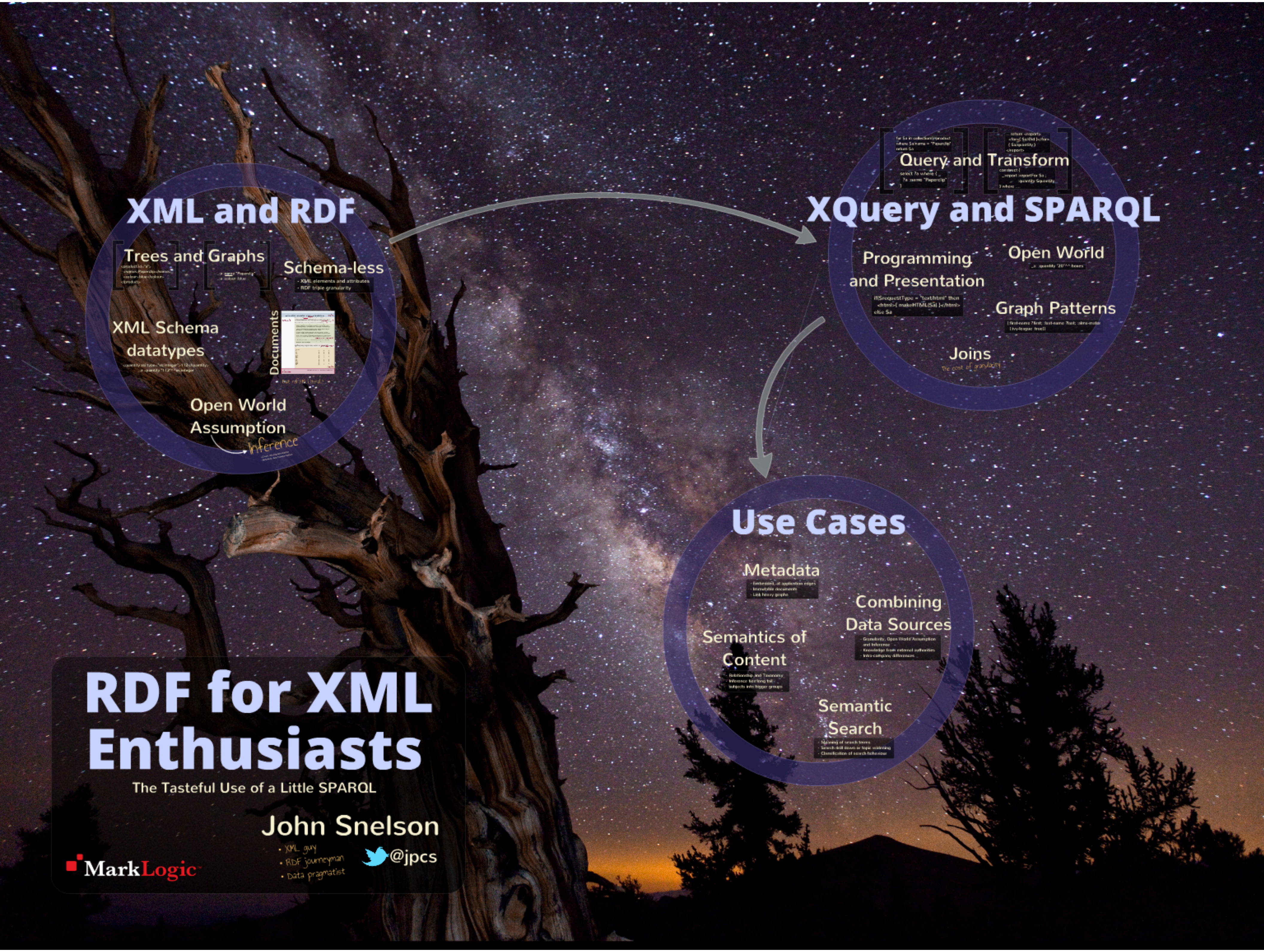
