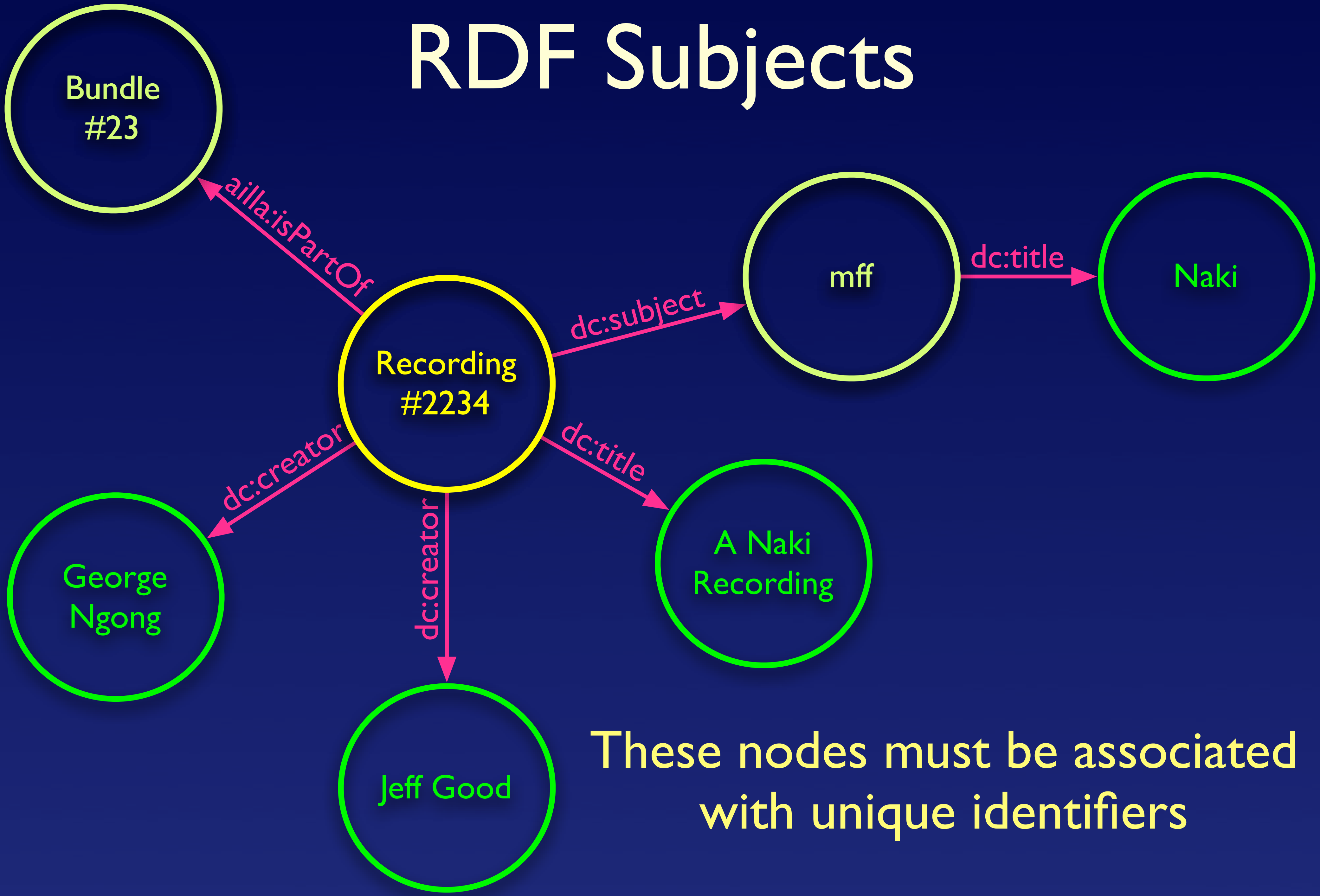
An RDF Graph


Bundle #23
ailla:isPartOf
Recording #2234
dc:subject
mff
dc:title
Naki
dc:creator
George Ngong
dc:creator
Jeff Good
dc:title
A Naki Recording

RDF Identifiers

- A key ingredient of RDF is restrictions relating to unique identifiers
  - Any node in the graph serving as the subject of a predicate must have a unique identifier
  - Nodes only serving as objects can be text strings
  - Unique identifiers must take the form of a URI (i.e., look like a web address)

