Understanding ECFramework…

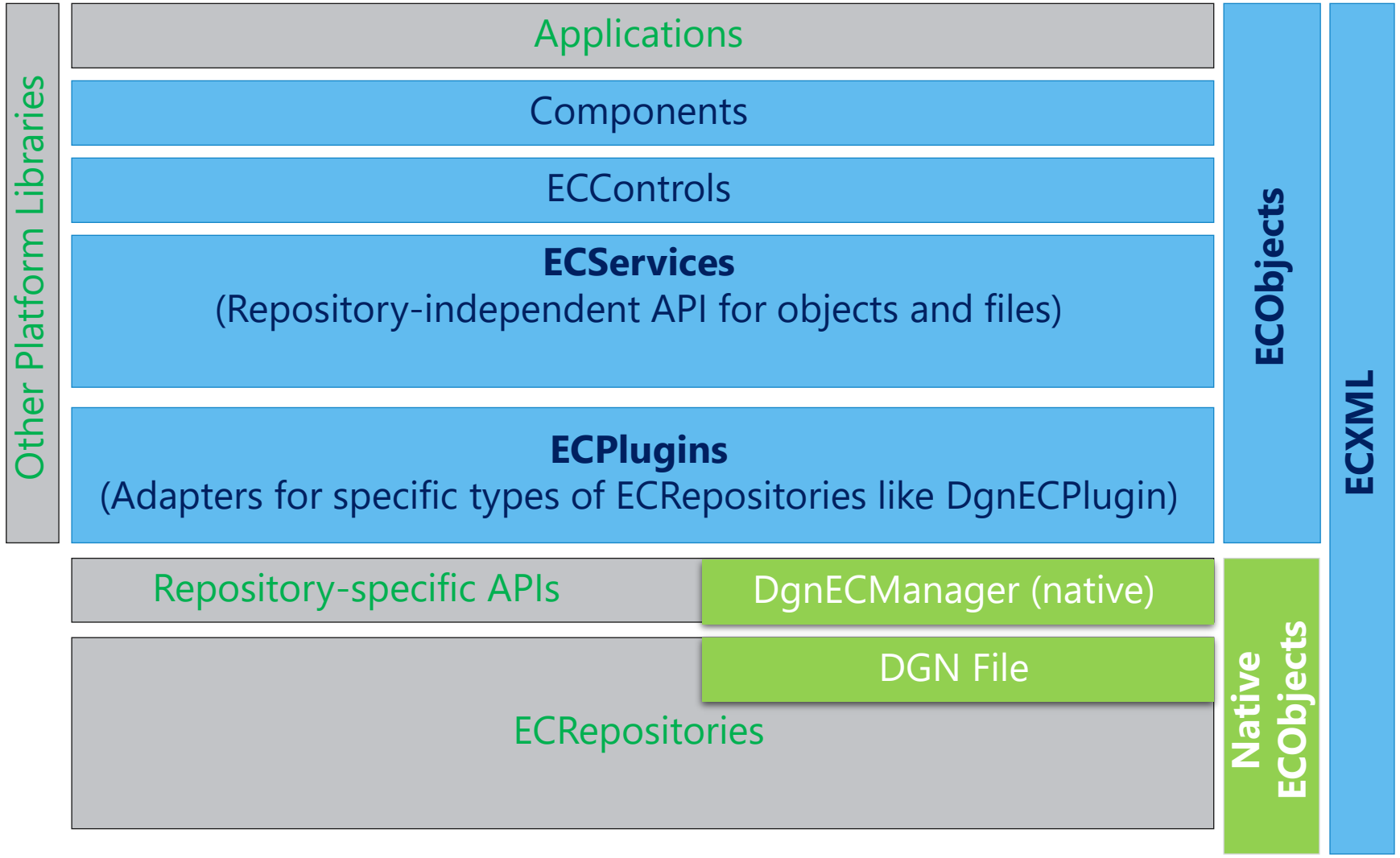
Bentley®

Why do we must know EC?