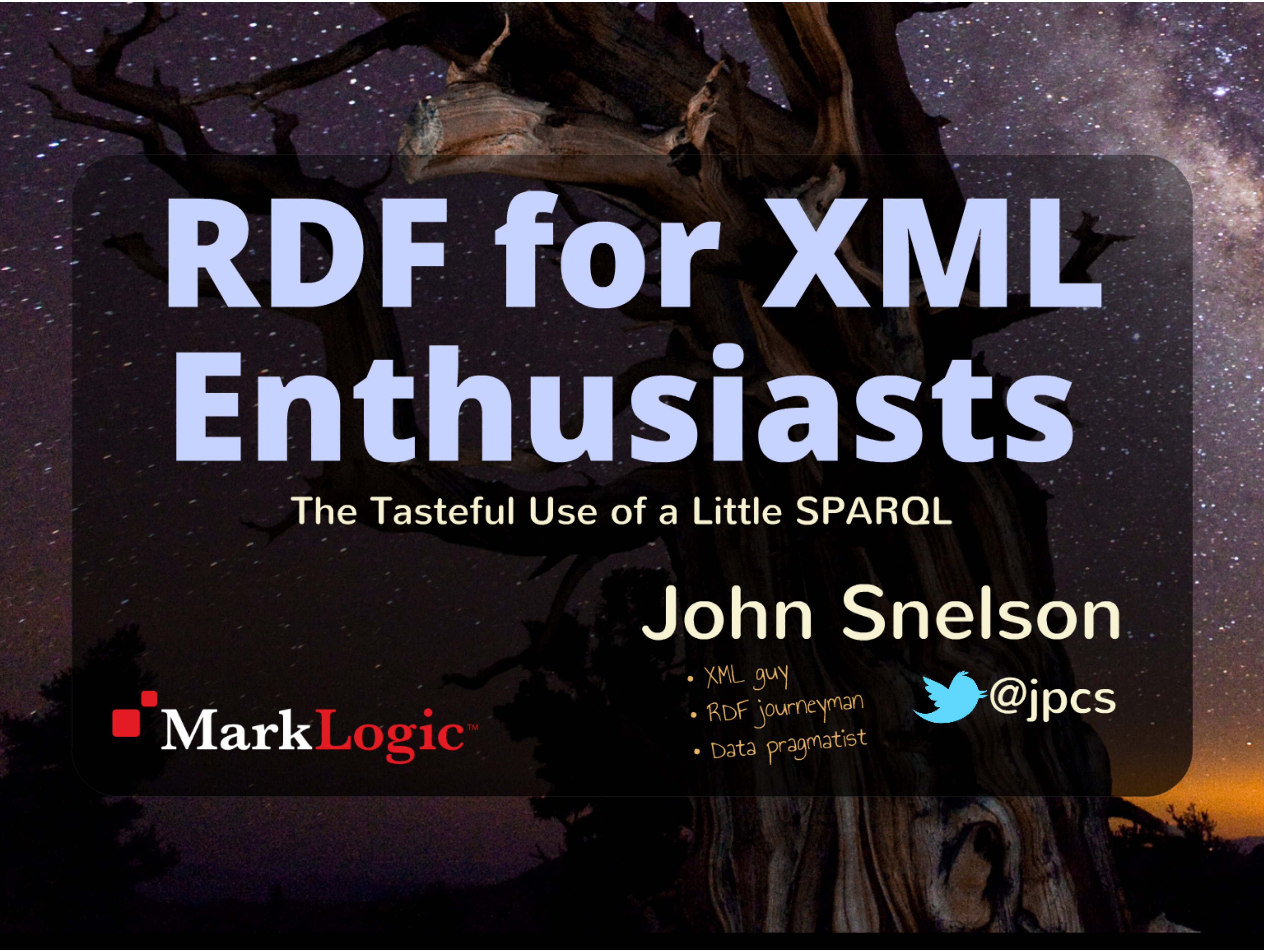

