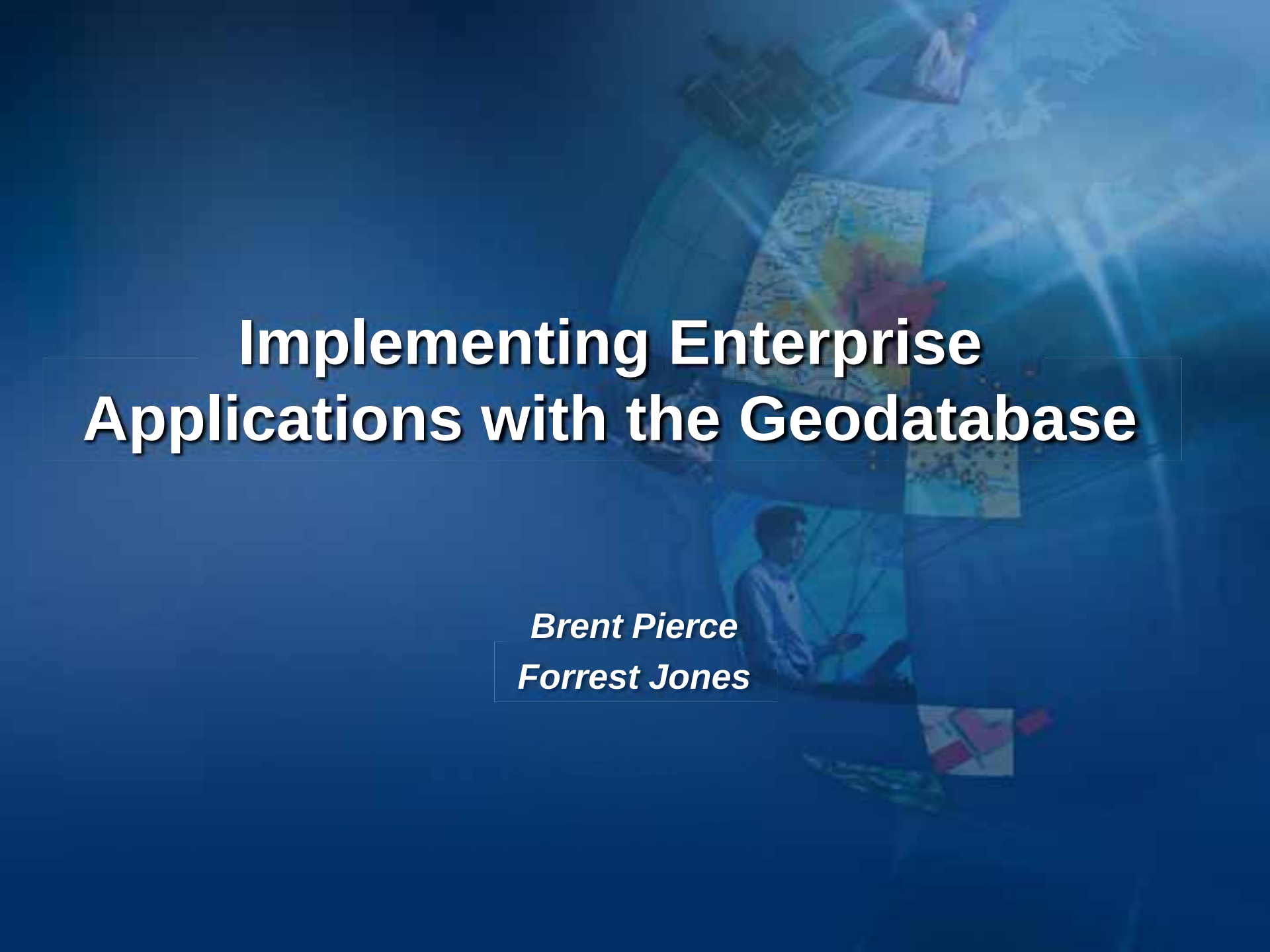
Implementing Enterprise Applications with the Geodatabase



Brent Pierce

Forrest Jones

Assumptions

- Good working knowledge of the Geodatabase
- Experience in programming against the GDB API
- Code examples will use C#.NET
  - Readily translatable to VB, VB.NET, VC++, etc.
  - Code samples available with the slides on EDN after the Summit
  - Code samples in the presentation might not compile
    - (shortened to improve readability)
- Lots of content, little time
  - Please hold all questions to the end
    - We'll be at the Tech Talk and Server Island Area for questions

What is covered in this session

- Connecting to the Geodatabase
- Working with Schemas
- Geodatabase cursor model
- Querying Data
  - Spatial / Non-Spatial
  - Views and Joins
- Transactions and Editing
  - Transactions
  - Versioned / Non-Versioned
  - Reconcile and Post
- Difference Cursors
- Archiving
- Version Events
- Code Instrumentation and Tracing

Workspaces

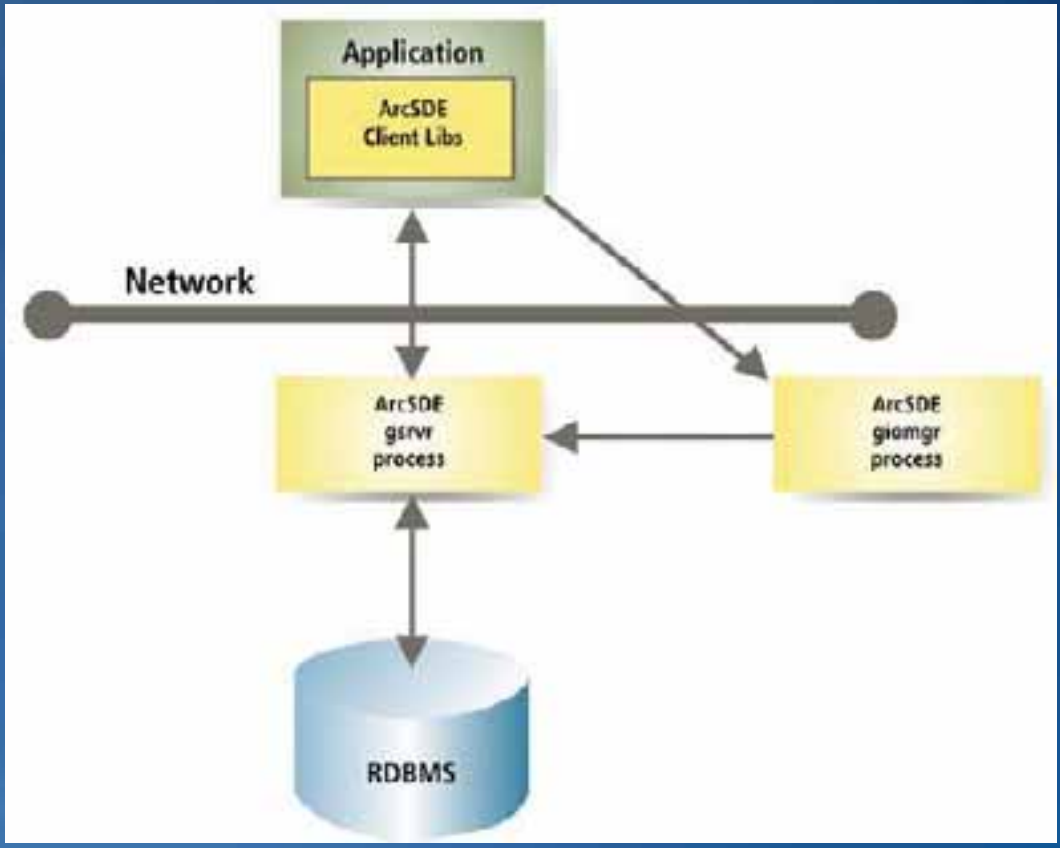
Connections and workspaces (Enterprise)

- The workspace is the DBMS connection and gives access to the data
- Each connection/workspace creates an ArcSDE and DBMS process.
  - Type of connection controls server/client load
- ArcGIS maintains a workspace/connection pool to save resources
  - Workspaces are uniquely instanced
- Opening too many connections consumes server and DBMS resources

Database Connection

Application Server

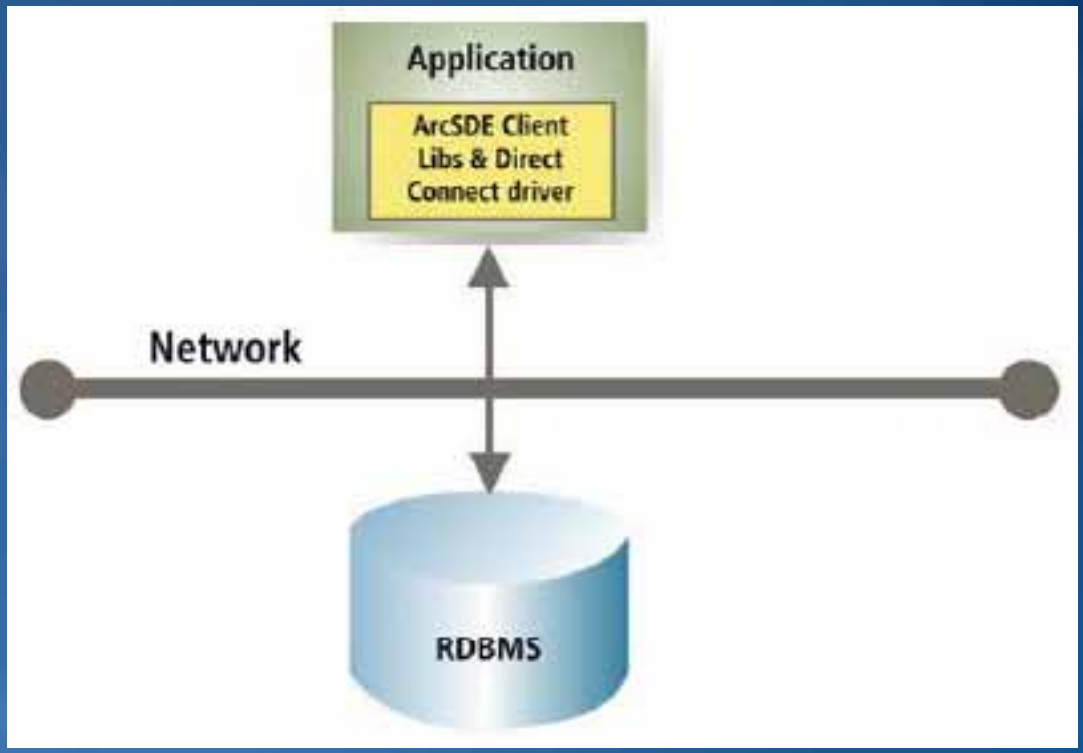
- **Server machine must run ArcSDE service**
  - **giomgr/gsrvr processes**
  - **Can decrease client load by increasing server load**



Database Connection

Direct Connect

- Connect directly to database
- No ArcSDE service required
  - No giomgr/gsrvr processes
  - Can decrease server load by increasing client load



Making a connection

- Connecting using a property set

```
IWorkspaceFactory sdeWkspFact = new SdeWorkspaceFactoryClass();
IPropertySet propset = new PropertySetClass();

propset.SetProperty("SERVER",“crimsontide");
propset.SetProperty("INSTANCE","5151");
propset.SetProperty("USER","brent");
propset.SetProperty("PASSWORD","brent");
propset.SetProperty("DATABASE","null");
propset.SetProperty("VERSION","SDE.DEFAULT");
propset.SetProperty("AUTHENTICATION_MODE","DBMS");

IWorkspace workspace = sdeWkspFact.Open(propset, 0);
```

- Connecting using a connection file

```
string nameOfFile = "D:\\data\\redarrow.sde”;
IWorkspace workspace = workspaceFactory.OpenFromFile(nameOfFile, 0);
```

ArcSDE for SQL Server Express Geodatabases

- **Detailed API**
  - Analyze statistics
  - Create a geodatabase

```
IDataServerManager DSManager = new DataServerManagerClass();
DSManager.ServerName = “crimsontide\\sqlexpress”;
DSManager.Connect();
IDataServerManagerAdmin DSMAdmin = (IDataServerManagerAdmin)DSManager;

DSMAdmin.CreateGeodatabase(“MyGDB",“C:\\MyGDB.mdf",100,“C:\\MyGDB.ldf“,100);
```

  - Detatch/Attach geodatabases

```
DSMAdmin.DetatchGeodatabase("MyGDB");

DSMAdmin.AttachGeodatabase(“MyGDB“,“C:\\MyGDB.mdf",“C:\\MyGDB.ldf“);
```

Once Connected…

Information about the connected workspace

- IDatabaseConnectionInfo2
  - GeodatabaseServerClass
    - esriServerClassEnterprise
    - esriServerClassWorkgroup
    - esriServerClassPersonal
  - ConnectionDBMS
    - esriDBMS_DB2
    - esriDBMS_Informix
    - esriDBMS_Oracle
    - esriDBMS_SQLServer
    - esriDBMS_PostgreSQL
  - ConnectionCurrentDateTime
  - ConnectedUser
  - ConnectedDatabase
- IFeatureClassStorage

Working With Schemas

Schema Cache

What is the schema cache?

- A cached snapshot of the GDB schema
  - Used by ArcGIS (opening Map Documents, Reconcile)
  - Requires a static data model
    - GDB schema changes will not be reflected in the cache
- Can improve performance when opening datasets
  - Reduces database round trips by using the cached information
- APIs to access the schema cache
  - `IWorkspaceFactorySchemaCache`
    - `EnableSchemaCache`
    - `EnableSchemaCaching`
    - `RefreshSchemaCache`
    - `DisableSchemaCache`

Schema Cache

When to use a schema cache?