- EC is strategic throughout BSW.
  - Microstation CE property system
  - i-models, iModel
  - OpenPlant, Civil frameworks
  - And many more…

Bentley

ECFramework 2.0 Overview



Bentley

EC Core Concepts

Bentley

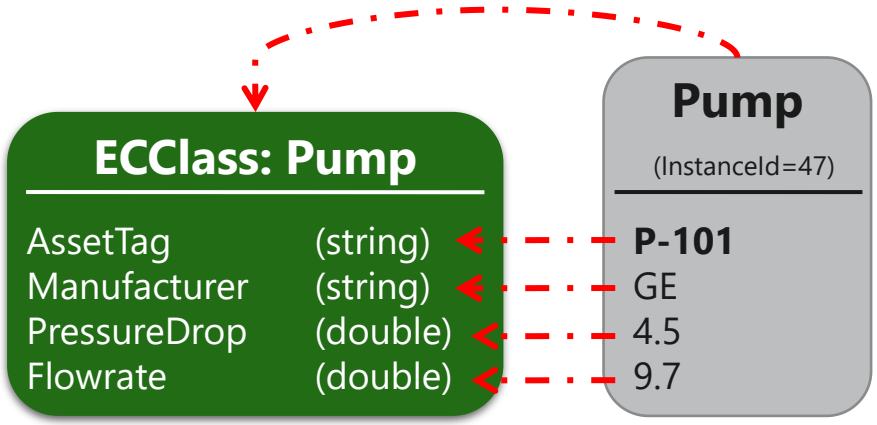
There is no such thing as an “ECObject”

- “Object” is ambiguous
  - ECClass defines an object
  - ECInstance is an instance of an object
- “ECObjects” refers to both concepts (and related)
- **ECInstance** ~ Entity
- **ECClass** ~ EntityType
- **ECProperty** ~ Attribute Definition
- **ECPropertyValue** ~ Attribute Value

Bentley

Data Abstractions

- **ECInstance**
  - References ECClass
  - Has InstanceId
  - Holds ECPropertyValues
- **ECPropertyValue**
  - References ECProperty
  - Holds a value
  - Can be NULL
- **ECRelationshipInstance**
  - Is an ECInstance (may have ECPropertyValues)
  - Has 1 “source” and 1 “target” ECInstance



Bentley

Metadata abstractions

- **ECSchema**—just a collection of ECClasses
- **ECSchemaReference**—reference other ECSchemas
- **ECClass**—holds ECProperties
- **ECProperty** (definition)—Name, datatype, etc.
  - **ECStructProperty**—embedded struct
  - **ECArrayProperty**—array of primitives or structs
- **ECRelationshipClass**—an ECClass that defines relationship
- **ECCustomAttribute**—custom metadata on ECSchema, ECClass, or ECProperty

ECXML is two different XML formats…

ECSchemaXML for the metadata.

```
<?xml version="1.0" encoding="utf-16" ?>
- <ECSchema schemaName="Demo" nameSpacePrefix="d" version="1.0" description="Demonstrate
  ECSchema Concepts" displayLabel="Demonstration"
  xmlns="http://www.bentley.com/schemas/Bentley.ECXML.2.0">
  - <ECClass typeName="PUMP" displayLabel="Pump" isDomainClass="True">
      <ECProperty propertyName="AssetTag" typeName="string" description="The "BusinessKey" of
        the PUMP ECClass" displayLabel="Tag" />
      <ECProperty propertyName="Manufacturer" typeName="string" />
      <ECProperty propertyName="PressureDrop" typeName="double" displayLabel="Pressure Drop" />
      <ECProperty propertyName="FlowRate" typeName="double" displayLabel="Flow rate" />
    </ECClass>
</ECSchema>
```

ECInstanceX

```
<PUMP instanceID="qj47" xmlns="Demo.01.00">
  <AssetTag>P-101</AssetTag>
  <Manufacturer>GE</Manufacturer>
  <PressureDrop>4.5</PressureDrop>
  <FlowRate>9.7</FlowRate>
</PUMP>
```

ECInstanceXml contains no metadata… just a name for looking the extensible metadaupta.