XML and RDF
Trees and Graphs
Schema-less
XML Schema datatypes
Documents
Open World Assumption
Inference
XQuery and SPARQL
Query and Transform
Programming and Presentation
Open World
Graph Patterns
Joins
Use Cases
Metadata
Combining Data Sources
Semantics of Content
Semantic Search
RDF for XML Enthusiasts
The Tasteful Use of a Little SPARQL
John Snelson
• XML guy
• RDF journeyman
• Data pragmatist
@jpcs
MarkLogic



RDF for XML Enthusiasts

The Tasteful Use of a Little SPARQL

John Snelson

- XML guy
- RDF journeyman
- Data pragmatist

@jpcs

MarkLogic™

f a Little SPARQL

John Snelson

- XML guy
- RDF journeyman
- Data pragmatist

@jpcs

RDF for XML Enthusiasts

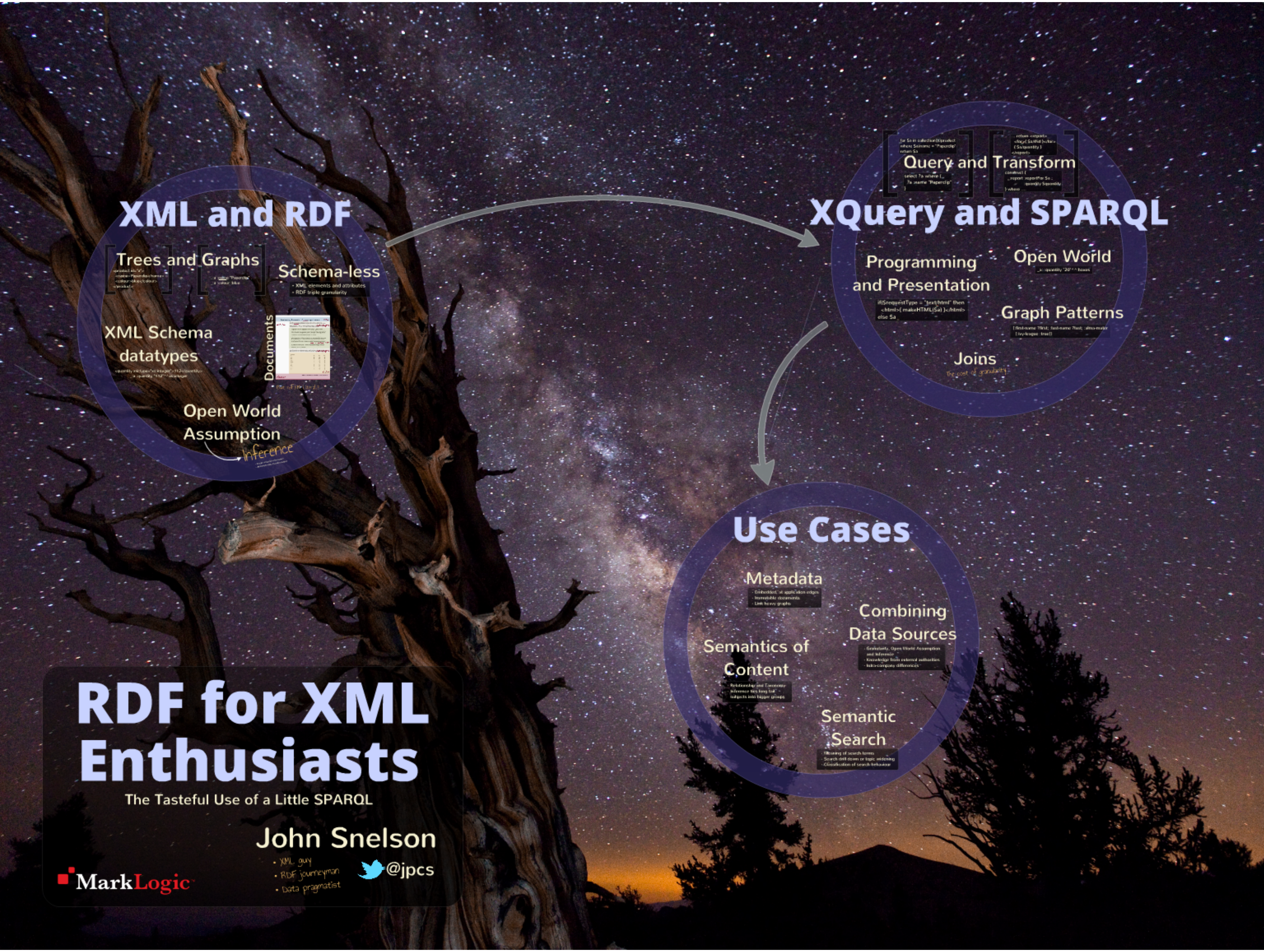
The Tasteful Use of a Little SPARQL

John Snelson

- XML guy
- RDF journeyman
- Data pragmatist

@jpcs

MarkLogic™



XML and RDF
Trees and Graphs
Schema-less
• XML elements and attributes
• RDF triple granularity
XML Schema datatypes
Documents
Open World Assumption
Inference
Query and Transform
XQuery and SPARQL
Programming and Presentation
Open World
Graph Patterns
Joins
Use Cases
Metadata
Combining Data Sources
Semantics of Content
Semantic Search
RDF for XML Enthusiasts
The Tasteful Use of a Little SPARQL
John Snelson
• XML guy
• RDF journeyman
• Data pragmatist
@jpcs
MarkLogic

XML and RDF

Trees and Graphs

```
<product id="a">
  <name>Paperclip</name>
  <colour>blue</colour>
</product>
```

```
_:a :name "Paperclip" .
_:a :colour :blue
```

