

# Adventures in Stateless Catalog Land

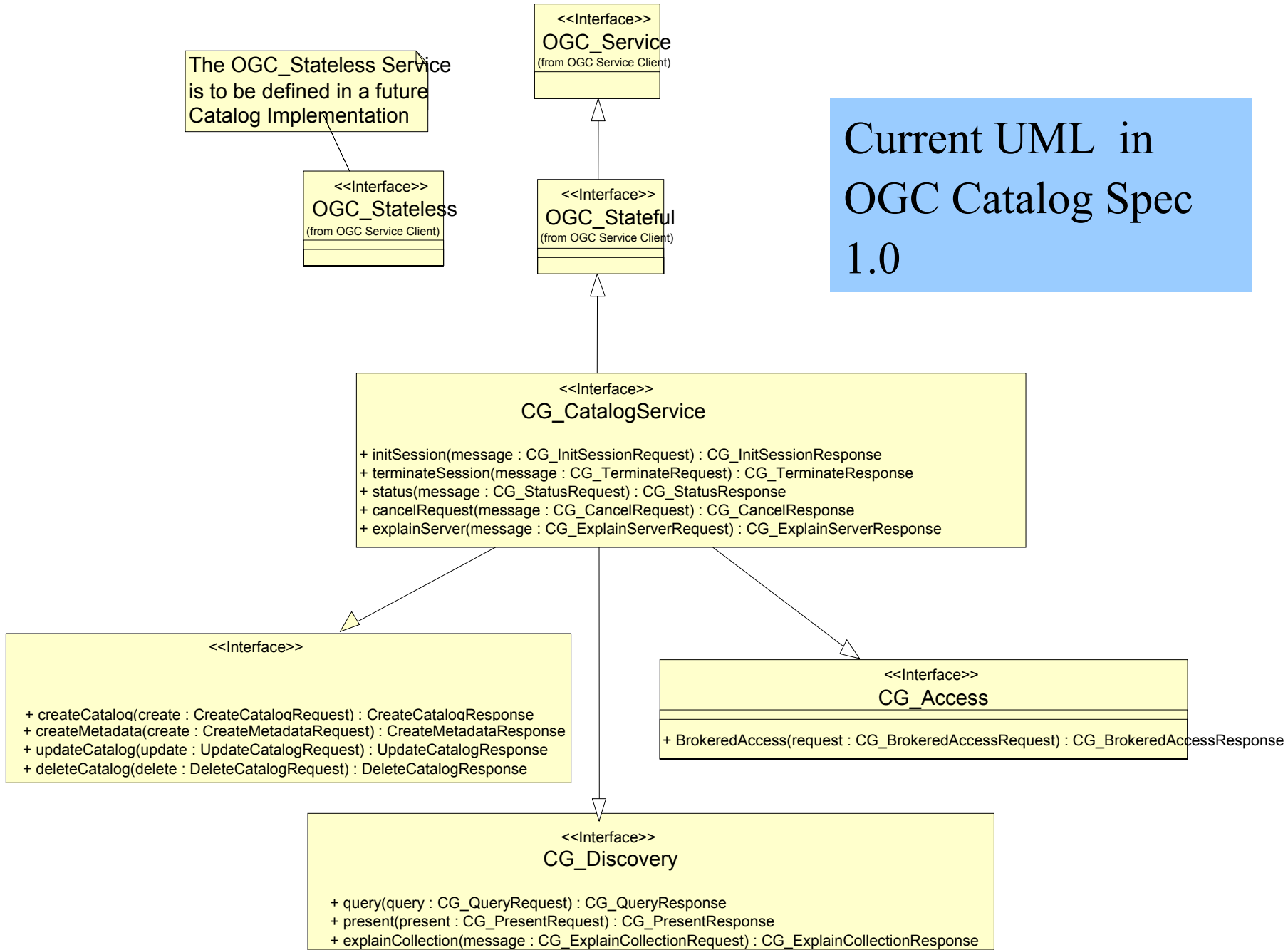
Lou Reich

6-03-2001

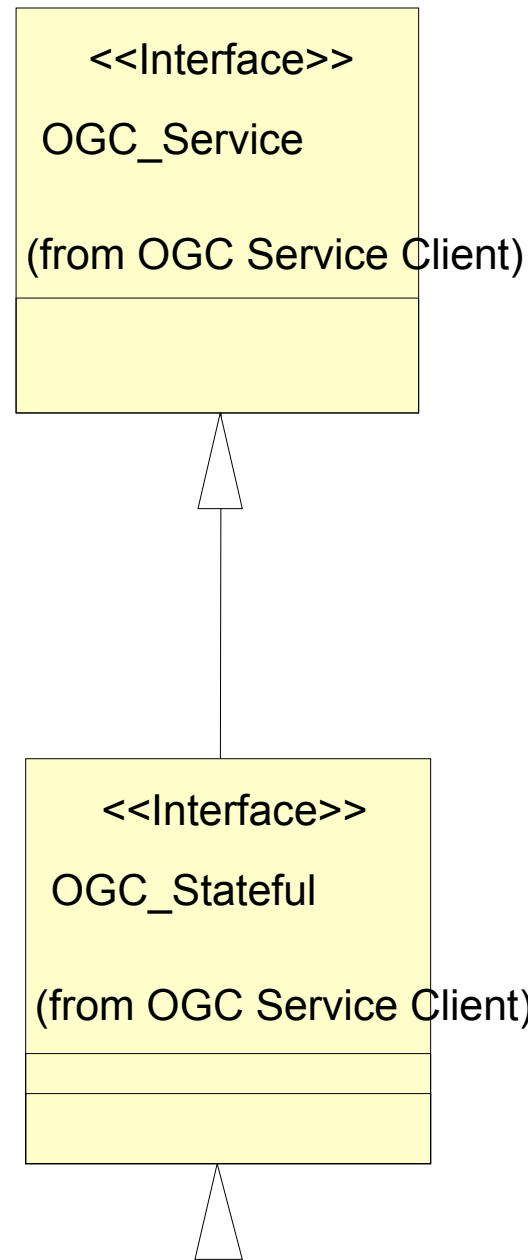
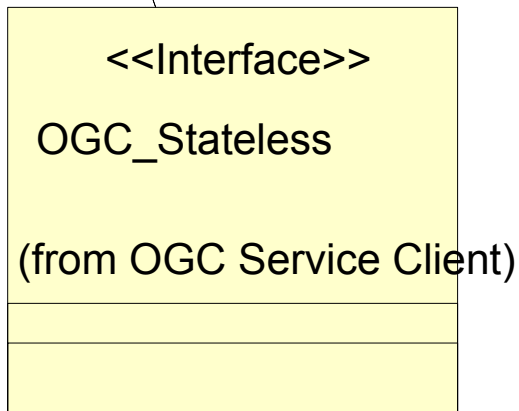
# Background

- OGC Catalog Specification
- OGC Stateless Catalog Prototype
- Web Feature Server
- Web Registry Service

# Current UML in OGC Catalog Spec 1.0



The OGC\_Stateless Service  
is to be defined in a future  
Catalog Implementation