Bentley

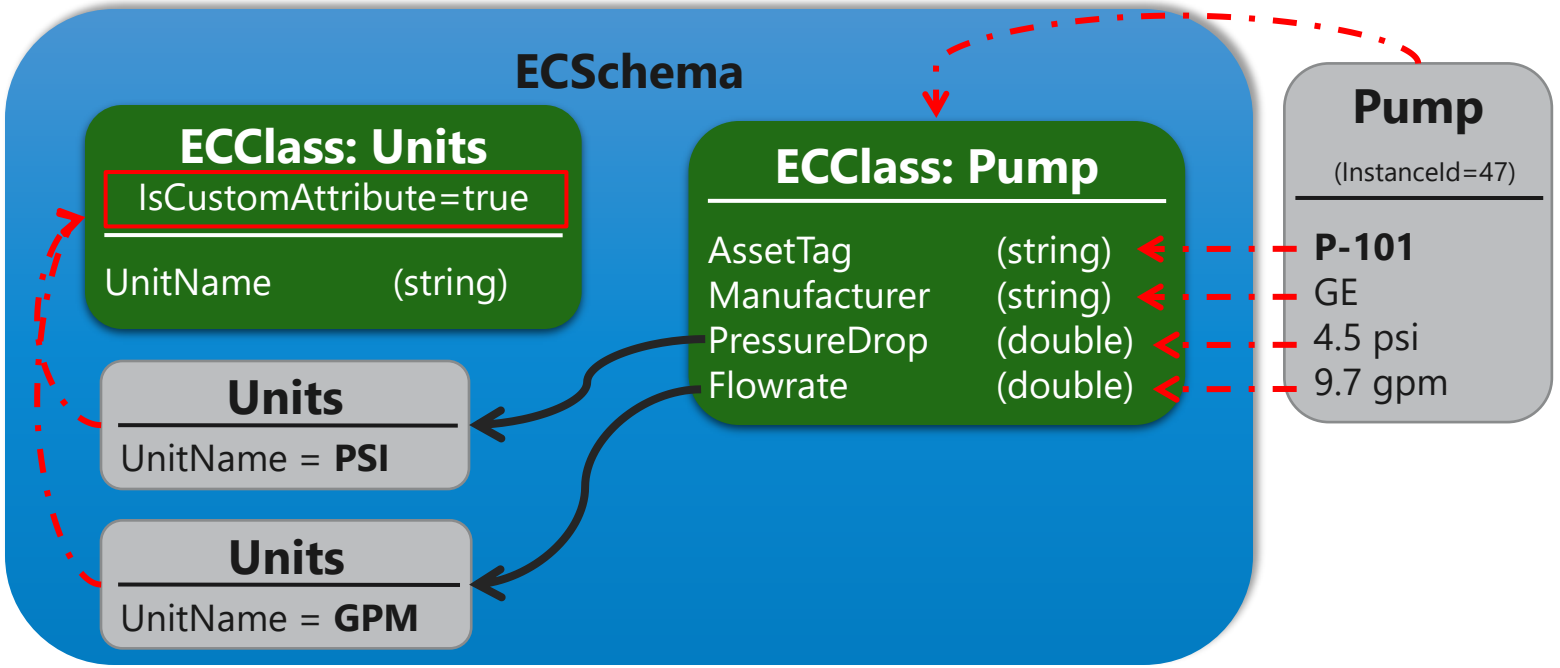
ECCustomAttributes

- Extensible metadata
- Not a “custom” or “user-defined” property
- Patterned after .NET “custom attributes”

ECCustomAttributes extend Metadata in ECObjects

An “ECCustomAttribute ECClass” is an ECClass with IsCustomAttribute = true.

Attach ECInstances of an ECCustomAttribute to ECClasses, ECProperties, and ECSchemas to extend them.