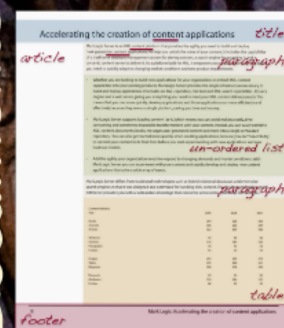
Schema-less

- XML elements and attributes
- RDF triple granularity

XML Schema datatypes

```
<quantity xsi:type="xs:integer">112</quantity>
_:a :quantity "112"^^xs:integer
```

Documents

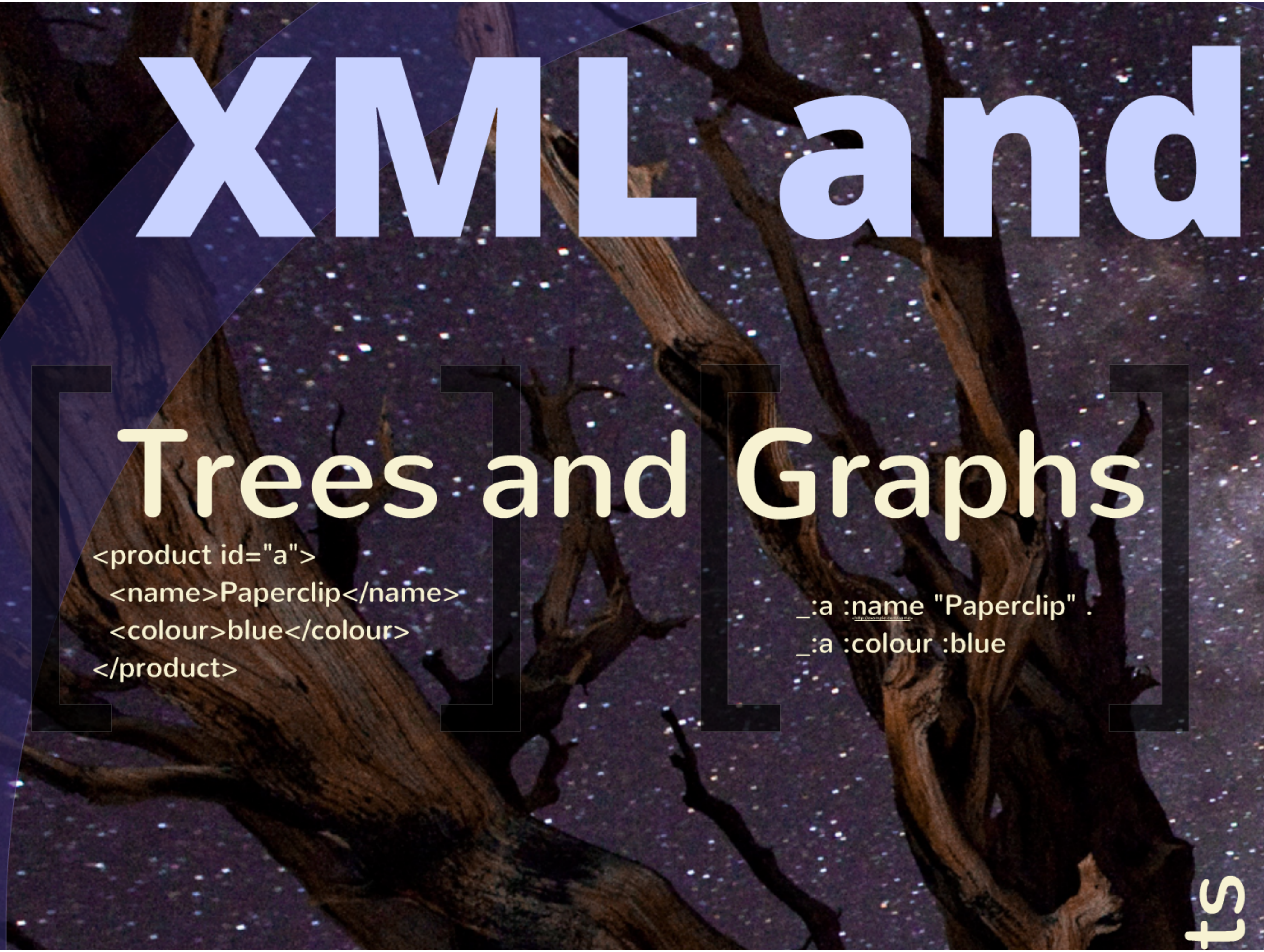


But rdf:XMLLiteral..?