- **Beneficial when working with large static data models**
  - **Tables, fields, domains, sub-types and relationships are well defined and will not change**
- **If the application opens and uses many classes**
  - **These should be opened at the start of the application and references maintained throughout the lifetime of the application**

Schema Cache

How to leverage in enterprise applications

- Enable schema cache before tables are opened
  - Calls to `OpenFeatureClass, OpenTable, IName.Open` will be optimized
  - Can enable schema caching at the factory level
- Cache needs to be “fresh”
  - If in dynamic environments schema changes will not be visible until cache is refreshed.

```
//Spatial Cache Refresh if stale
if(sCache.IsSchemaCacheStale(workspace))
{
   sCache.RefreshSchemaCache(workspace);
}
```

- your responsibility to disable the cache
  - Must be disabled before releasing the workspace object that is cached

Using Schema Caching

1. Cast to `IWorkspaceFactorySchemaCache` from the workspace factory and open the workspace

```
IWorkspaceFactorySchemaCache sCache =
(IWorkspaceFactorySchemaCache)workspaceFactory;

IWorkspace workspace = workspaceFactory.Open(propset, 0);

sCache.EnableSchemaCache(workspace);

IFeatureWorkspace featureWorkspace = (IFeatureWorkspace)workspace;
IFeatureClass parcel = featureWorkspace.OpenFeatureClass("ParcelClass");
//...open all feature classes used in application

sCache.DisableSchemaCache(workspace);
```

Using Schema Caching

❷ **Enable the schema cache on the workspace**

Alternatively `EnableSchemaCaching` will enable caching on all workspace passed out by the factory

```
IWorkspaceFactorySchemaCache sCache =
(IWorkspaceFactorySchemaCache)workspaceFactory;

IWorkspace workspace = workspaceFactory.Open(propset, 0);

sCache.EnableSchemaCache(workspace);

IFeatureWorkspace featureWorkspace = (IFeatureWorkspace)workspace;
IFeatureClass parcel = featureWorkspace.OpenFeatureClass("ParcelClass");
//...open all feature classes used in application

sCache.DisableSchemaCache(workspace);
```

Using Schema Caching

❸ **Open all feature classes used in application**

Largest benefit for stand alone feature classes and tables

```
IWorkspaceFactorySchemaCache sCache =
(IWorkspaceFactorySchemaCache)workspaceFactory;

IWorkspace workspace = workspaceFactory.Open(propset, 0);

sCache.EnableSchemaCache(workspace);

IFeatureWorkspace featureWorkspace = (IFeatureWorkspace)workspace;
IFeatureClass parcel = featureWorkspace.OpenFeatureClass("ParcelClass");
//...open all feature classes used in application

sCache.DisableSchemaCache(workspace);
```

Using Schema Caching

4. Disable the schema cache after the classes have been opened

```
IWorkspaceFactorySchemaCache sCache =
(IWorkspaceFactorySchemaCache)workspaceFactory;

IWorkspace workspace = workspaceFactory.Open(propset, 0);

sCache.EnableSchemaCache(workspace);

IFeatureWorkspace featureWorkspace = (IFeatureWorkspace)workspace;
IFeatureClass parcel = featureWorkspace.OpenFeatureClass("ParcelClass");
//...open the all feature classes used in application

sCache.DisableSchemaCache(workspace);
```

Using Schema Caching

- Gotchas
  - If the GDB schema changes cache will have to be refreshed
    - Geodatabase schema changes will not be visible until cache is refreshed

```
IWorkspaceFactorySchemaCache sCache =
(IWorkspaceFactorySchemaCache)workspaceFactory;

IWorkspace workspace = workspaceFactory.Open(propset, 0);

sCache.EnableSchemaCache(workspace);

IFeatureWorkspace featureWorkspace = (IFeatureWorkspace)workspace;
IFeatureClass parcel = featureWorkspace.OpenFeatureClass("ParcelClass");
//...open all feature classes used in application

sCache.DisableSchemaCache(workspace);
```

Cursors

Cursors

What are they?

- A geodatabase object used for the iteration of records returned from a query

- 3 Class Cursors
  - Search (general query cursor)
  - Update (positioned update cursor)
  - Insert (bulk inserts)

- 1 QueryDef Cursor
  - Defined query (e.g. IQueryDef.Evaluate)

- What's the difference?
  - Rows created by Class cursors are bound to the class which created the cursor, rows created by a QueryDef cursor are not bound to a class

Class Cursors

Types

- **Search cursors**
  - Returns rows specified by a Query or Spatial Filter
- **Update cursors**
  - Update and delete rows specified by the filter
  - Specify the ObjectID field
- **Insert cursors**
  - Used for inserting rows into a table
- **Accessed by**
  - Corresponding methods on table or feature class

Class Cursors: Recycling Property

What is Recycling?

- A recycling cursor is a cursor that hands out the same row object on each call to **`NextRow`**
  - Internal data structures and objects will be reused

```
//example of a recycling cursor
IFeatureCursor searchCursor = featureClass.Search(queryFilter, true)
```

- A non-recycling cursor is a cursor that hands out a new row object on each call to **`NextRow`**
  - New internal data structures and objects will be created for each row

```
//example of a non-recycling cursor
IFeatureCursor searchCursor = featureClass.Search(queryFilter, false)
```

Class Cursors: Recycling

When to use a recycling cursor

- Use recycling cursors when references to the current row and its values **do not need** to be kept beyond the next call to `NextRow/NextFeature`
- do not pass the references around as some other method may decide to hold it

Class Cursors: Non-Recycling

When to use a non-recycling cursor

- Use non-recycling cursors when references to the current row and its values are needed beyond the next call to `NextRow/NextFeature`
- Commonly used to cache sets of rows (long lived references)
- Some Geodatabase APIs require sets of rows – should be retrieved as non-recycled rows

QueryDef Cursors

- Query based on one or more tables
  - Analogous to an SQL Query
  - Get a cursor back
  - Not bound to a class
- Tables must be in database
  - Do not have to be registered with the geodatabase
- Can result in a feature layer
  - Need to use TableQueryName::IQueryName2 if no ObjectIDs in input tables
- Are also used to establish joins

Example: QueryDef Cursors

- Simple Query (One Table)

```
//Create the query definition
IQueryDef queryDef = featureWorkspace.CreateQueryDef();

//Provide a list of tables to join
queryDef.Tables = "PoleFeature";

//Retrieve the fields from all tables
queryDef.SubFields = "bob.PoleFeature.TAG, bob.PoleFeature.SHAPE";

//Set up the join based on the owner_name attribute
queryDef.WhereClause = "PoleFeature.TAG > 1000";

ICursor cursor = queryDef.Evaluate();

IRow row = cursor.NextRow();
```

Example: QueryDef Joins

- Simple 1:M table join

```
//Create the query definition
IQueryDef queryDef = featureWorkspace.CreateQueryDef();

//Provide a list of tables to join
queryDef.Tables = "PoleFeature, TransformerFeature";

//Retrieve the fields from all tables
queryDef.SubFields = "bob.PoleFeature.TAG, bob.PoleFeature.SHAPE,
bob.TransformerFeature.Tag_Val";

//Set up the join based on the owner_name attribute
queryDef.WhereClause = "PoleFeature.TAG = TransformerFeature.Tag_Val";

ICursor cursor = queryDef.Evaluate();
IRow row = cursor.NextRow();
```

Example: QueryDef Joins

- Complex 1:M table Join

```
IQueryDef queryDef = featureWorkspace.CreateQueryDef();
queryDef.Tables = "ElectricStation, CircuitSource, CableJunction";

queryDef.SubFields = "ElectricStation.StationName,
  ElectricStation.StationAbbreviation,
  CableJunction.ObjectID";
queryDef.WhereClause = "((ElectricStation.CircuitID LIKE 'A-235%' " +
    "AND SUBSTR(ElectricStation.CircuitID,5+1,1) not between '0' and '9')" +
    "OR (ElectricStation.CircuitID = 'A-235'))" +
    "AND ElectricStation.StationTypeCode = 'GEN'" +
    "AND CircuitSource.ObjectID = ElectricStation.CircuitSourceObjectID" +
    "AND ElectricStation.ObjectID = CircuitSource.ElectricStationObjectID" +
    "AND CableJunction.DeviceObjectID = ElectricStation.ObjectID" +
    "AND CableJunction.DeviceType = 'ENG'" +
    "OR  INSTR(UPPER(CircuitSource.PhaseDesignation),'123') > 0)";

ICursor cursor = queryDef.Evaluate();
IRow row = cursor.NextRow();
```

Querying and Joining Data

Querying and Joining Data

- Defining the query
  - Query Filters
  - Spatial Filters
- Optimizing queries
  - Spatial Caching
- Persisted Views
  - ArcSDE Views (Multi Versioned Views)
- On-The-Fly Joins
  - QueryTables
  - RelQueryTables

Defining the query

IQueryFilter

- Used when querying a single class
- `IQueryFilterDefinition::PostFixClause`
  - Supports order by

```
IQueryFilter queryFilter = new QueryFilterClass();
queryFilter.SubFields = "OBJECTID,FULLNAME,ParcelID";
queryFilter.WhereClause = “FULLNAME like ‘D%’";

IQueryFilterDefinition queryFilterDef =
(IQueryFilterDefinition)queryFilter;
queryFilterDef.PostfixClause = "ORDER BY FULLNAME";

IFeatureCursor featureCursor = featureClass.Search(queryFilter, true);
```

Defining the query

ISpatialFilter

- Used to query spatial aspects of a feature class
  - Inherits from `IQueryFilter`
- Will be satisfied by spatial cache if within cache extent

```
ISpatialFilter spatialFilter = new SpatialFilterClass();

spatialFilter.SubFields = "OBJECTID,FULLNAME,ParcelID,SHAPE";
spatialFilter.Geometry = searchShape;
spatialFilter.SpatialRel = within;
spatialFilter.WhereClause = "FULLNAME like 'D%'";

IFeatureCursor featureCursor = featureClass.Search(spatialFilter, true);
```

Defining the query

ISpatialFilter

- Used to query spatial aspects of a feature class
  - Inherits from `IQueryFilter`
- Will be satisfied by spatial cache if within cache extent

```
ISpatialFilter spatialFilter = new SpatialFilterClass();

spatialFilter.SubFields = "OBJECTID,FULLNAME,ParcelID,SHAPE";
spatialFilter.Geometry = searchShape;
spatialFilter.SpatialRel = within;
spatialFilter.WhereClause = "FULLNAME like 'D%'";

IFeatureCursor featureCursor = featureClass.Search(spatialFilter, true);
```

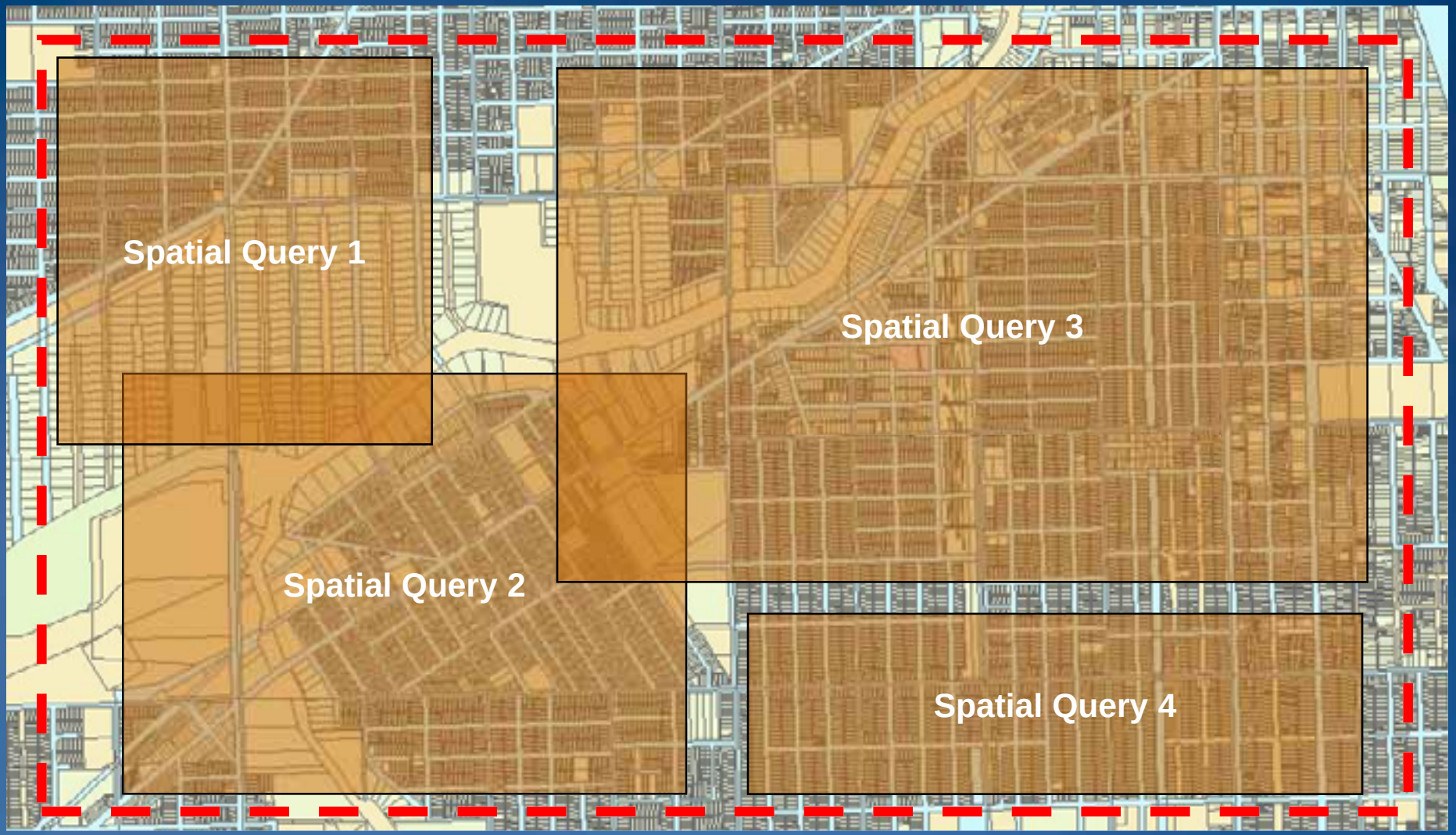
Spatial Caching

What is it?