Bentley

ECCustomAttributes in ECSchemaXML

```
<ECClass typeName="PUMP" displayLabel="Pump" isDomainClass="True">
  <ECProperty propertyName="AssetTag" typeName="string" description="The "BusinessKey" of t
  <ECProperty propertyName="Manufacturer" typeName="string" />
- <ECProperty propertyName="PressureDrop" typeName="double" displayLabel="Pressure Drop">
  - <ECCustomAttributes>
    - <UnitSpecification xmlns="Unit_Attributes.01.00">
        <KindOfQuantityName>PRESSURE</KindOfQuantityName>
        <DimensionName>M_PER_L_T2</DimensionName>
        <UnitName>POUND_FORCE_PER_INCH_SQUARED</UnitName>
        <AllowableUnits />
      </UnitSpecification>
    </ECCustomAttributes>
  </ECProperty>
  <ECProperty propertyName="FlowRate" typeName="double" displayLabel="Flow rate" />
</ECClass>
```


Bentley

Examples of ECCustomAttributes

UnitSpecification
CalculatedECPropertySpecification
BusinessKeySpecification
InstanceLabelSpecification
Display metadata (e.g. “Category” for an ECProperty)
Mapping to other systems (e.g. a database table)
Name of a class to supply “behaviors” for ECInstances.

ECRelationshipClass

- Like DB “link table”
- Is an ECClass
- Strength (Referencing, Holding, Embedding)
- Source and Target
  - Constraints (Which class? Polymorphic?)
  - Cardinality on this “end”
  - RoleLabel (when “reading” starting from this end)
    - “holds” from source end
    - “is held by” from target end

```
<ECRelationshipClass typeName="WidgetHasGadgets"
description="WidgetHasGadgets" strength="referencing">
    <Source cardinality="(1,1)" roleLabel="has Gadgets"
polymorphic="False">
        <Class class="Widget" />
    </Source>
    <Target cardinality="(1,N)" roleLabel="are held by
Widget" polymorphic="False">
        <Class class="Gadget" />
    </Target>
</ECRelationshipClass>
```