Open World Assumption

Inference

- Create missing information
- Automate data transformation



XML and

Trees and Graphs

```
<product id="a">
  <name>Paperclip</name>
  <colour>blue</colour>
</product>
```

```
_:a :name "Paperclip" .
     <http://example.com/name>
_:a :colour :blue
```

ts

Trees a

```
<product id="a">
  <name>Paperclip</name>
  <colour>blue</colour>
</product>
```

Graphs

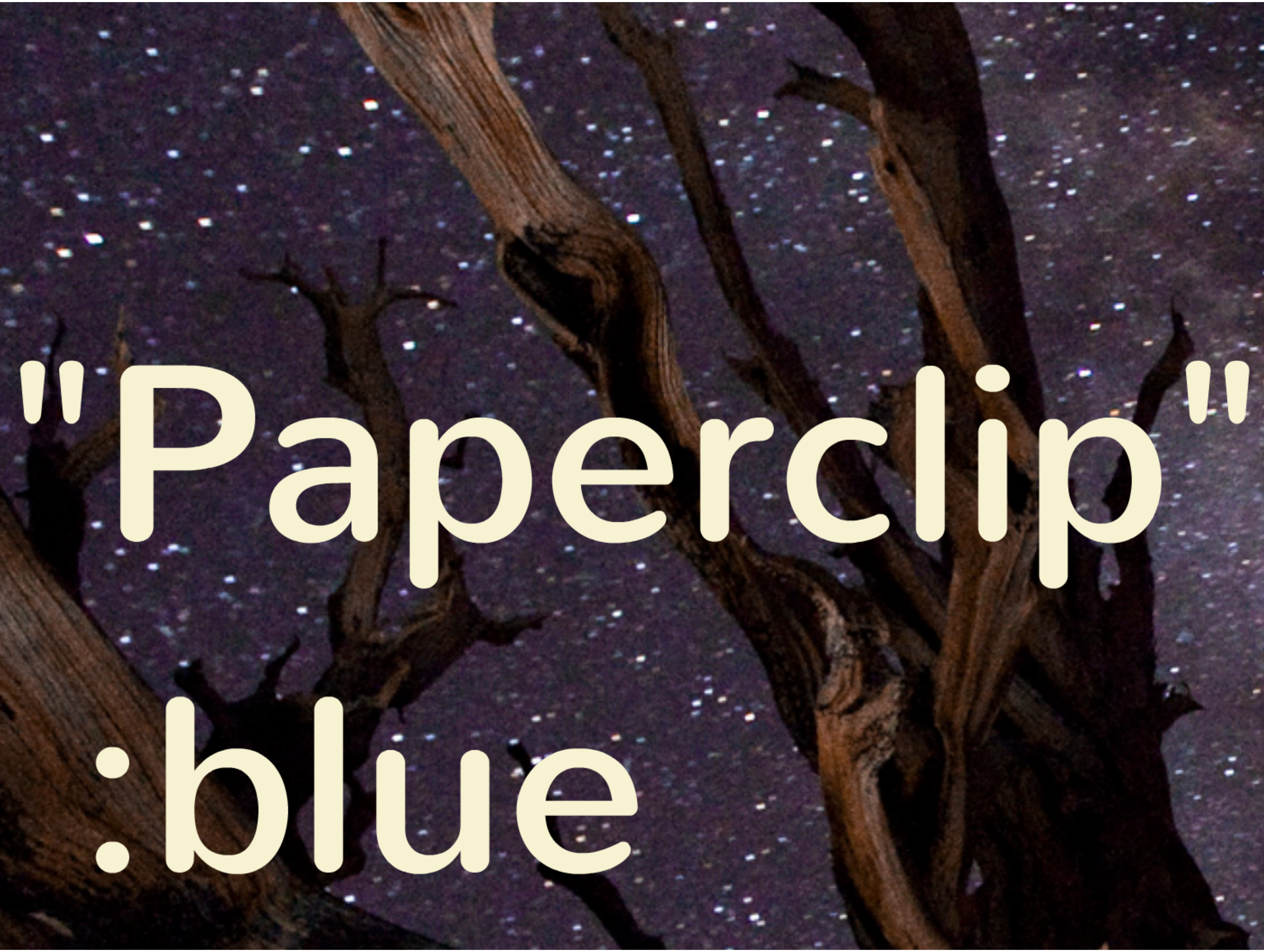
```
_:a :name "Paperclip" .
```

<http://example.com/name>

```
_:a :colour :blue
```


_:a


:name
<http://example.com/name>


"Paperclip"
:blue

Graphs

_:a :name "Paperclip" .