RDF Subjects

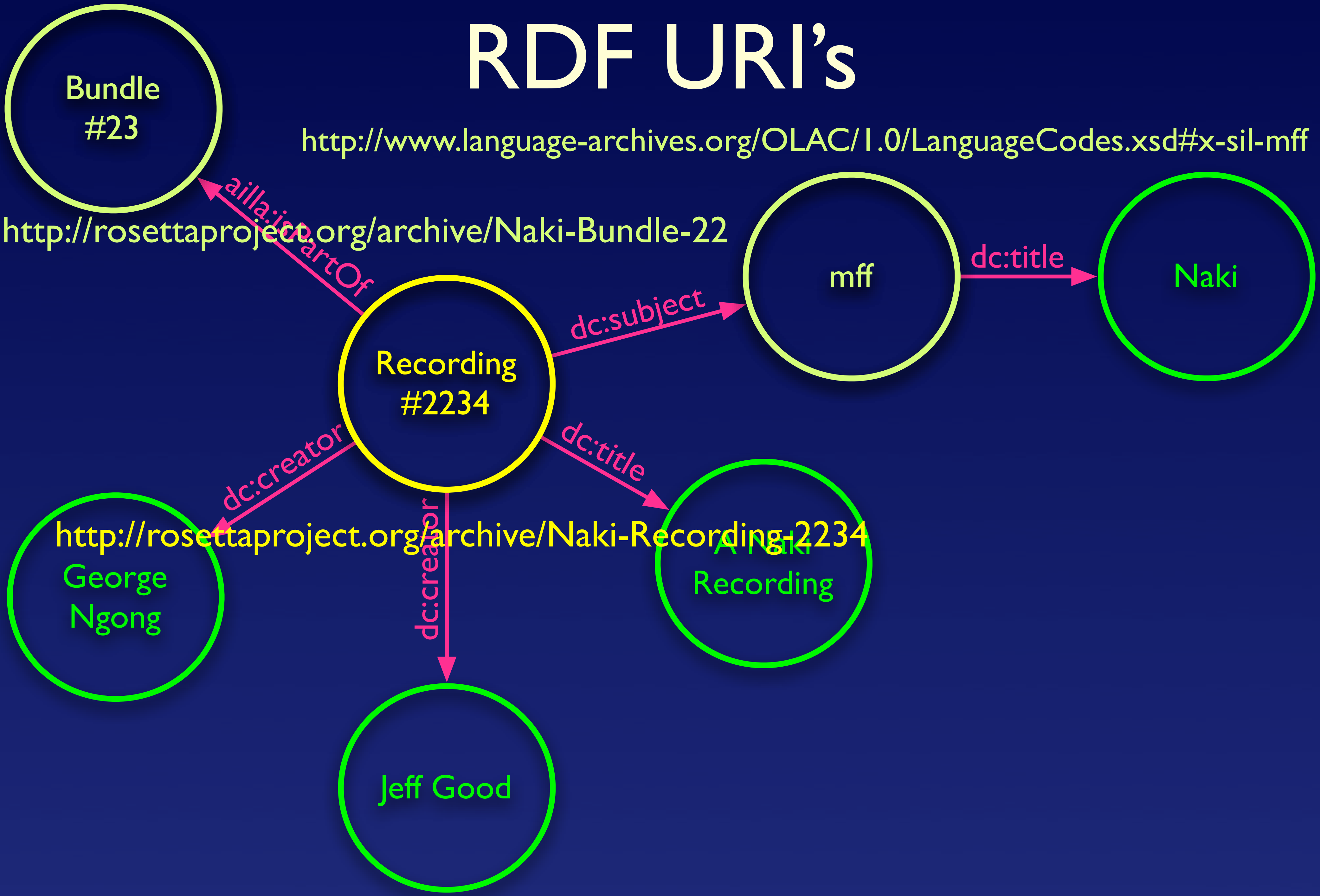


These nodes must be associated with unique identifiers

RDF Identifiers

- Why URI's
  - They are familiar and web-friendly
  - They are convenient for an "open" universal ID system—that is, unlike a traditional database ID, the idea is that a URI ID will be made publicly known
  - This will allow others to say things about your resources in a machine-readable way without coordination