- Client side caching of feature values over a given spatial extent (MapCache in ArcMap)
- Can speed up queries
  - Reduces roundtrips to the database
- `ISpatialCacheManager2`
  - `FillCache \ EmptyCache`
  - `CacheExtent`
- When to use?
  - If making many spatial queries within a common extent

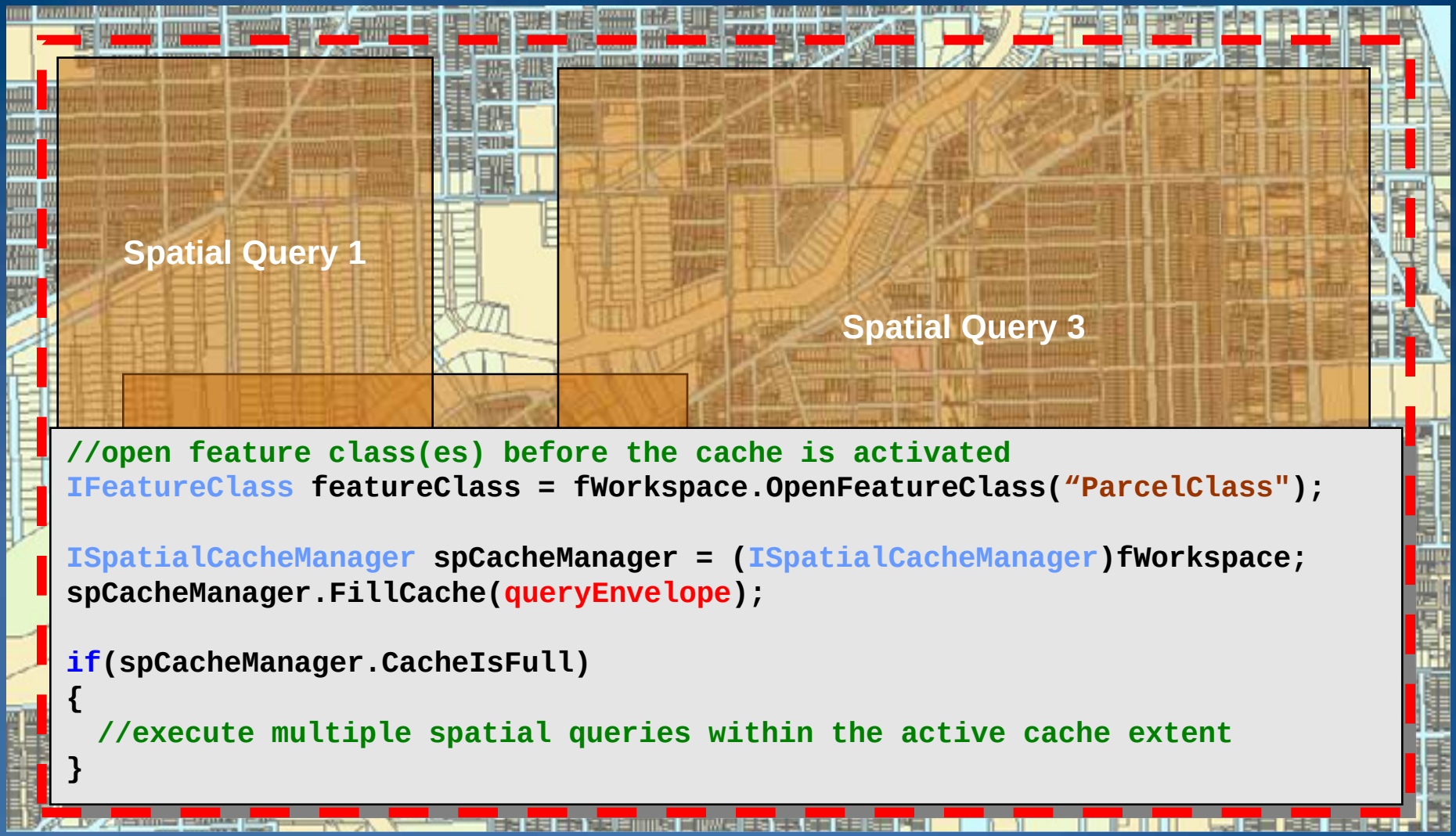
Spatial Caching

How to leverage spatial caching in enterprise applications



Spatial Caching

How to leverage spatial caching in enterprise applications



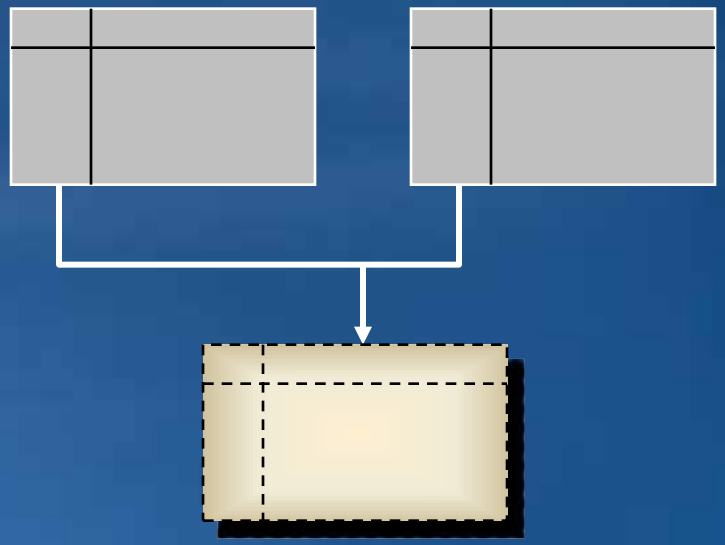

```
//open feature class(es) before the cache is activated
IFeatureClass featureClass = fWorkspace.OpenFeatureClass("ParcelClass");

ISpatialCacheManager spCacheManager = (ISpatialCacheManager)fWorkspace;
spCacheManager.FillCache(queryEnvelope);

if(spCacheManager.CacheIsFull)
{
  //execute multiple spatial queries within the active cache extent
}
```

Views, Table Joins and Complex Queries

- Persisted
  - ArcSDE Views (MultiVersion Views)
- On-The-Fly
  - QueryTables
    - Same Database
      - Spatial and Non-Spatial
  - RelQueryTables
    - Cross Database
      - Spatial and Non-Spatial



Persisted Views

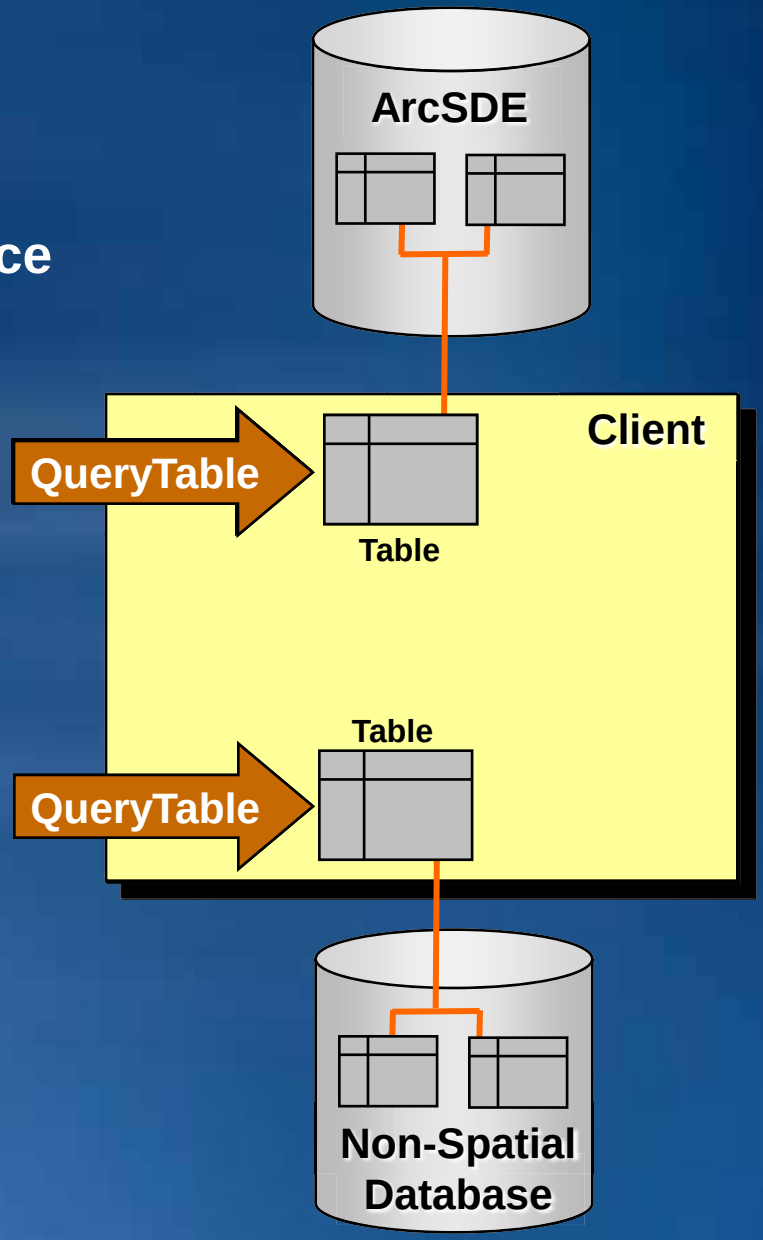
- Open views as tables (read only)

```
//open view through API as (read only) table
ITable myJoinedTable = fWorkspace.OpenTable(“databaseView");
```

- Must satisfy geodatabase rules for a valid table or feature class
  - Only one spatial column
  - Supported data types in returned fields

On-The-Fly Joins

- QueryTables (`TableQueryName`)
  - Tables must be within same datasource
  - Matches all candidates
  - Uses QueryDef object
  - Can be used with non-spatial tables



Example: QueryTables (TableQueryName)

- Steps to create a join via IQueryName2
  1. Create a new TableQueryName object
  2. Set the QueryDef and PrimaryKey property
  3. Cast to IDatasetName setting WorkspaceName and Name
  4. Open the name object as a table

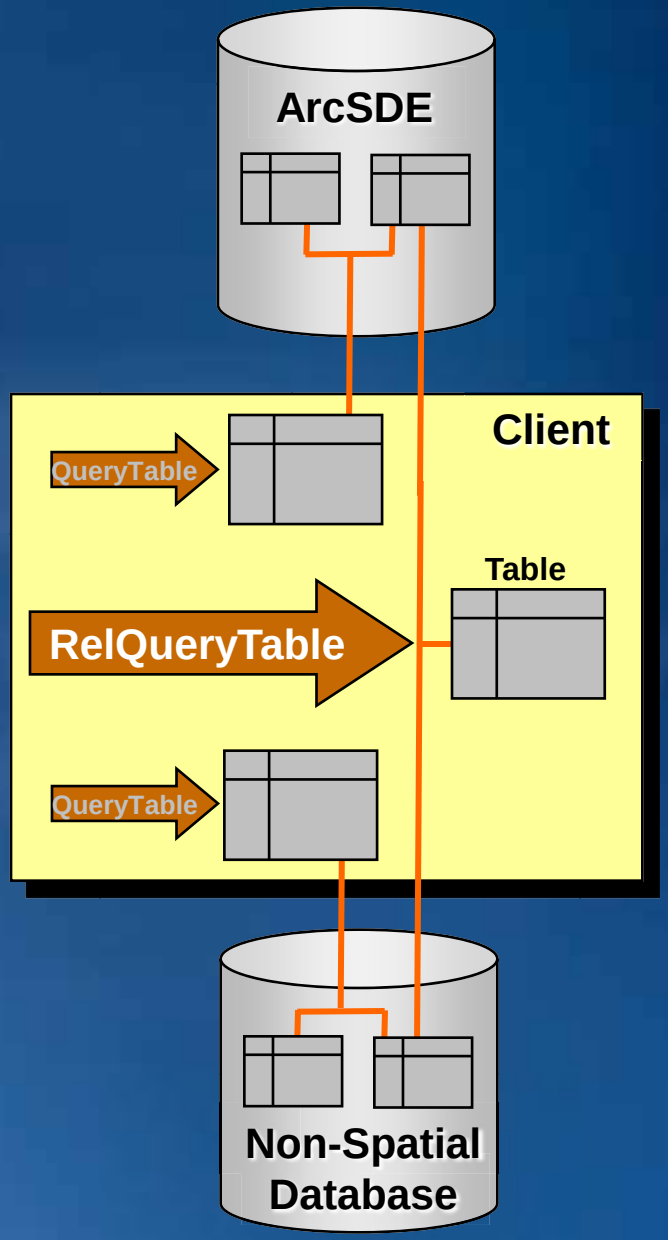
```
// Make the new TableQueryName
IQueryName2 qn2 = (IQueryName2)new TableQueryName();
qn2.QueryDef = qdef;
qn2.PrimaryKey = "ObjectID";
qn2.CopyLocally = false;

// Set the workspace and name of the new QueryTable
IDatasetName pDSName = (IDatasetName)qn2;
pDSName.WorkspaceName = WSName;
pDSName.Name = TableName;

// Open and return the table
IName name = (IName)qn2;
ITable table = (ITable)name.Open();
```

On-The-Fly Joins

- QueryTables (`TableQueryName`)
  - Tables must be within same datasource
  - Matches all candidates
  - Uses QueryDef object
  - Can be used with non-spatial tables
- RelQueryTables (`IRelQueryTable`)
  - Tables can be in different datasources
  - Matches only first candidate on 1:M join
  - Uses in-memory or relationship classes



Example: RelQueryTables (IRelQueryTable)

- Steps to create a join via IRelQueryTable
  1. Create a new memory RelationshipClass
     - Passing in the ObjectClasses that need to be joined
     - An existing relationship class can be used
  2. Call open on RelQueryTableFactory casting to ITable
     - Passing in the relationship class

```
// build a memoryrelationshipclass
IMemoryRelationshipClassFactory mRCfactory = new
MemoryRelationshipClassFactoryClass();

// open the memoryrelationshipclass
1 IRelationshipClass memRC = mRCfactory.Open("memrc", targetObjectClass,
fromField, joinObjectClass, toField, "forward", "backward",
esriRelCardinality.esriRelCardinalityOneToOne);

// Open the relquerytable as a table
IRelQueryTableFactory rqtfactory = new RelQueryTableFactoryClass();

2 ITable rqTable = (ITable)rqtfactory.Open(memRC, true, null, null, "",
false, true);
```