<http://example.com/name>

_:a :colour :blue

XML and

Trees and Graphs

```
<product id="a">
  <name>Paperclip</name>
  <colour>blue</colour>
</product>
```

```
_:a :name "Paperclip" .
     <http://example.com/name>
_:a :colour :blue
```

ts

XML Schema datatypes

<quantity xsi:type="xs:integer">112</quantity>

_:a :quantity "112"^^xs:integer

xs:integer">112</qu

"112"^^xs:integer

XML Schema datatypes

<quantity xsi:type="xs:integer">112</quantity>

_:a :quantity "112"^^xs:integer

Schema-less

- XML elements and attributes
- RDF triple granularity

Documents

Accelerating the creation of content applications

title

article

MarkLogic Server is an XML content platform that provides the agility you need to build and deploy next-generation content applications. To help you unlock the value of your content, it includes the capabilities of a traditional database management system for storing content, a search engine for accessing it, and a dynamic content server to deliver it. As a platform built for XML, it empowers your organization with the agility you need to quickly adapt to changing market conditions and new product requirements.

paragraph

- Whether you are looking to build new applications for your organization or embed XML content capabilities into your existing products, MarkLogic Server provides the single infrastructure necessary to build and deploy applications. It includes an XML repository , full-text and XML search capabilities , XQuery engine and a web server, giving you everything you need to meet your XML content delivery needs. This means that you can more quickly develop applications, and those applications run more efficiently and effectively because they are on a single platform, saving you time and money.
- MarkLogic Server supports loading content "as-is", which means you can avoid making costly, time consuming and sometimes impossible transformations with your content. Instead, you can easily combine XML content, documents, books, messages, user generated content and more into a single centralized repository . You can also get started more quickly when creating applications, because you don't need to try to convert your content to its final form before you start experimenting with new applications and new business models.
- Add the agility your organization needs to respond to changing demands and market conditions. With MarkLogic Server you can experiment with your content and rapidly develop and deploy new content applications that solve a wide array of needs.

un-ordered list

MarkLogic Server differs from traditional technologies such as hybrid-relational databases and enterprise search engines in that it was designed and optimized for handling XML content. This fundamental architectural difference provides you with a sustainable advantage that cannot be achieved by traditional technologies.

paragraph

Content statistics

Year	2005	2006	2007
Books	384	308	390
Journals	305	200	234
Articles	923	839	990
Abstracts	45	56	60
Sections	110	116	125
Paragraphs	49	50	54
Footers	55	70	76
Images	203	256	319
Tables	495	488	501
Diagrams	368	378	390
Elements	69	74	85
Attributes	110	124	135
Entities	88	90	95

table

footer

Docum

XML content, documents, books, messages, user generated content and more into a single centralized repository . You can also get started more quickly when creating applications, because you don't need to try to convert your content to its final form before you start experimenting with new applications and new business models.

un-ordered list

- Add the agility your organization needs to respond to changing demands and market conditions. With MarkLogic Server you can experiment with your content and rapidly develop and deploy new content applications that solve a wide array of needs.

MarkLogic Server differs from traditional technologies such as hybrid-relational databases and enterprise search engines in that it was designed and optimized for handling XML content. This fundamental architectural difference provides you with a sustainable advantage that cannot be achieved by traditional technologies.

paragraph

Content statistics

Year	2005	2006	2007
Books	384	308	390
Journals	305	200	234
Articles	923	839	990
Abstracts	45	56	60
Sections	110	116	125
Paragraphs	49	50	54
Footers	55	70	76
Images	203	256	319
Tables	495	488	501
Diagrams	368	378	390
Elements	69	74	85
Attributes	110	124	135
Entities	88	90	95

table

footer

But rdf:XMLLiteral..?

^^xs:integer

But rdf:X

Open World Assumption

Inference

- Create missing information
- Automate data transformation

ption

Inference

- Create missing information
- Automate data transformation

XML and RDF

Trees and Graphs

```
<product id="a">
  <name>Paperclip</name>
  <colour>blue</colour>
</product>
```

```
_:a :name "Paperclip" .
_:a :colour :blue
```

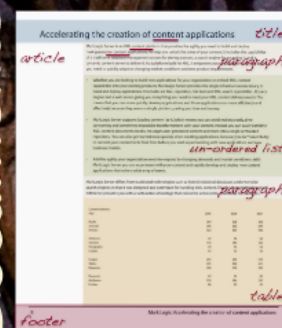
Schema-less

- XML elements and attributes
- RDF triple granularity

XML Schema datatypes

```
<quantity xsi:type="xs:integer">112</quantity>
    _:a :quantity "112"^^xs:integer
```

Documents



But rdf:XMLLiteral..?