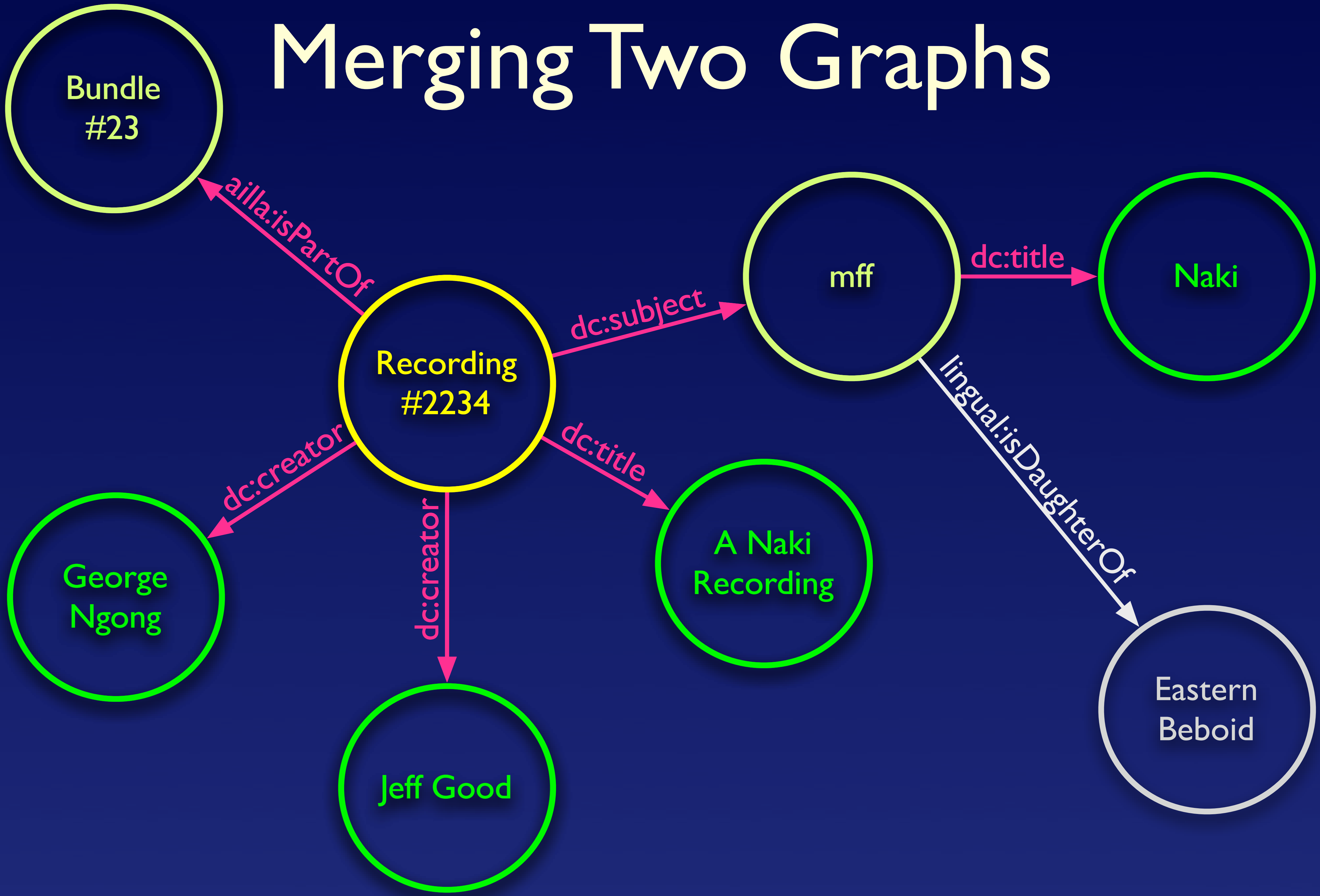
RDF URI's


Bundle #23
http://www.language-archives.org/OLAC/1.0/LanguageCodes.xsd#x-sil-mff
ailla:isPartOf
http://rosettaproject.org/archive/Naki-Bundle-22
mff
dc:title
Naki
dc:subject
Recording #2234
dc:creator
dc:title
http://rosettaproject.org/archive/Naki-Recording-2234
George Ngong
dc:creator
A Naki Recording
Jeff Good

RDF Identifiers

- An envisioned scenario
  - An archive publishes its RDF metadata
  - A linguist using a resource from that archive uses the URI's in the resource metadata to add new information to the original resource metadata
  - Another linguist "merges" the two sets of data into one big informative RDF graph