Editing and Transactions

Geodatabase & DBMS Transactions

Transaction Models

- **Non-versioned**
  - Utilizes database transactions
  - Transaction scope controlled through `StartEditing` and `StopEditing` calls
- **Versioned**
  - Utilizes database transactions (extended as long transaction)
  - Provides read consistency and isolation
  - Geodatabase transaction is defined by the scope of the edit session
  - Changes are only viewable by the editor until the edit session is explicitly saved
  - Provides undo and redo
  - Transaction scope controlled through `StartEditOperation` and `StopEditOperation` calls

Geodatabase & DBMS Transactions

Outside of an Edit Session (i.e. Updating)

- **`ITransactions` interface – `StartTransaction`, `CommitTransaction`, `AbortTransaction`, `InTransaction`**
- **`ITransactionsOptions` interface – `AutoCommitInterval`**
  - Number of modifications before a commit is automatically executed
  - Default is 1000 (server config)
  - 0 == no automatic commits (developers should set to 0 if server is configured with appropriate rollback resources)
- **DBMS rules apply – DDL may commit DML**

Geodatabase & DBMS Transactions

Within a Non-Versioned Edit Session (i.e. Editing)

- The logical transaction
  - `StartMultiuserEditing(NVES) == StartTransaction`
  - `StopEditing(true)  == CommitTransaction`
  - `StopEditing(false) == AbortTransaction`
- `StartMultiUserEditing` turns off auto commit (set to 0)

```
muWEdit.StartMultiuserEditing(esriMultiuserEditSessionMode.esriMESMNonVersioned);
```

- Editing errors need to be handled around the API committing the transaction – if error need to call `StopEditing(false)`
- Don't mix use of `ITransaction` and `IMultiUserWorkspaceEdit`

Geodatabase & DBMS Transactions

Within a Versioned Edit Session (i.e. Editing)

- The logical transaction
  - `StartEditOperation == StartTransaction`
  - `StopEditOperation == CommitTransaction`
  - `AbortEditOperation == AbortTransaction`
- `StartMultiUserEditing` turns off auto commit (set to 0)

```
muWEdit.StartMultiuserEditing(esriMultiuserEditSessionMode.esriMESMVersioned);
```

- Editing errors need to be handled around the API committing the transaction – if error need to call `AbortEditOperation`

Geodatabase & DBMS Transactions

Developer Considerations

- **Geodatabase "DDL" and other objects which invoke it may cause a commit**
  - CreateFeatureClass
  - Geoprocessing Tools
  - Selection in Map over 100 features (may call create table)
  - Etc…
- **Transactions are per "connection" not per Version**

Geodatabase & DBMS Transactions

Transaction Scope

- **Non-Versioned Edit Session**
  - Make many calls to `StopEditing(true)`
  - Edit Operations are no ops

```
muWEdit.StartMultiuserEditing(esriMESMNonVersioned);
          // Edit Code
workspaceEdit.StopEditing(true);
```

} Database Transaction

- **Versioned Edit Session**
  - Avoid excessive calls to `StopEditing(True)`

```
muWEdit.StartMultiuserEditing(esriMESMVersioned);
  workspaceEdit.StartEditOperation();
          // Edit Code
  workspaceEdit.StopEditOperation();
          //…
  workspaceEdit.StartEditOperation();
          // Edit Code
  workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
```

} Database Transaction

} Database Transaction

Example: Editing a Versioned Geodatabase

- Steps to making a version edit
  1. Get a reference to the `version\workspace` that is to be edited
  2. Use `IMultiUserWorkspaceEdit` to start the appropriate edit session
  3. Use `IWorkspaceEdit.StartOperation` to start an edit operation
  4. Make edit
  5. Stop operation and stop edit session saving edits

Versioned Editing

1. Get a reference to the `Version` that is to be edited

   Open the `FeautreClass` to be edited from this version

```
IFeatureWorkspace fWorkspace =
(IFeatureWorkspace)versionedWorkspace.FindVersion("SDE.DEFAULT");

IFeatureClass parcelClass = fWorkspace.OpenFeatureClass(“ParcelClass");

IWorkspaceEdit workspaceEdit = (IWorkspaceEdit)featureWorkspace;
IMultiuserWorkspaceEdit muWEdit =  (IMultiuserWorkspaceEdit)featureWorkspace;

if(muWEdit.SupportsMultiuserEditSessionMode(esriMESMVersioned))
{
muWEdit.StartMultiuserEditing(esriMESMVersioned);
      workspaceEdit.StartEditOperation();
          IFeature feature = parcelClass.GetFeature(2);
          feature.set_Value(8, "TEG");
          feature.Store();
       workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
}
```

Versioned Editing

- ❷ Use `IMultiUserWorkspaceEdit` to start the appropriate edit session

```
IFeatureWorkspace fWorkspace =
(IFeatureWorkspace)versionedWorkspace.FindVersion("SDE.DEFAULT");

IFeatureClass parcelClass = fWorkspace.OpenFeatureClass(“ParcelClass");

IWorkspaceEdit workspaceEdit = (IWorkspaceEdit)featureWorkspace;
IMultiuserWorkspaceEdit muWEdit =  (IMultiuserWorkspaceEdit)featureWorkspace;

if(muWEdit.SupportsMultiuserEditSessionMode(esriMESMVersioned))
{
muWEdit.StartMultiuserEditing(esriMESMVersioned);
      workspaceEdit.StartEditOperation();
          IFeature feature = parcelClass.GetFeature(2);
          feature.set_Value(8, "TEG");
          feature.Store();
       workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
}
```

Versioned Editing

❸ Use `IWorkspaceEdit.StartOperation` to start an edit operation

```
IFeatureWorkspace fWorkspace =
(IFeatureWorkspace)versionedWorkspace.FindVersion("SDE.DEFAULT");

IFeatureClass parcelClass = fWorkspace.OpenFeatureClass(“ParcelClass");

IWorkspaceEdit workspaceEdit = (IWorkspaceEdit)featureWorkspace;
IMultiuserWorkspaceEdit muWEdit =  (IMultiuserWorkspaceEdit)featureWorkspace;

if(muWEdit.SupportsMultiuserEditSessionMode(esriMESMVersioned))
{
muWEdit.StartMultiuserEditing(esriMESMVersioned);
      workspaceEdit.StartEditOperation();
          IFeature feature = parcelClass.GetFeature(2);
          feature.set_Value(8, "TEG");
          feature.Store();
       workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
}
```

Versioned Editing

4. Make edit(s) to the open feature class

```
IFeatureWorkspace fWorkspace =
(IFeatureWorkspace)versionedWorkspace.FindVersion("SDE.DEFAULT");

IFeatureClass parcelClass = fWorkspace.OpenFeatureClass(“ParcelClass");

IWorkspaceEdit workspaceEdit = (IWorkspaceEdit)featureWorkspace;
IMultiuserWorkspaceEdit muWEdit =  (IMultiuserWorkspaceEdit)featureWorkspace;

if(muWEdit.SupportsMultiuserEditSessionMode(esriMESMVersioned))
{
muWEdit.StartMultiuserEditing(esriMESMVersioned);
      workspaceEdit.StartEditOperation();
          IFeature feature = parcelClass.GetFeature(2);
          feature.set_Value(8, "TEG");
          feature.Store();
       workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
}
```

Versioned Editing

- ❺ Stop operation and stop edit session saving edits

```
IFeatureWorkspace fWorkspace =
(IFeatureWorkspace)versionedWorkspace.FindVersion("SDE.DEFAULT");

IFeatureClass parcelClass = fWorkspace.OpenFeatureClass(“ParcelClass");

IWorkspaceEdit workspaceEdit = (IWorkspaceEdit)featureWorkspace;
IMultiuserWorkspaceEdit muWEdit =  (IMultiuserWorkspaceEdit)featureWorkspace;

if(muWEdit.SupportsMultiuserEditSessionMode(esriMESMVersioned))
{
muWEdit.StartMultiuserEditing(esriMESMVersioned);
      workspaceEdit.StartEditOperation();
          IFeature feature = parcelClass.GetFeature(2);
          feature.set_Value(8, "TEG");
          feature.Store();
       workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
}
```

Versioned Editing

- Gotchas
  - Feature class needs to be open in the version before edit session is started

```
IFeatureWorkspace fWorkspace =
(IFeatureWorkspace)versionedWorkspace.FindVersion("SDE.DEFAULT");

IFeatureClass parcelClass = fWorkspace.OpenFeatureClass("ParcelClass");

IWorkspaceEdit workspaceEdit = (IWorkspaceEdit)featureWorkspace;
IMultiuserWorkspaceEdit muWEdit =  (IMultiuserWorkspaceEdit)featureWorkspace;

if(muWEdit.SupportsMultiuserEditSessionMode(esriMESMVersioned))
{
muWEdit.StartMultiuserEditing(esriMESMVersioned);
      workspaceEdit.StartEditOperation();
          IFeature feature = parcelClass.GetFeature(2);
          feature.set_Value(8, "TEG");
          feature.Store();
       workspaceEdit.StopEditOperation();
workspaceEdit.StopEditing(true);
}
```

Versioned Editing

Reconcile and Post

- **Ability to merge changes from one version to another**

```
// To reconcile and post the changes, we need to start a
// new edit session.
muWorkspaceEdit.StartMultiuserEditing(esriMultiuserEditSessionMode.esriMESMVersioned);

// Cast the workspace to the IVersionEdit4 interface and reconcile the changes
// with the QA version.
IVersionEdit4 versionEdit4 = (IVersionEdit4)workspace;
Boolean conflictsDetected = versionEdit4.Reconcile4("QA", true, true, false, true);

// CanPost indicates that posting can occur, post the changes to the QA version.
if (versionEdit4.CanPost())
{
      versionEdit4.Post("QA");
      workspaceEdit.StopEditing(true);
}
else
{
      // Stop the edit session, without saving any edits.
      workspaceEdit.StopEditing(false);
}
```

Archiving

Archiving with the Geodatabase

- The ability to archive edits made to versioned feature classes
- The archives will support historical queries
  - Display the data at a particular time or over a time span
- Historical Version another type of version
  - Classes can be opened from a historical version
- Queries go against the archive table
  - May be faster than querying Default

Archiving API

- Querying a class open in a historical version

```
IHistoricalWorkspace hWorkspace = (IHistoricalWorkspace)pWksp;
IHistoricalVersion hVersion = hWorkspace.FindHistoricalVersionByTimeStamp(date);

IFeatureWorkspace featureWorkspace = (IFeatureWorkspace)hVersion;
IFeatureClass parcel = featureWorkspace.OpenFeatureClass("ParcelClass");
```

- Querying the archive class directly

```
IArchivableClass parcel =
(IArchivableClass)featureWorkspace.OpenFeatureClass("ParcelClass");
ITable parcelArchive = parcel.Archive;

IQueryFilter queryFilter = new QueryFilterClass();
queryFilter.SubFields = “APN";
queryFilter.WhereClause = “APN = 12336 AND GDB_FROM_DATE > '2007-03-10
20:01:48.000'";
```

Difference Cursors

Difference Cursors

What are they?

- A cursor of object differences between versions
- Common APIs which create difference cursors
  - `IVersionTable.Differences`
- Returns a `IDifferenceCursor` Object
- Can be used for Historical or Transactional versions
- Difference type must be specified

Difference Cursors

Transactional Versions

- Returns differences between any two versioned tables
  - **`IRow`** object not returned for **`DeleteNoChange`** difference case
  - To get deleted row use **`IVersion2.GetCommonAncestor`**

```
// get delete no change differences between the two versions
IDifferenceCursor differenceCursor = childTable.Differences(parentTable,
                                                esriDifferenceTypeDeleteNoChange,
                                                null);

differenceCursor.Next(out oid, out differenceRow);
featureWorkspace =
(IFeatureWorkspace)childVersion.GetCommonAncestor(parentVersion);
caFeatureClass = featureWorkspace.OpenFeatureClass("parcelClass");

while (oid != -1)
{
      differenceRow = caFeatureClass.GetFeature(oid);
      Console.WriteLine("The difference row has an OID of {0}", oid);

      differenceCursor.Next(out oid, out differenceRow);
}
```

Difference Cursors

Historical Versions

- Does not support all differences
  - Supported for **`Insert, UpdateNoChange, DeleteNoChange`**
  - **`IRow`** object not returned for **`DeleteNoChange`** difference case

```
// get insert differences between the two historical versions
IDifferenceCursor differenceCursor = hvYearEnd2006.Differences(hvQ12007,
                                                               esriDifferenceTypeInsert,
                                                               null);

differenceCursor.Next(out oid, out differenceRow);

while (oid != -1)
{
    Console.WriteLine("The difference row has an OID of {0}", oid);
    differenceCursor.Next(out oid, out differenceRow);
}
```

Differences Within An Edit Session

EditDataChanges

- Ability to determine what's changed
  - Within a Edit Session or Edit Operation
  - Returns difference cursor
- Works on all types of geodatabases

```
// get insert differences in the edit session

IDataChangesEx editDataChanges;
editDataChanges = workspaceEdit2.get_EditDataChanges(WithinSession);

ESRI.ArcGIS.esriSystem.IEnumBSTR enumBSTR = editDataChanges.ModifiedClasses;
string changedClass = enumBSTR.Next();

IDifferenceCursorEx differenceCursor;
differenceCursor = editDataChanges.ExtractEx(changedClass, TypeInsert);
```

Events

Version Events

- **IVersionEvents / IVersionEvents2**
  - Used for executing specific code blocks when events fire
  - Reconcile, Post, Version Redefine, Archive Update events

```
//event handler method
private void versionEvent_OnReconcile(string targetVersionName, bool
  HasConflicts)
{
        //filter false conflicts from conflict classes
}

//event handler method
private void versionEvent_OnArchiveUpdated(string targetVersionName, bool
  HasConflicts)
{
        //modify user maintained date fields in the archive
}
```

Code Instrumentation and Tracing

Code instrumentation and tracing

What is it?