Open World Assumption

Inference

- Create missing information
- Automate data transformation

Query and Transform

```
for $a in collection()//product
where $a/name = "Paperclip"
return $a
```

```
... return <report>
  <for>{ $a/@id }</for>
  { $a/quantity }
</report>
```

```
select ?a where {
  ?a :name "Paperclip"
}
```

```
construct {
  _:report :reportFor $a ;
           :quantity $quantity
} where ....
```

XQuery and SPARQL

Programming and Presentation

```
if($requestType = "text/html" then
  <html>{ makeHTML($a) }</html>
else $a
```

Open World

```
_a: :quantity "20"^^:boxes
```

Graph Patterns

```
[:first-name ?first; :last-name ?last; :alma-mater
  [:ivy-league :true]]
```

Joins

The cost of granularity

```
for $a in collection()//product
where $a/name = "Paperclip"
return $a
```

```
... return <report>
  <for>{ $a/@id }</for>
  { $a/quantity }
</report>
```

Query and Transform

```
select ?a where {
  ?a :name "Paperclip"
}
```

```
construct {
  _:report :reportFor $a ;
           :quantity $quantity
} where ....
```

ery and SPA

```
for $a in collection()//product
where $a/name = "Paperclip"
return $a
```

Query an

```
select ?a where {
  ?a :name "Paperclip"
}
```

```
... return <report>
  <for>{ $a/@id }</for>
  { $a/quantity }
</report>
```

Transform

```
construct {
  _:report :reportFor $a ;
           :quantity $quantity
} where ....
```

```
for $a in collection()//product
where $a/name = "Paperclip"
return $a
```

```
... return <report>
  <for>{ $a/@id }</for>
  { $a/quantity }
</report>
```

Query and Transform

```
select ?a where {
  ?a :name "Paperclip"
}
```

```
construct {
  _:report :reportFor $a ;
           :quantity $quantity
} where ....
```

ery and SPA

Query and

Programming and Presentation

```
if($requestType = "text/html" then
  <html>{ makeHTML($a) }</html>
else $a
```

Open World

```
_a: :quantity "20"^^:boxes
```

Open World

```
_a: :quantity "20"^^:boxes
```

Graph Patterns

```
[:first-name ?first; :last-name ?last; :alma-mater
  [:ivy-league :true]]
```

Graph Patterns

```
[:first-name ?first; :last-name ?last; :alma-mater
  [:ivy-league :true]]
```

Joins

The cost of granularity

[:firs

[:iv

```
for $a in collection()//product
where $a/name = "Paperclip"
return $a
```

```
... return <report>
  <for>{ $a/@id }</for>
  { $a/quantity }
</report>
```

Query and Transform

```
select ?a where {
  ?a :name "Paperclip"
}
```

```
construct {
  _:report :reportFor $a ;
           :quantity $quantity
} where ....
```

XQuery and SPARQL

Programming and Presentation

```
if($requestType = "text/html" then
  <html>{ makeHTML($a) }</html>
else $a
```

Open World

```
_a: :quantity "20"^^:boxes
```

Graph Patterns

```
[:first-name ?first; :last-name ?last; :alma-mater
  [:ivy-league :true]]
```

Joins

The cost of granularity

Use Cases

Metadata

- Embedded, at application edges
- Immutable documents
- Link heavy graphs

Combining Data Sources

- Granularity, Open World Assumption and Inference
- Knowledge from external authorities
- Intra-company differences

Semantics of Content

- Relationship and Taxonomy
- Inference ties long tail subjects into bigger groups

Semantic Search

- Meaning of search terms
- Search drill down or topic widening
- Classification of search behaviour

Metadata

- Embedded, at application edges
- Immutable documents
- Link heavy graphs

Combining Data Sources

- Granularity, Open World Assumption and Inference
- Knowledge from external authorities
- Intra-company differences

Semantics of Content

- Relationship and Taxonomy
- Inference ties long tail subjects into bigger groups

Semantic Search

- Meaning of search terms
- Search drill down or topic widening
- Classification of search behaviour