Merging Two Graphs



Rosetta Background

- Rosetta did not start to use RDF because of metadata management problems
- It used it to encode genetic and geographic relationships among language
- These sorts of relationships, involving "trees", are much more easily formalized using graphs than relational tables
- Therefore, RDF was a better solution than a relational database

Rosetta Background

- But, we knew that RDF was good practice and that there was an RDF Dublin Core specification
- So, given that we were developing RDF tools anyway, it made sense to use RDF for our metadata as well

Technical Details

- As a W3C standard, there are well-defined specifications for encoding RDF
- An important one of these is RDF/XML, an XML way of representing RDF
- But there are others, as well, including the relatively compact N3 format
- At Rosetta, we use a Python library for RDF processing. There are also Java and Perl libraries (and maybe others)

RDF Pluses

- A good RDF library abstracts away from things like XML formatting details
- All you have to think about is your triples and their URI's—the tools take care of creating the archival output
- Having used RDF tools, *I never want to go back to XML again*
- Why think in terms of a format, when you can think in terms of a data structure?

RDF Pluses

- You get for free
  - Interoperability with others using your namespaces and URI schemes
  - The possibility to integrate with ontologies
  - A best-practice data model which can be expressed in an archival format
  - Interoperability with yourself

A Rosetta Application

- At Rosetta, our website runs off of a database optimized for web applications
- We want our resource metadata to be stored in that database
- However, our primary resources at present, scanned images, are not stored in a database but rather in the filesystem of our image server

A Rosetta Application

- We have a quandary: We want our metadata in two places
- Old method: The database was the only place where the metadata was and you would query it to get information about a file in the filesystem

A Rosetta Application

- New method
  - We store RDF metadata in a small text file in the same directory as the resource itself
  - We have a script that looks for those metadata files, merges them into one big RDF graph, and outputs a new XML file containing all the metadata
  - This file is then fed into our database

A Rosetta Application

- Our website database could implode, but we'd still have our metadata on the filesystem
- This whole process is simple because RDF was designed with this kind of "merger" in mind
- "XML" is not designed with this in mind

An N3 metadata snippet

```
<http://rosettaproject.org/archive/aaa/vertxt-1>
        a lingbib:ScannedDocument ;
        dc:title "Ghotuo Glossed Text" ;
        dc:creator "Author to be added" ;
        dc:date "Date to be added" ;
        lingbib:refersToLanguoid <http://rosettaproject.org/archive/aaa> ;
        lingbib:rosettaType "vertxt" ;
        lingbib:hasPages "5" ;
        lingbib:imageNamePrefix "aaa-vertxt-1" ;
        lingbib:hasStatus "live" ;
        lingbib:directory "rosettaproject/archive/a/a/aaa/vertxt-1/" .
```

An XML metadata snippet

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  xmlns:lingbib='http://rosettaproject.org/ns/lingbib/1.0/'
  xmlns:dc='http://purl.org/dc/elements/1.1/'
>
  <lingbib:ScannedDocument rdf:about="http://rosettaproject.org/archive/aaa/
vertxt-1">
    <lingbib:imageNamePrefix>aaa-vertxt-1</lingbib:imageNamePrefix>
    <dc:title>Ghotuo Glossed Text</dc:title>
    <dc:creator>Author to be added</dc:creator>
    <lingbib:refersToLanguoid rdf:resource="http://rosettaproject.org/archive/aaa"/>
    <lingbib:rosettaType>vertxt</lingbib:rosettaType>
    <dc:date>Date to be added</dc:date>
    <lingbib:directory>rosettaproject/archive/a/a/aaa/vertxt-1/</lingbib:directory>
    <lingbib:hasPages>5</lingbib:hasPages>
    <lingbib:hasStatus>live</lingbib:hasStatus>
  </lingbib:ScannedDocument>
</rdf:RDF>
```

The Aggregation Script

```
from rdflib.Graph import Graph
from rdflib.Namespace import Namespace

store = Graph()

import os.path
for root, dirs, files in os.walk('rosettaproject/archive'):
    for file in files:
         if file == "MyMetadata.n3":
            store.load(root+"/"+file, format="n3")

resourceMDfile = open("resources.rdf", "w")
resourceMDfile.write(store.serialize())
```

RDF considerations

- It's still a pretty new technology. So, support is limited.
- Rosetta has a custom-built system for working with its data.
- That being said, I found it to be pretty easy to learn to use the basic tools.
- ...much easier than XML