- **Code Instrumentation[1]**
  - **Adding code to an application for debug**
  - **Time certain operations in your application**
- **Tracing**
  - **More information about what applications are doing**
  - **Using existing tools to trace**
  - **Instrument code to produce own trace files**
- **Allows a look at how an application is performing**

1- Tom Kyte blog post http://tkyte.blogspot.com/2005/06/instrumentation.html

Code instrumentation and tracing

- Why do you need it?
  - Check how efficient the application works with the database
    - e.g. number of calls and the performance of individual operations
  - Info about deployed applications
  - Determine application transaction times
  - Identify bottlenecks
  - Testing overall performance

- Symptoms to look out for
  - Repeated calls to the database
  - Slow running operations

Code instrumentation and tracing

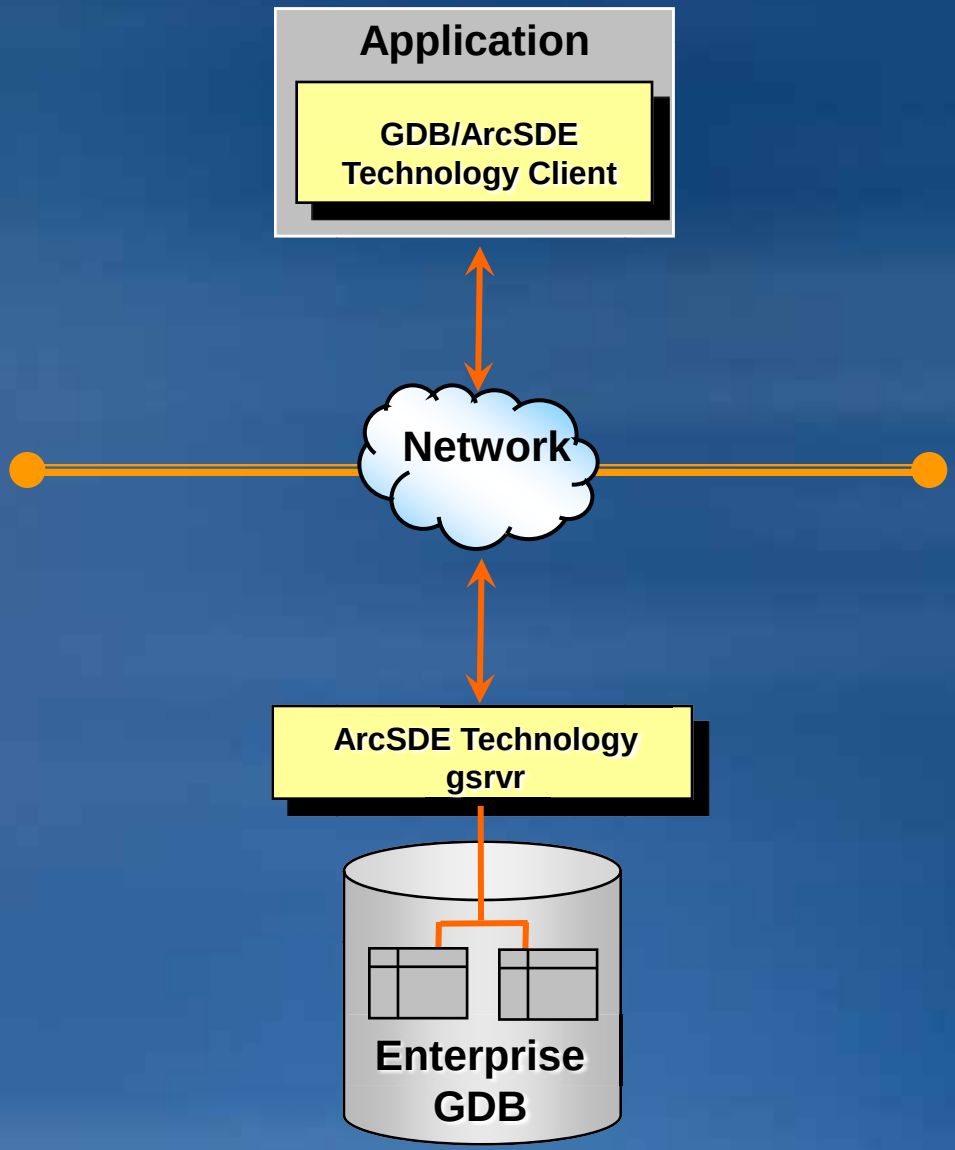
Ways to accomplish

- **Custom applications**
  - **Add code to time critical operations**
    - **Timers around particular functions or business logic**
    - **Data fetching**
  - **Enable DBMS tracing from app**
- **Built-in and already available**
  - **DBMS trace files**
  - **ArcSDE intercept**
  - **ArcSDE/ArcIMS/ArcGIS Server logs**

Code instrumentation and tracing

Client

Server



Example: Fire Danger Application

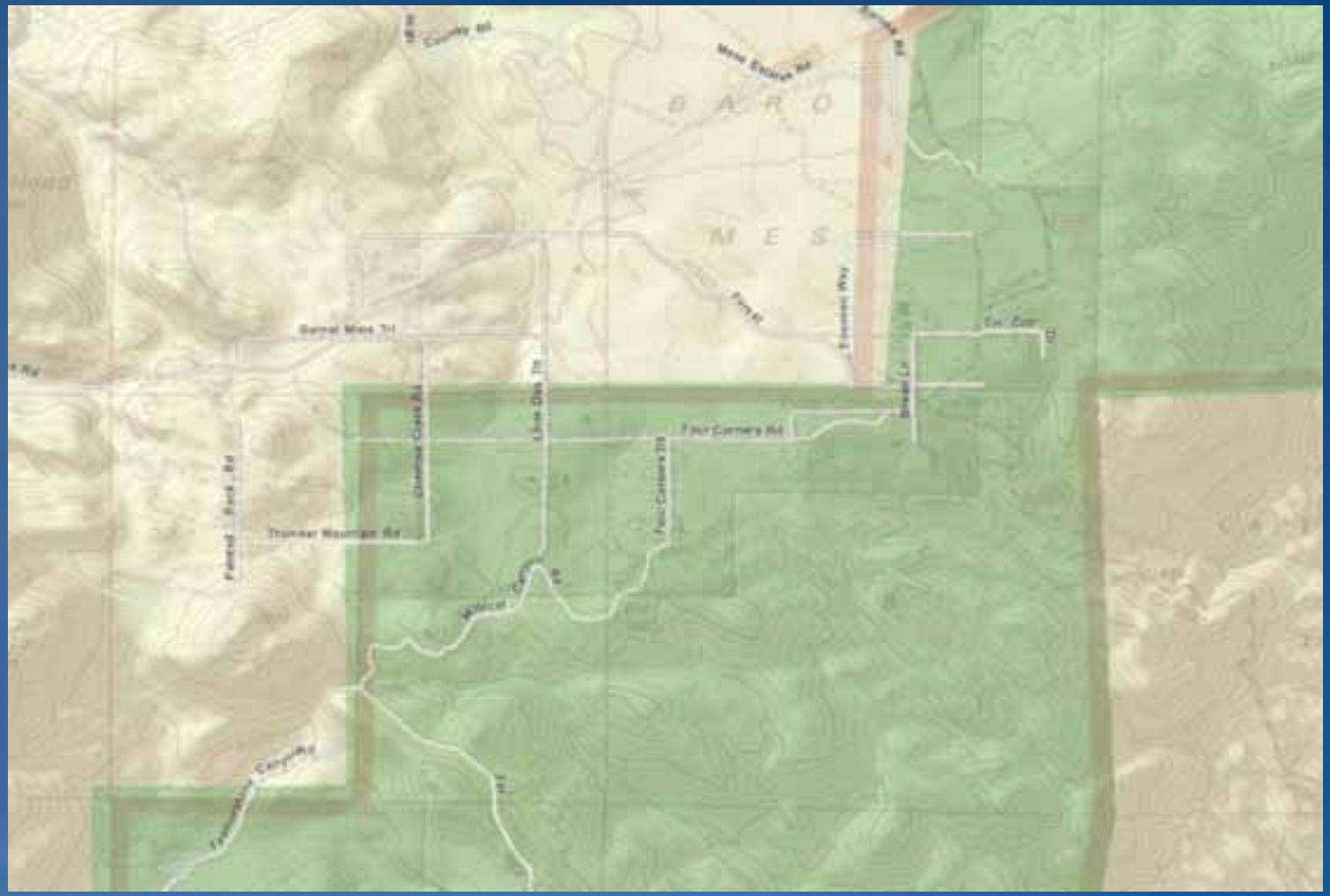
- Application receives a boundary of fire
- Then determines what parcels are in danger
- Reports affected parcels on map

Example: Fire Danger Application

- Business logic
  - Fire boundary imported
  - Algorithm to determine buffer distance
  - Buffer performed
  - Intersection of buffer with parcels

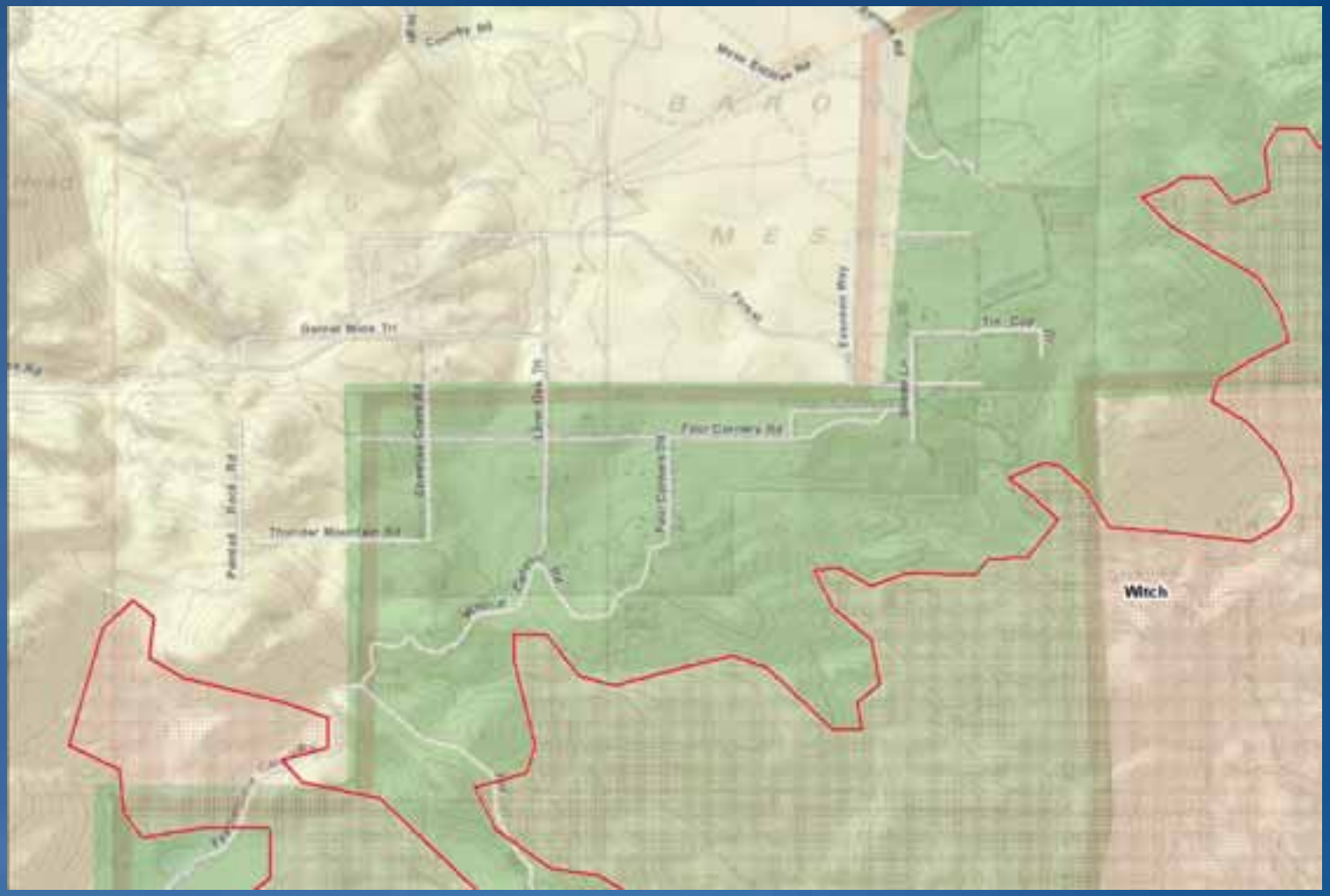
Example: Fire Danger Application

- **Business logic: coded function()**
  - **Fire boundary imported: impFireShp(inFC)**



Example: Fire Danger Application

- **Business logic: coded function()**
  - **Fire boundary imported: impFireShp(inFC)**