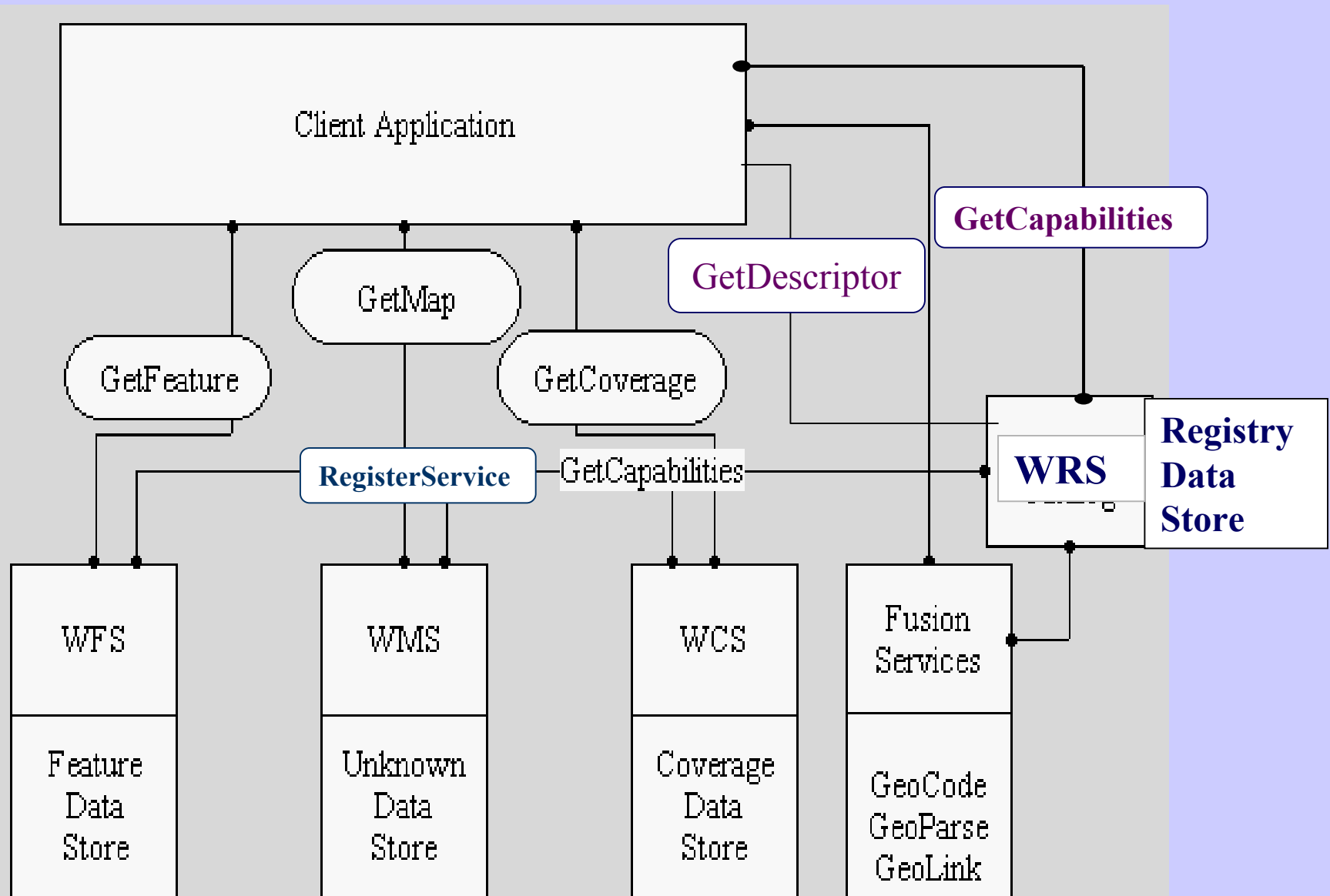
# Current Stateless Query Prototype

The URL (GET or POST) syntax supported is:

<http://host[:port]>http://host[:port]/path/script?[DOCID=docid]&Query=ogc\_common\_query&ESN=esn&RS=rs&MAXRECS=maxrecs

Name	Description	Allowed Values
----	-----	-----
ESN	element set name	'b', 's', 'f' (for brief, summary and full)
RS	results set format	' XML'
QUERY	ogc simple query	See Below
DOCID	id of service	String
SCHEMA	Schema used for result sets	'FGDC', 'ISO19119'
'MAXRECS'	max number of records to be returned.	integer