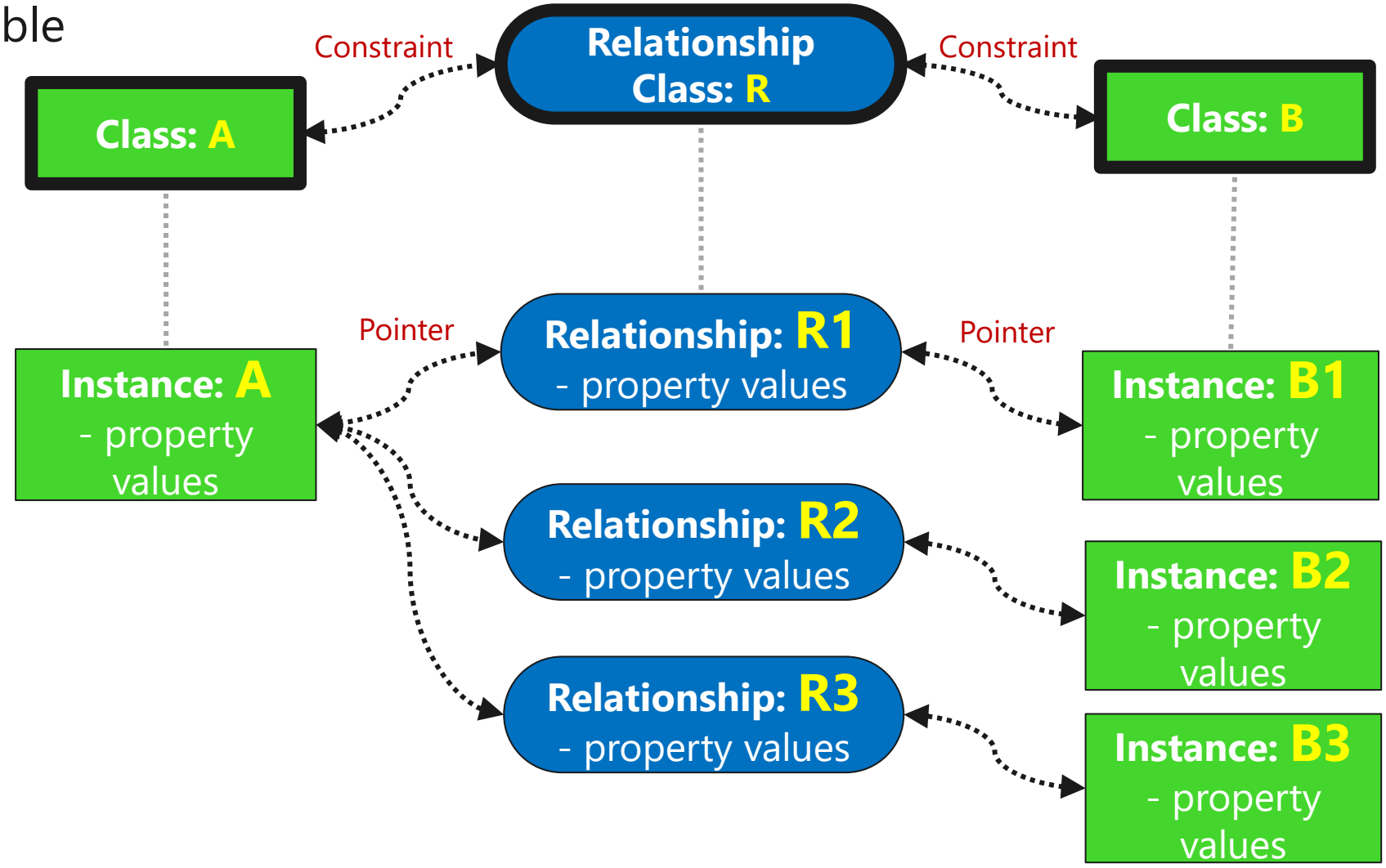

Bentley

Cardinality

- Bentley.ECObjects.Schema.StandardCardinality
  - OneOne (1,1)
  - ZeroOne (0,1)
  - ZeroMany (0,n)
  - OneMany (1,n)
- Only referring to **one** “end” of the relationship!
  - A one-to-one ECRelationship will have
    - source cardinality of (1,1)
    - Target cardinality of (1,1)
  - A one-to-many ECRelationship will have
    - source cardinality of (1,1)
    - Target cardinality of (1,n) or (0, n)

ECRelationships

- Like a “row” in a link table
- Allows property values



Bentley

ECRelationshipInstances

- Hold a reference to the source ECInstance and target ECInstance
- Like a “row” in a link table
- In a one-to-many relationship where “many”=3, there will be 3 ECRelationshipInstances

Bentley

ECObjects Implementations

- One concrete implementation of ECSchema/ECClass
- Abstract base class IECInstance with one default implementation and other implementations in DgnEC
- XML serialization/deserialization
- Binary serialization/deserialization