Example: Fire Danger Application

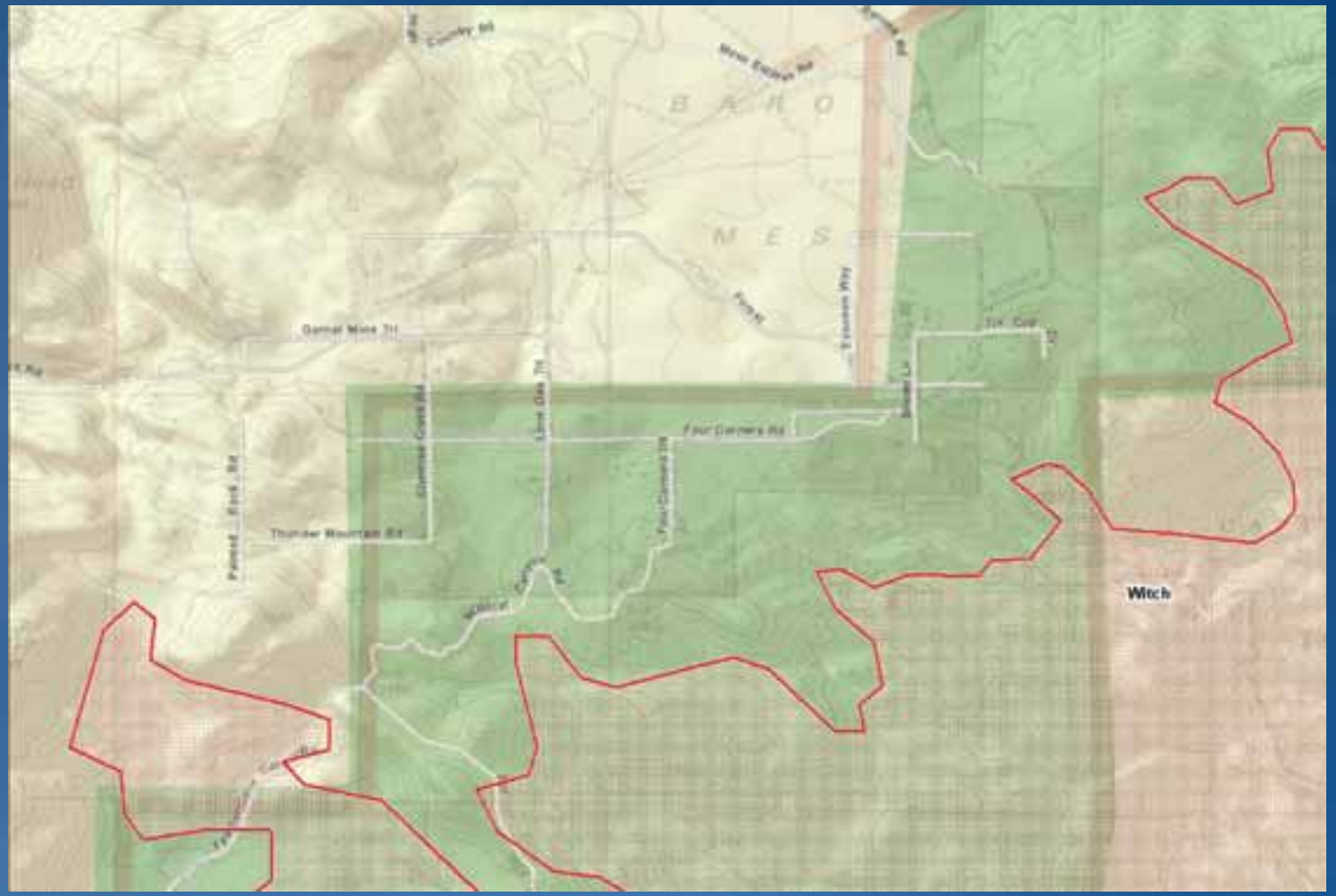
- **Business logic: coded function()**
  - **Algorithm to determine buffer distance, uses a model:**

    calcBuffer(inVeg, inBound, inWeather, inTopo, …)

http://www.fs.fed.us/psw/topics/fire_science/craft/craft/Resources/BBN_templates.htm

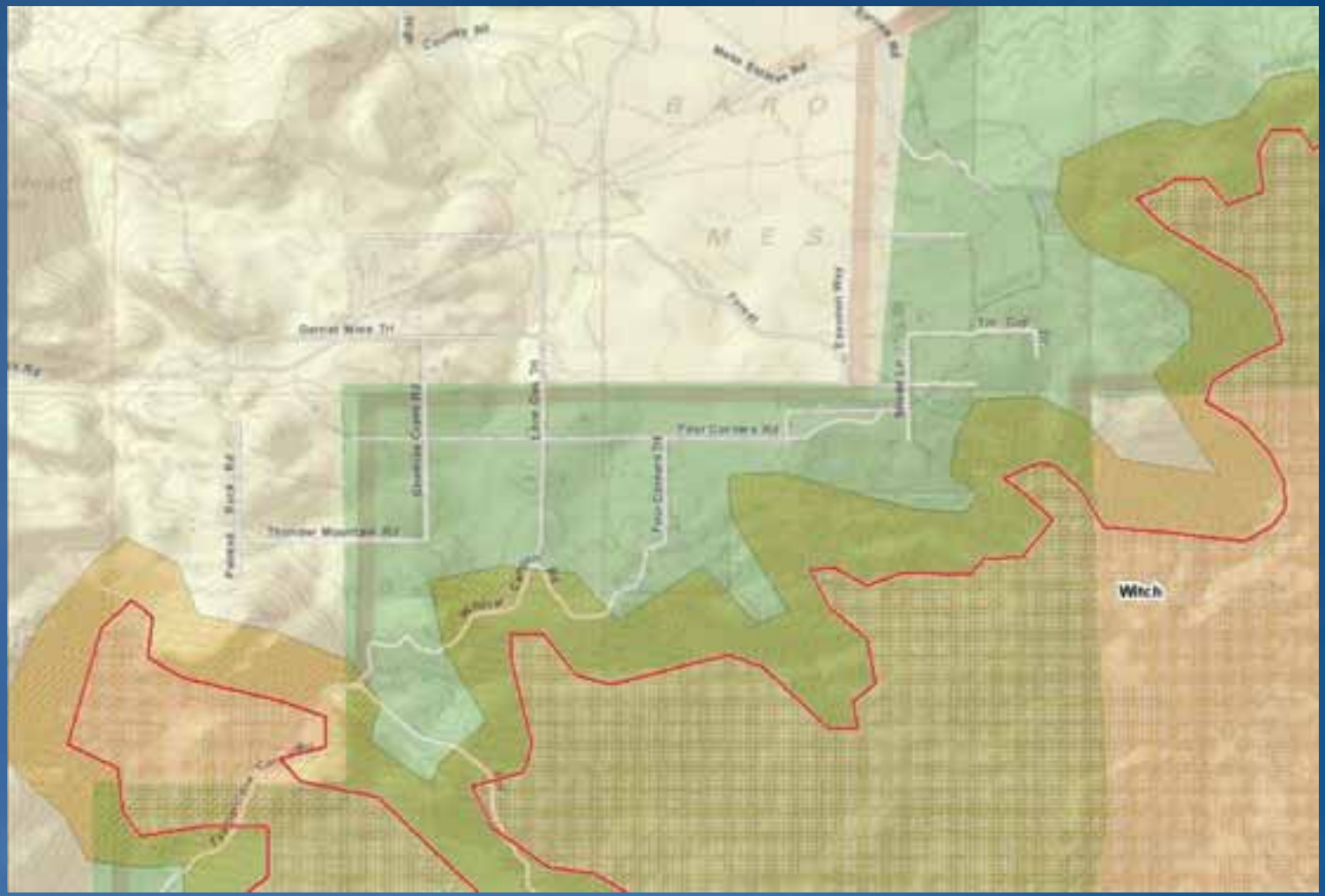
Example: Fire Danger Application

- Business logic: coded function()
  - Buffer performed: runBuffer(inFC, inDistance)



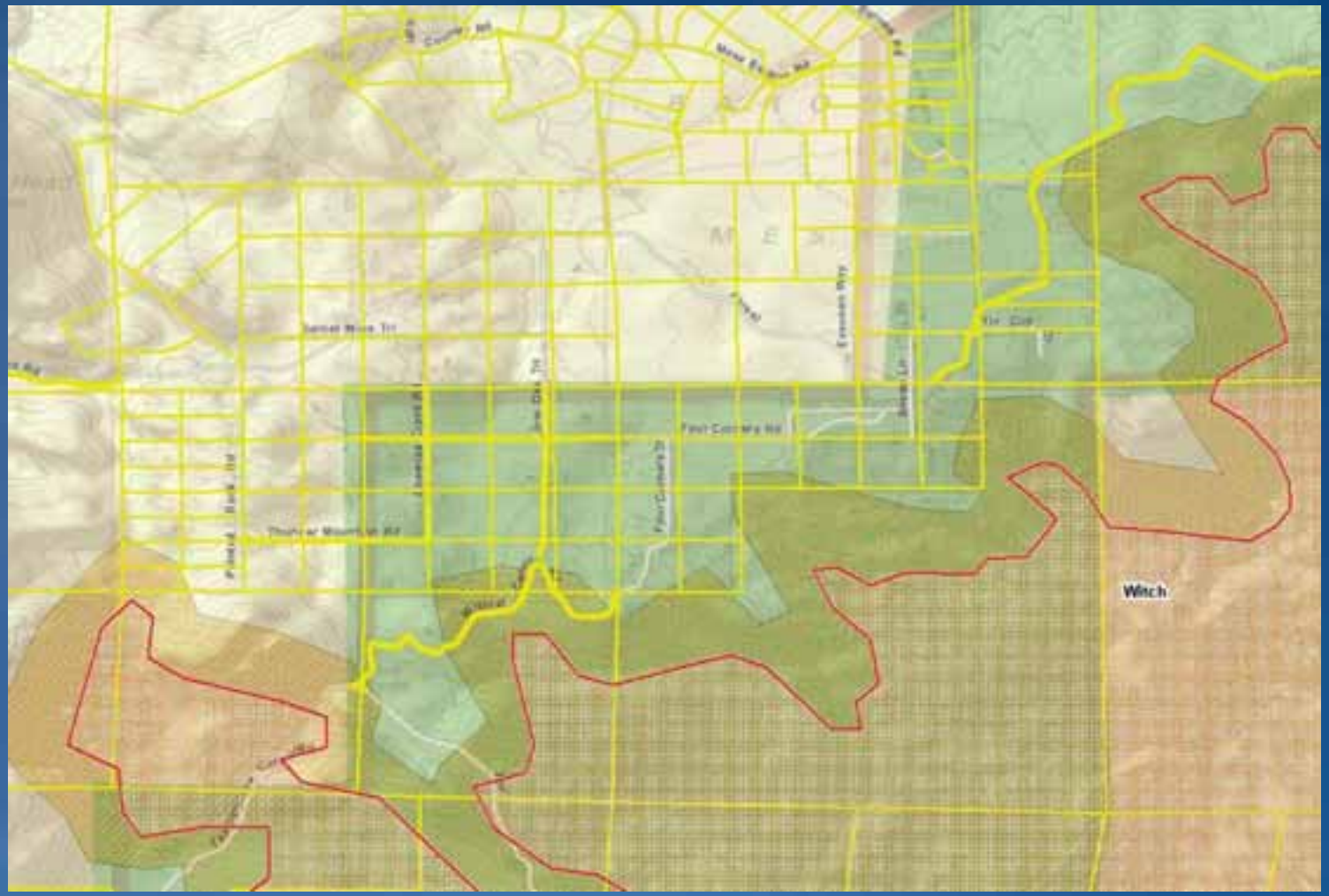
Example: Fire Danger Application

- **Business logic: coded function()**
  - **Buffer performed: runBuffer(inFC, inDistance)**



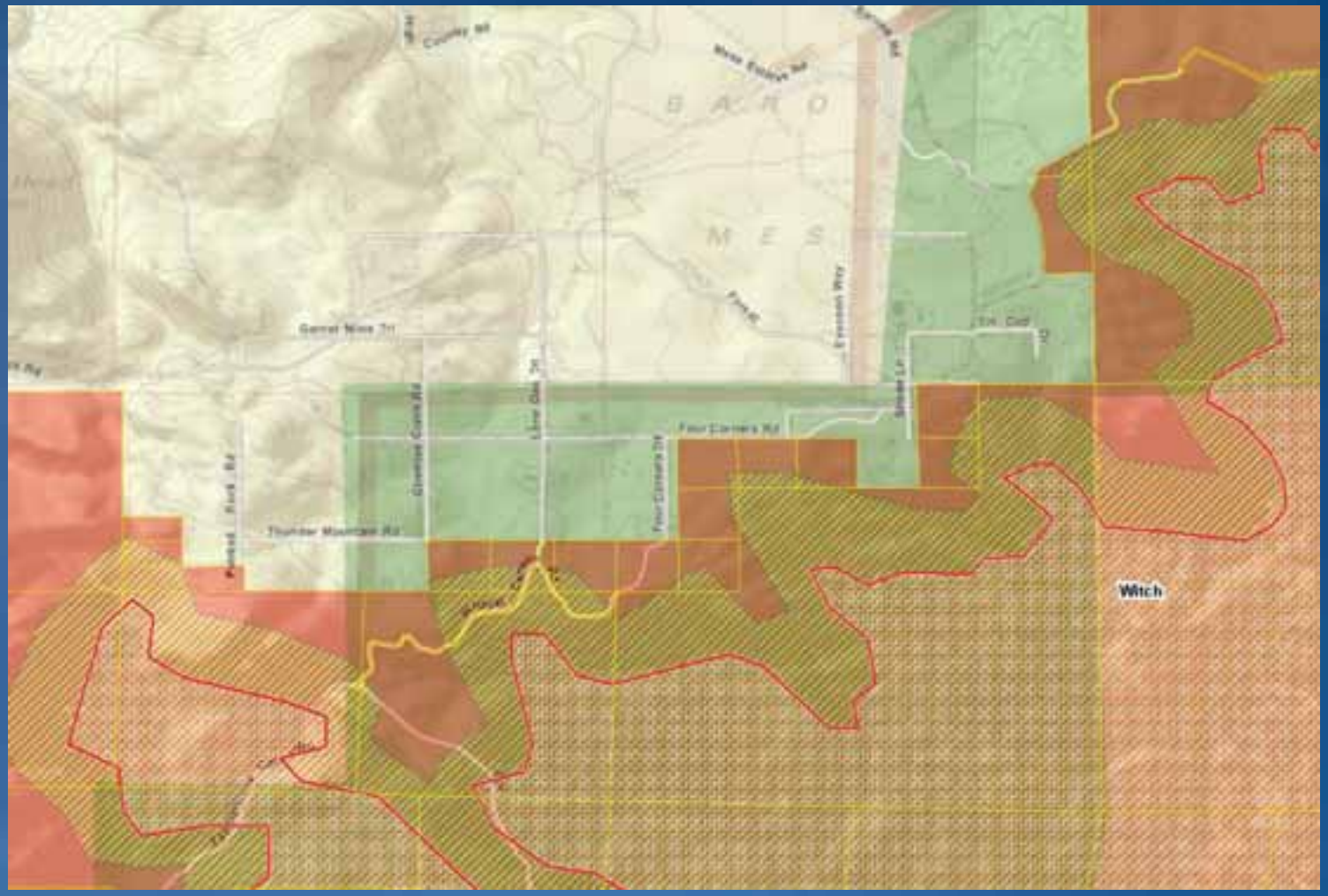
Example: Fire Danger Application

- Business logic: coded function()
  - Intersection of buffer with parcels: findParcels(inBufShp, inParcelFC)



Example: Fire Danger Application

- Business logic: coded function()
  - Intersection of buffer with parcels: findParcels(inBufShp, inParcelFC)