# Web Registry Service



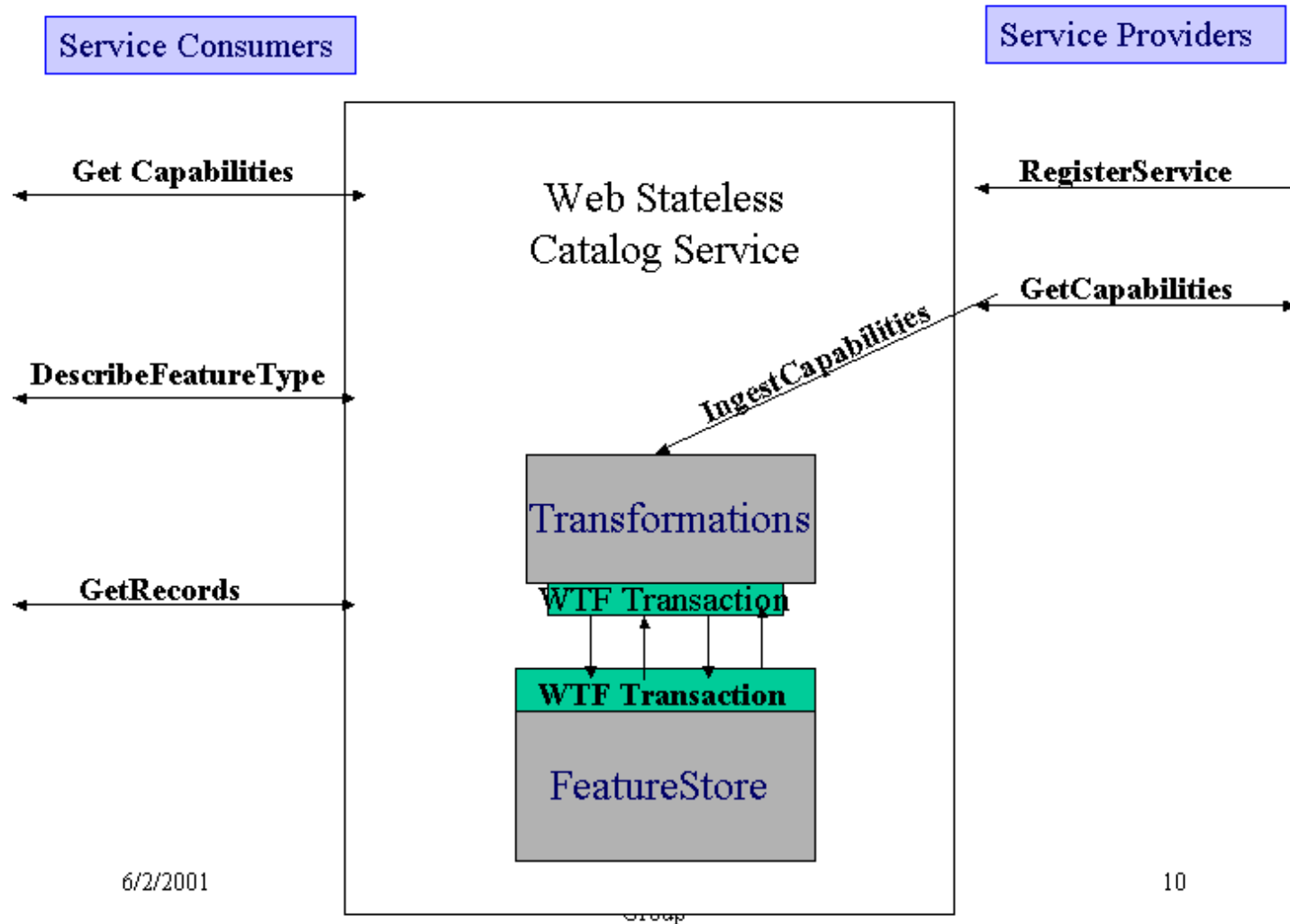
# Liege Proposals

- Catalog SIG expressed concern the WRS should be integrated into OGC Catalog Specification V1.1 (Cat RWG voted agreement)
- Proposals both in Catalog SIG and Architecture SIG to investigate use of WFS implementation of WRS interfaces
  - The WFS would need to add a RegisterService interface
  - The WFS DescribeFeatureType interface provided a working implementation of the Catalog Services ExplainCollection interface which was missing from the WRS draft and WFS Transaction interface to implement CG\_CatalogManager Interface in OGC Catalog Spec
  - There was concern about the use of the current WFS interface names due to the overloading of the term Feature
- Series of telecons set up to discuss service model and catalog issues

# MAPPING StSC/WFS Operations

<b>General Model Operation</b>	<b>StSC/WRS Spec</b>	<b>WFS Spec</b>
<b>CG_ExplainServerRequest</b>	<b>GetCapabilities</b>	<b>GetCapabilities</b>
<b>CG_ExplainServerResponse</b>	<b>GetCapabilities</b>	<b>GetCapabilities</b>
<b>CG_QueryRequest</b>	<b>GetDescriptors</b>	<b>GetFeature</b>
<b>CG_QueryResponse</b>	<b>GetDescriptors</b>	<b>GetFeature</b>
<b>CG_PresentRequest</b>	<b>GetDescriptors (N/A)</b>	<b>GetFeature (N/A)</b>
<b>CG_PresentResponse</b>	<b>GetDescriptors (N/A)</b>	<b>GetFeature (N/A)</b>
<b>CG_ExplainCollectionRequest</b>	<b>GetCapabilities</b>	<b>DescribeFeatureType</b>
<b>CG_ExplainCollectionResponse</b>	<b>GetCapabilities</b>	<b>DescribeFeatureType</b>
	<b>RegisterService</b>	
<b>CG_CatalogManager</b>		<b>Transaction</b>
<b>CG_CatalogManger</b>		<b>LockFeature</b>

# Stateless Catalog Conceptual Architecture



# Data Model Issues

- Catalog data model was fine tuned for Information Discovery where little was known about target servers
- WFS data model was based on classic database model
- Specific differences included
  - WFS Features have one schema where all properties are queryable
  - Catalogs have queryable attributes and presentable element set
    - *Add present as a feature operation, and allow the queried feature type and the present feature type to be different*
  - WFS allow the client to specify a specific list of properties or all properties of a feature to be returned
  - Catalogs allow predefined subsets of element sets (brief,summary and full) to be returned
    - *Allow either parameter lists or defined subsets to be returned*
  - WFS allows only “Filter” XML queries, Catalogs allow client to choose from several query language
    - *Allow choice of query languages*