Bentley

ECObjects in DGN

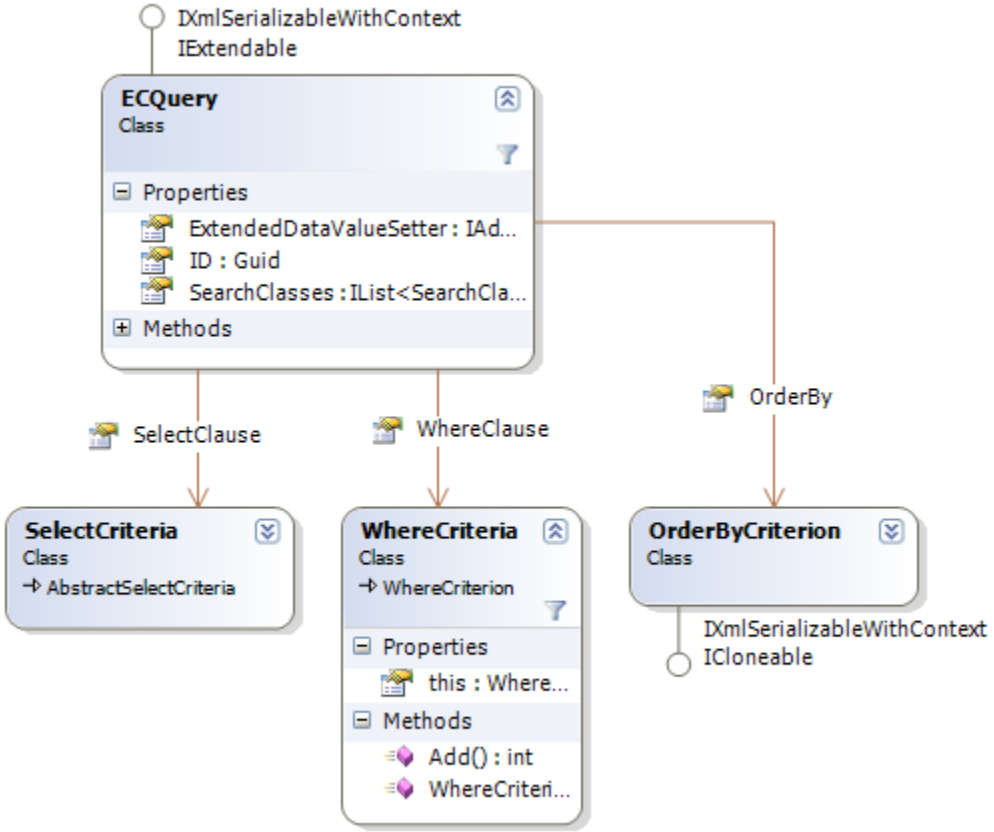
- Microstation
  - DgnECManager
  - DgnECPlugin3

Bentley

ECQuery

Conceptually similar to SQL Queries

Not a text format, but an object graph



Bentley

EC Expressions

- An ECExpression is an interpreted expression language with first class access to the values of ECInstances
- ECExpressions are designed to be easy to write and guaranteed to not crash the application if they fail
- One of its many uses is as the language for calculated properties.

Employee ECClass

```
<ECClass typeName="Employee" description="Employee">
  <ECCustomAttributes>
    <InstanceLabelSpecification xmlns="Bentley_Standard_CustomAttributes.01.00">
      <PropertyName>FullName</PropertyName>
    </InstanceLabelSpecification>
  </ECCustomAttributes>
  <ECProperty propertyName="FullName" typeName="string" displayLabel="Full name">
    <ECCustomAttributes>
      <CalculatedECPropertySpecification xmlns="Bentley_Standard_CustomAttributes.01.00">
        <ECExpression>this.Firstname &amp; " " &amp; this.Lastname</ECExpression>
        <FailureValue>Anonymous</FailureValue>
        <ParserRegularExpression>^(?<Firstname> [\w]+)(?<Lastname>[\w]+)</ParserRegularExpression>
      </CalculatedECPropertySpecification>
    </ECCustomAttributes>
  </ECProperty>
  <ECProperty propertyName="Lastname" typeName="string" description="Lastname" />
  <ECProperty propertyName="Firstname" typeName="string" description="Firstname" />
</ECClass>
```


Bentley

Further readings

- Please refer to DgnEC training session

Bentley

For more information

- Developer Portal
  - Developer.Bentley.com
- Programming Community
  - Communities.Bentley.com/products/programming
- MicroStation Programming Community
  - https://communities.bentley.com/products/programming/microstation_programming/
- MicroStation Programming Blog
  - https://communities.bentley.com/products/programming/microstation_programming/b/weblog

Bentley

Connect with Bentley

Connect with us

www.bentley.com

developer.bentley.com

communities.bentley.com/products/programming

Social Media

Medium.com

MicroStation