Example: Fire Danger Application

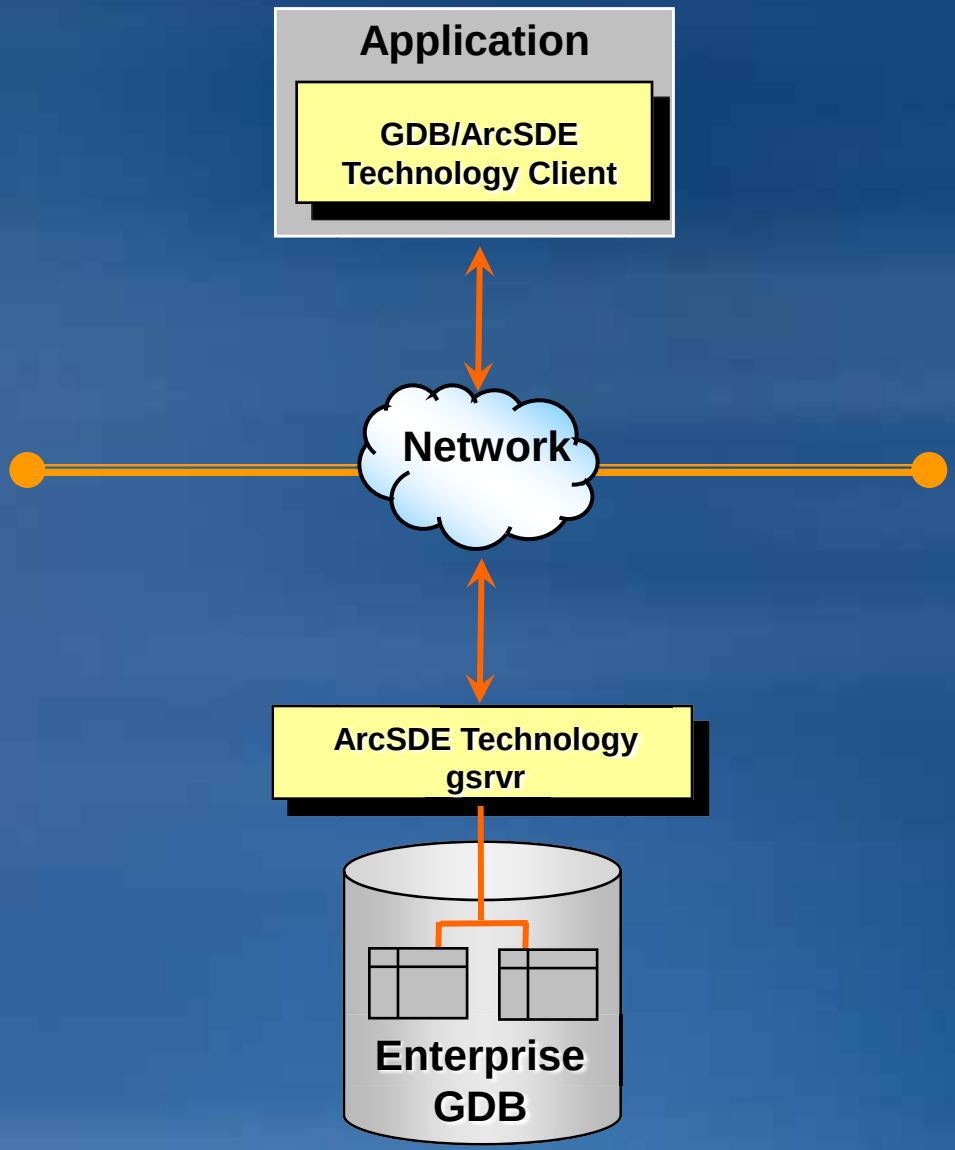
- Business logic
  - Fire boundary imported
  - Algorithm to determine buffer distance
  - Buffer performed
  - Intersection of buffer with parcels

- Application is performing slow

Code instrumentation and tracing

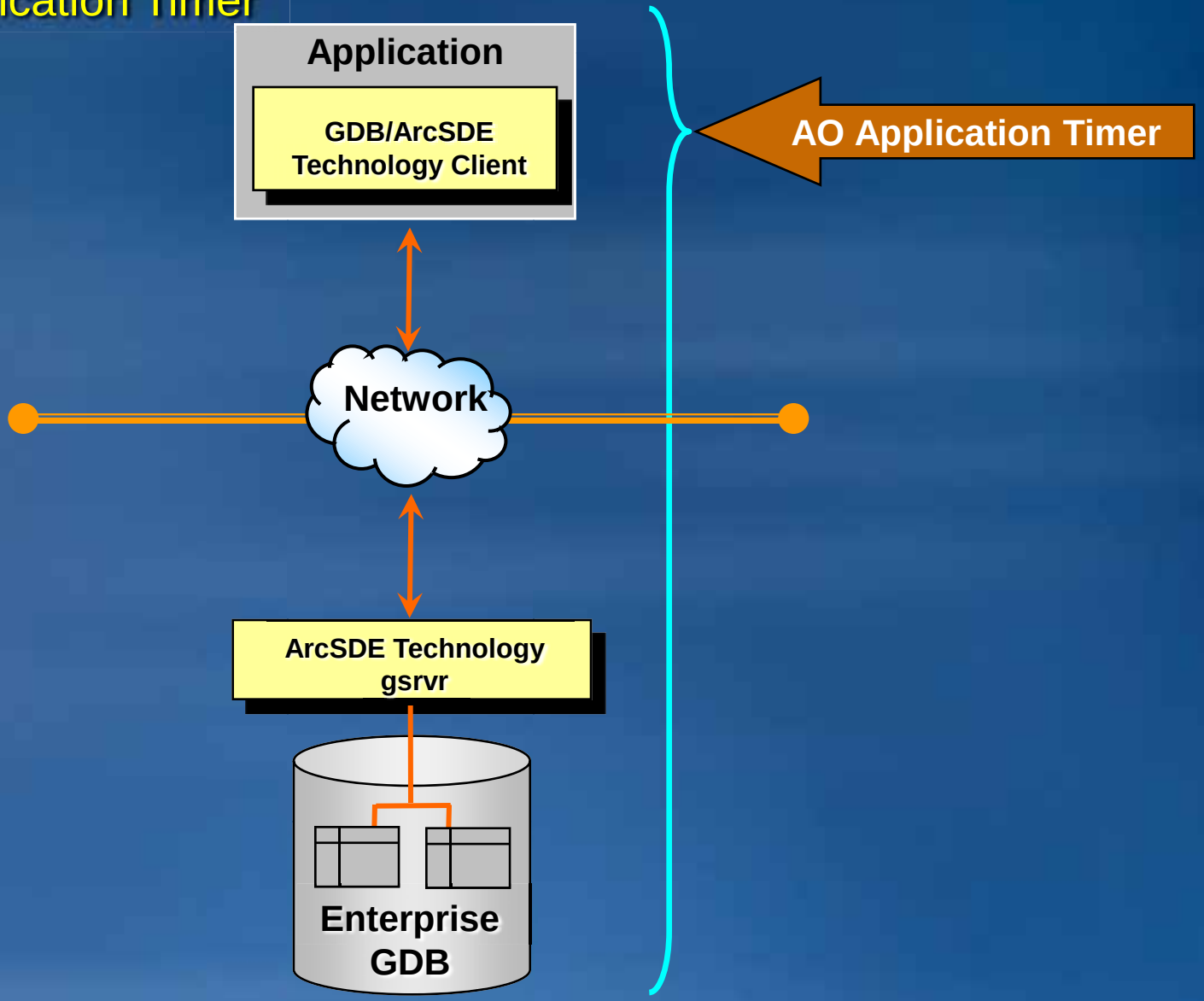
Client

Server



Code instrumentation and tracing

ArcObjects Application Timer



Example: Fire Danger Application

ArcObjects Application Timer

- Using .NET Stopwatch
- Wrap a timer around the coded functions

```
using System.Diagnostics;
//Create a stopwatch
Stopwatch stopWatch = new Stopwatch();
```

1

http://msdn.microsoft.com/en-us/library/system.diagnostics.stopwatch.aspx

Example: Fire Danger Application

ArcObjects Application Timer

- Using .NET Stopwatch
- Wrap a timer around the coded functions

```
using System.Diagnostics;
//Create a stopwatch
1 Stopwatch stopWatch = new Stopwatch();

stopWatch.Start();
2     IFeatureClass parcelsInDangerFC = findParcels(inBufShp,inParcelFC);
stopWatch.Stop();
```

http://msdn.microsoft.com/en-us/library/system.diagnostics.stopwatch.aspx

Example: Fire Danger Application

ArcObjects Application Timer

- Using .NET Stopwatch
- Wrap a timer around the coded functions

```
using System.Diagnostics;
//Create a stopwatch
Stopwatch stopWatch = new Stopwatch();

stopWatch.Start();
     IFeatureClass parcelsInDangerFC = findParcels(inBufShp,inParcelFC);
stopWatch.Stop();

// Get the elapsed time as a TimeSpan value.
TimeSpan ts = stopWatch.Elapsed;

// Format and display the TimeSpan value.
string elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",
ts.Hours, ts.Minutes, ts.Seconds,
ts.Milliseconds / 10);

Console.WriteLine(elapsedTime, "RunTime");
```

1
2
3

http://msdn.microsoft.com/en-us/library/system.diagnostics.stopwatch.aspx

Example: Fire Danger Application

ArcObjects Application Timer

- Using .NET Stopwatch
- Wrap a timer around the coded functions

```
using System.Diagnostics;
//Create a stopwatch
1 Stopwatch stopWatch = new Stopwatch();

stopWatch.Start();
2     IFeatureClass parcelsInDangerFC = findParcels(inBufShp,inParcelFC);
stopWatch.Stop();

// Get the elapsed time as a TimeSpan value.
TimeSpan ts = stopWatch.Elapsed;

3 // Format and display the TimeSpan value.
string elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",
ts.Hours, ts.Minutes, ts.Seconds,
ts.Milliseconds / 10);

Console.WriteLine(elapsedTime, "RunTime");

4 //Repeat for other functions
```

http://msdn.microsoft.com/en-us/library/system.diagnostics.stopwatch.aspx

Example: Fire Danger Application

ArcObjects Application Timer

- **Business logic**

Operation	Elapsed Time (HH:MM:SS.MS)
Fire boundary imported	00:00:03.56
Algorithm to determine buffer distance	00:00:01.19
Buffer performed	00:00:07.63
Intersection of buffer with parcels (findParcels)	00:00:30.72

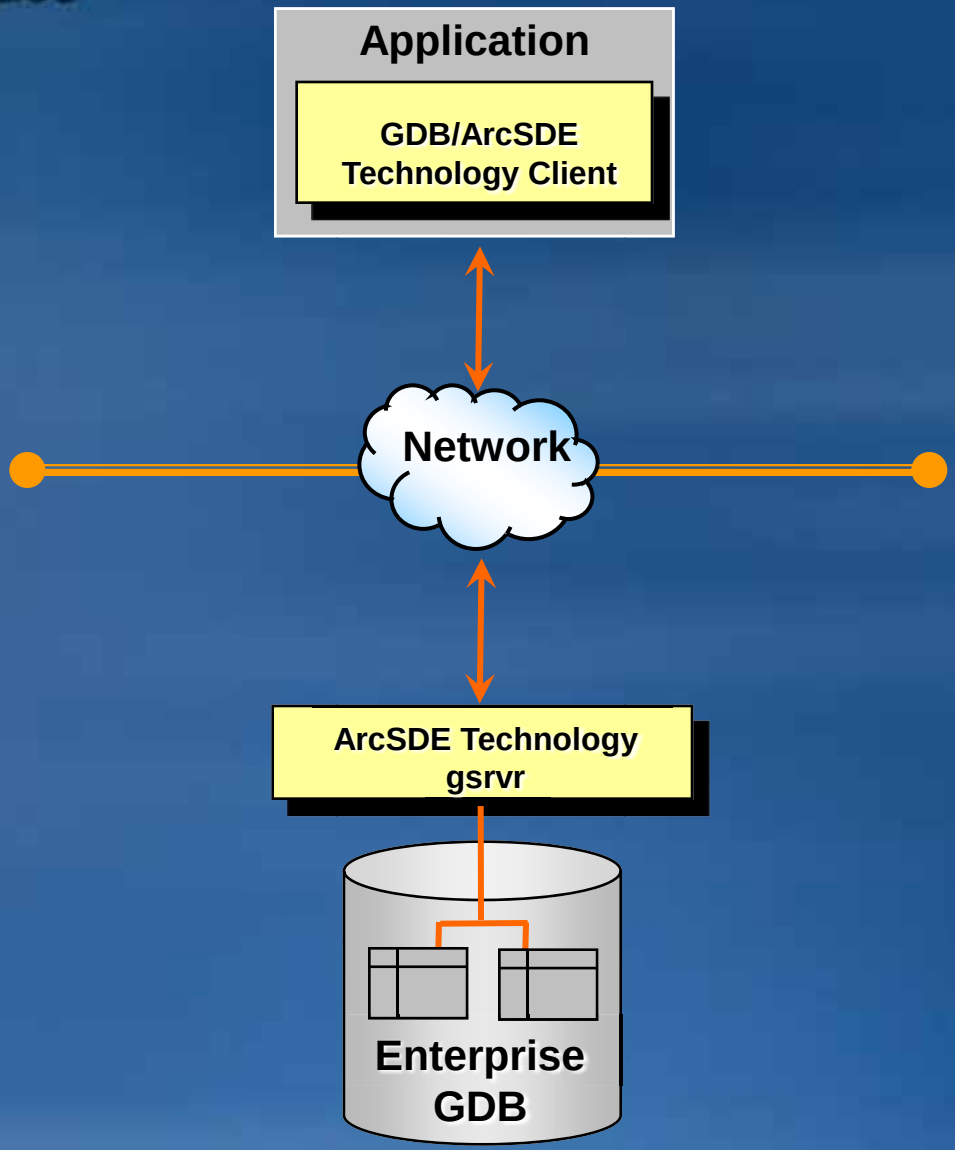
- **Use timers to determine where the bottleneck is**