# Design Approach

- Create an OGC Web Service Profile of the OGC Stateless Catalog based on minimal modification of WFS Operation Specification which supports both data and service instance discovery
- Based on lessons learned, develop a general model of Stateless Catalog Services
- Include both sections in the next version of OGC Catalog Services

# Definition of Stateless Service

- Services which are intended to support lightweight clients and servers that support HTTP version 1.0 type functionality and do not necessarily maintain information required to enable persistent sessions.
- Individual implementations may support state preservation mechanisms such as cookies to optimize performances. The uses of such services may be negotiated but not required.

# Stateless Catalog Operations

## **Discovery Operations**

**<GetCapabilities>**

**<GetRecord>**

**<DescribeRecordType>**

## **Catalog Management Operations**

**<Transaction>**

**<LockFeature>**

## **Service Registration Operations**

**<RegisterService>**

# Changes to WFS Capabilities DTD

```
<!ELEMENT Operations (Insert | Update | Delete  
                      | Present | Query | Lock)+>
```

```
<!ENTITY % SCHEMALANGUAGES "XMLSCHEMA|DTD" >
```

```
<!ENTITY % RESULTFORMATS "GML2|XML">
```

```
<!ELEMENT MetadataURL (#PCDATA) >
```

```
<!ATTLIST MetadataURL
```

```
    type ( TC211 | FGDC | DEDSL | ISO11179) #REQUIRED
```

```
    format (XML | SGML | TXT) #REQUIRED>
```

# Changes to WFS Capabilities DTD (con't)

element	number	comments
=====	=====	=====
FeatureType	1	this is the name of the feature type
Title	1	an optional Meaningful title for the feature type (e.g. "Ontario Roads" for ROADL_1M")
Abstract	0/1	optional; no Default
Keywords	0/1	optional; no Default
SRS	<b>0/1</b>	the SRS that should be used when specifying the statof the feature
LatLonBoundingBox	<b>0/1</b>	optional; default from parent
MetadataURL	0+	optional; no default
Operations	1	a <b>list</b> of available operations

# GetFeature DTD

```
<!ELEMENT GetFeature (Query+,Filter?)>
<!ATTLIST GetFeature outputFormat (GML2) #IMPLIED
                handle CDATA #IMPLIED
                maxFeatures CDATA #IMPLIED>

<!ELEMENT GetFeatureWithLock (Query+,Filter?)>
<!ATTLIST GetFeatureWithLock outputFormat (GML2) #IMPLIED
                handle CDATA #IMPLIED
                maxFeatures CDATA #IMPLIED>

<!ELEMENT Query (PropertyName*,Filter?)>
<!ATTLIST Query handle CDATA #IMPLIED
                typeName CDATA #REQUIRED
                version (ALL|CDATA) #IMPLIED>
```

# GetRecord DTD

```
<!ELEMENT GetRecord (Query+,QuerySpec?)>
<!ATTLIST GetRecord
    outputFormat (HTML|XML|GML2) #DEFAULT="XML"
    handle CDATA #IMPLIED
    maxRecords CDATA #IMPLIED
    outputRecType CDATA>
<!ELEMENT Query (PropertySpec,Queryspec?)>
<!ATTLIST Query handle CDATA #IMPLIED
    typeName CDATA #REQUIRED
    version (ALL|CDATA) #IMPLIED>
<!ELEMENT PropertySpec (PropertySet|PropertyName*)
<
<!ELEMENT PropertySet EMPTY>
<!ATTLIST PropertySpec
    setName CDATA ("Brief"|"Summary"|"Full"|"Hits")>
<!ELEMENT PropertyName (#PCDATA)>
<!ELEMENT QuerySpec ANY>
<!ATTLIST QuerySpec
    Querylang CDATA ("CQLXML"|"CQLSQL"|"SFSQL"|"SQLMM")
```

# Conclusion

- What I really defined is a GetObject which is the parent of GetFeature and GetDescription
- Inheritance should be at the Operation
- Classification Hierarchy
  - GetCapabilities
    - GetMap
    - RenderCoverage
    - GetView
  - GetStructuredObject
    - GetFeature
    - GetCoverage
    - GetDescription