Use Cases

Metadata

- Embedded, at application edges
- Immutable documents
- Link heavy graphs

Combining Data Sources

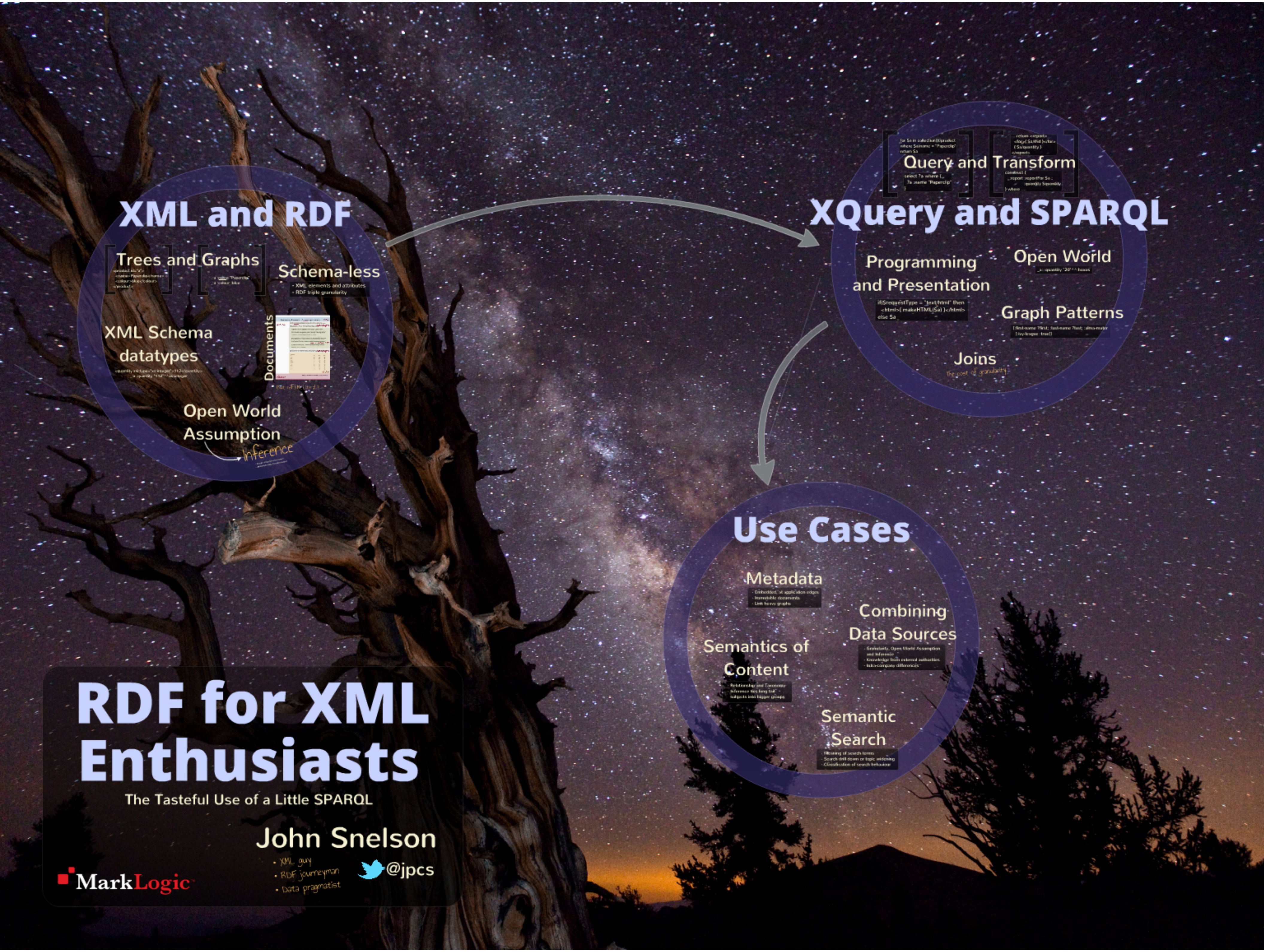
- Granularity, Open World Assumption and Inference
- Knowledge from external authorities
- Intra-company differences

Semantics of Content

- Relationship and Taxonomy
- Inference ties long tail subjects into bigger groups

Semantic Search

- Meaning of search terms
- Search drill down or topic widening
- Classification of search behaviour


XML and RDF
Trees and Graphs
Schema-less
XML Schema datatypes
Documents
Open World Assumption
Inference
Query and Transform
XQuery and SPARQL
Programming and Presentation
Open World
Graph Patterns
Joins
Use Cases
Metadata
Combining Data Sources
Semantics of Content
Semantic Search
RDF for XML Enthusiasts
The Tasteful Use of a Little SPARQL
John Snelson
• XML guy
• RDF journeyman
• Data pragmatist
@jpcs
MarkLogic