Code instrumentation and tracing

Database Trace

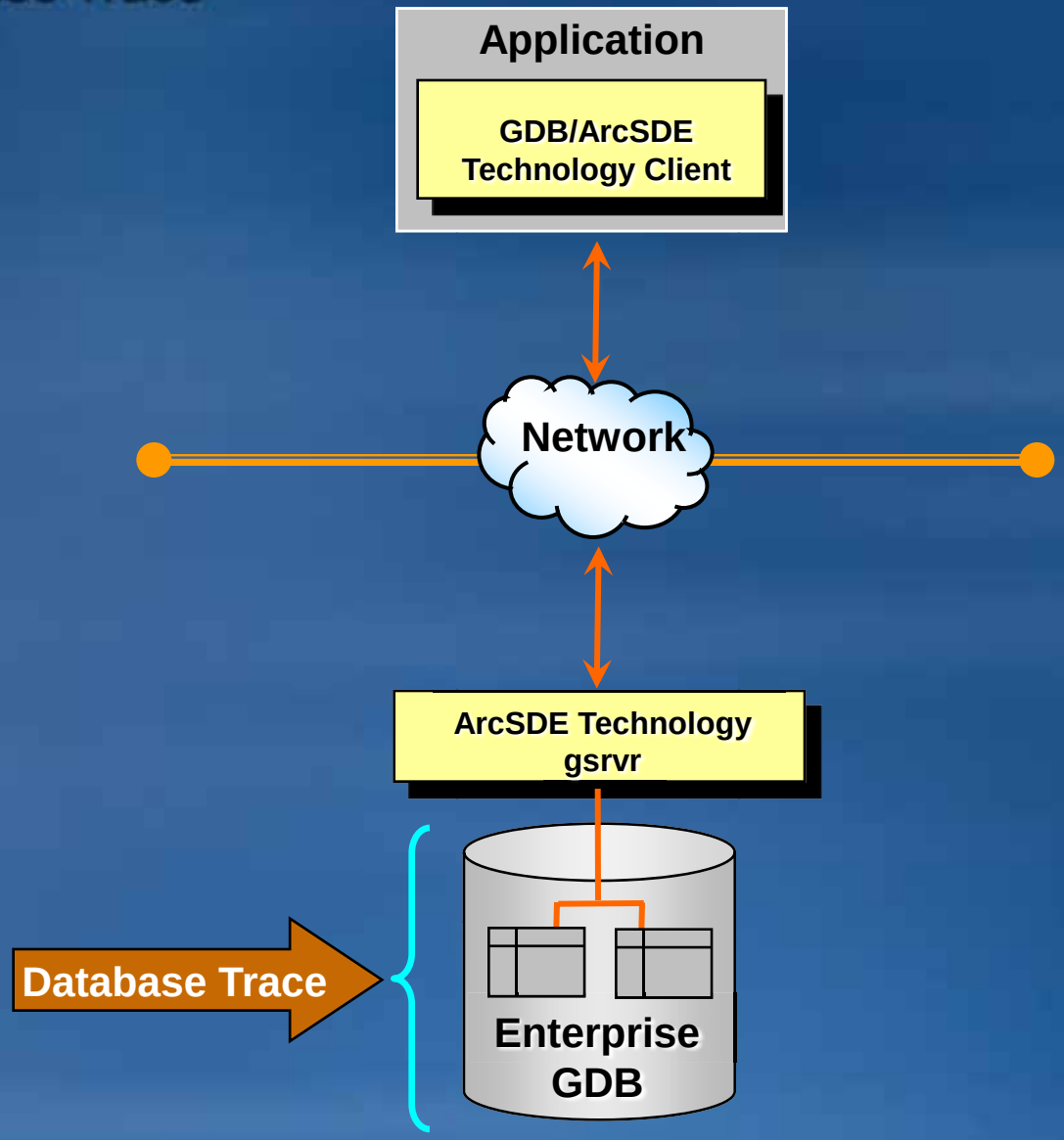
Client

Server



Code instrumentation and tracing

Database Trace



Code instrumentation and tracing

Database Trace

- Allows you to figure out how much time is being spent inside the DBMS and how individual SQL statements perform, e.g. Definition Queries
- What does an Oracle trace look like?
- Coding the application to perform DBMS traces
  - enables operations to be profiled in the database

Example: DBMS trace output on Oracle

Database Trace: Contents

- **TKPROF output contains SQL execution paths, row counts, wait events, and more.**
  - **For sql statements and overall trace session information**

```
OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        0      0.00       0.00          0          0          0           0
Execute     16      0.17       0.19          0          0          0           0
Fetch      189      2.96       3.53          0       1605          0       17820
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total      205      3.14       3.72          0       1605          0       17820

Misses in library cache during parse: 0

Elapsed times include waiting on following events:
  Event waited on                             Times   Max. Wait  Total Waited
  ----------------------------------------   Waited  ----------  ------------
  SQL*Net message to client                      200        0.00          0.00
  SQL*Net message from client                    199        0.52          2.50
  SQL*Net more data to client                    923        0.00          0.09
  direct path read                                44        0.05          0.15
  log file sync                                   12        0.00          0.00
```

Example: DBMS trace output on Oracle

Database Trace: Contents

- TKPROF output contains SQL execution paths, row counts, wait events, and more.
  - For sql statements and overall trace session information

```
OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        0      0.00       0.00          0          0          0           0
Execute     16      0.17       0.19          0          0          0           0
Fetch      189      2.96       3.53          0       1605          0       17820
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total      205      3.14       3.72          0       1605          0       17820

Misses in library cache during parse: 0

Elapsed times include waiting on following events:
  Event waited on                             Times   Max. Wait  Total Waited
  ----------------------------------------   Waited  ----------  ------------
  SQL*Net message to client                      200        0.00          0.00
  SQL*Net message from client                    199        0.52          2.50
  SQL*Net more data to client                    923        0.00          0.09
  direct path read                                44        0.05          0.15
  log file sync                                   12        0.00          0.00
```

Example: DBMS trace output on Oracle

Database Trace: Contents

- **TKPROF output contains SQL execution paths, row counts, wait events, and more.**

Column heading values are described at the beginning of the TKPROF output

```
********************************************************************************
count    = number of times OCI procedure was executed
cpu      = cpu time in seconds executing
elapsed  = elapsed time in seconds executing
disk     = number of physical reads of buffers from disk
query    = number of buffers gotten for consistent read
current  = number of buffers gotten in current mode (usually for update)
rows     = number of rows processed by the fetch or execute call
********************************************************************************
```

- **Good reference book:**
  - **"Expert One-on-One Oracle" by Thomas Kyte**

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

Operation	Elapsed Time (HH:MM:SS.MS)
Fire boundary imported	00:00:03.56
Algorithm to determine buffer distance	00:00:01.19
Buffer performed	00:00:07.63
Intersection of buffer with parcels (findParcels)	00:00:30.72

- Profile the findParcels function in DBMS

Example: enable DBMS trace

Database Trace: Programmatically

- Determine DBMS flavor with IDatabaseConnectionInfo2.ConnectionDBMS
- Use the IWorkspace.ExecuteSQL to alter the database session to begin tracing

```
//Find the dbms flavor you are connected to construct a SQL Trace Statement
String sqlStatement;
```

NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: enable DBMS trace

Database Trace: Programmatically

- Determine DBMS flavor with IDatabaseConnectionInfo2.ConnectionDBMS
- Use the IWorkspace.ExecuteSQL to alter the database session to begin tracing

```
//Find the dbms flavor you are connected to construct a SQL Trace Statement
String sqlStatement;

if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_Oracle)
 sqlStatement = "alter session set events '10046 trace name context
  forever,  level 12'";

else if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_SQLServer)
 sqlStatement = "master..sp_trace_setstatus 1";

else if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_Informix)
 sqlStatement = "set explain on";

else return;
```

1

NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: enable DBMS trace

Database Trace: Programmatically

- Determine DBMS flavor with IDatabaseConnectionInfo2.ConnectionDBMS
- Use the IWorkspace.ExecuteSQL to alter the database session to begin tracing

```
//Find the dbms flavor you are connected to construct a SQL Trace Statement
String sqlStatement;

if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_Oracle)
 sqlStatement = "alter session set events '10046 trace name context
  forever,  level 12'";

else if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_SQLServer)
 sqlStatement = "master..sp_trace_setstatus 1";

else if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_Informix)
 sqlStatement = "set explain on";

else return;

//Execute the SQL statement on the Workspace
workspace.ExecuteSQL(sqlStatement);
```

1

2

NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: enable DBMS trace

Database Trace: Programmatically

- Determine DBMS flavor with IDatabaseConnectionInfo2.ConnectionDBMS
- Use the IWorkspace.ExecuteSQL to alter the database session to begin tracing

```
//Create a stopwatch
Stopwatch stopWatch = new Stopwatch();

//Perform operation to profile
stopWatch.Start();
    IFeatureClass parcelsInDangerFC = findParcels(inBufShp,inParcelFC);
stopWatch.Stop();
```

❸

NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: enable DBMS trace

Database Trace: Programmatically

- Determine DBMS flavor with IDatabaseConnectionInfo2.ConnectionDBMS
- Use the IWorkspace.ExecuteSQL to alter the database session to begin tracing

```
//Create a stopwatch
Stopwatch stopWatch = new Stopwatch();

//Perform operation to profile
stopWatch.Start();
     IFeatureClass parcelsInDangerFC = findParcels(inBufShp,inParcelFC);
stopWatch.Stop();

//Perform logic to determine how to turn trace off
if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_Oracle)
 sqlStatement = "alter session set events '10046 trace name context forever,
  level 0'";
```

❸

❹

NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: enable DBMS trace

Database Trace: Programmatically

- Determine DBMS flavor with IDatabaseConnectionInfo2.ConnectionDBMS
- Use the IWorkspace.ExecuteSQL to alter the database session to begin tracing

```
//Create a stopwatch
Stopwatch stopWatch = new Stopwatch();

//Perform operation to profile
stopWatch.Start();
     IFeatureClass parcelsInDangerFC = findParcels(inBufShp,inParcelFC);
stopWatch.Stop();

//Perform logic to determine how to turn trace off
if (dbmsInfo.ConnectionDBMS == esriConnectionDBMS.esriDBMS_Oracle)
 sqlStatement = "alter session set events '10046 trace name context forever,
  level 0'";


//Execute the SQL statement on the Workspace
workspace.ExecuteSQL(sqlStatement);
```

3

4

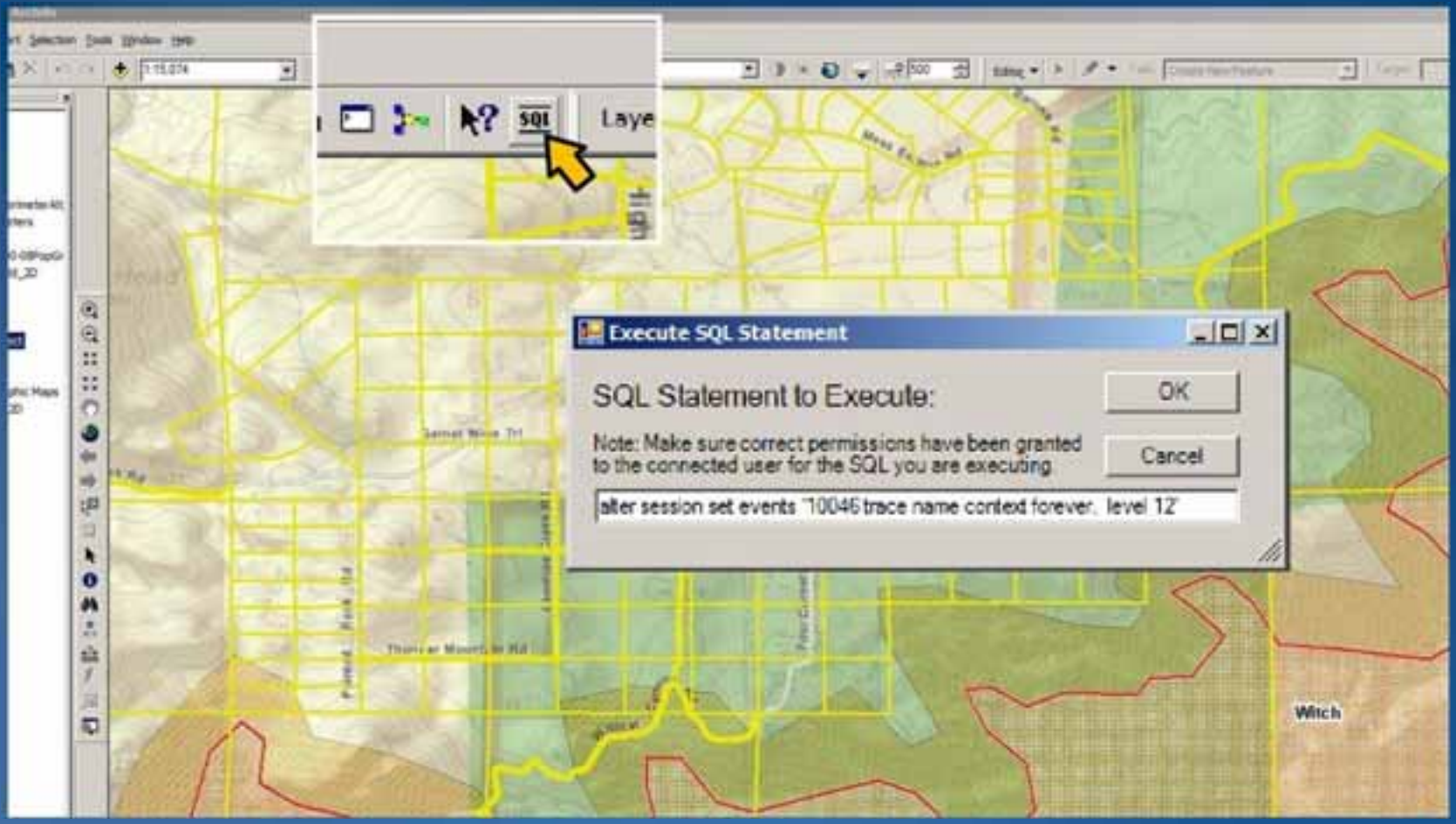
5

NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: enable DBMS trace

Database Trace: from ArcMap

- IWorkspace.ExecuteSQL GUI tool (Custom add-in command for ArcGIS)
  - Available at http://resources.esri.com in the ArcGIS Desktop DotNet Code Gallery
  - Search for "ExecuteSQL Command for ArcMap"



NOTE: Permissions may need to be granted for the connected user to start a trace from your application.

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query with the longest time
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query with the longest time

SQL ID: b8xarztp3ncmb
Plan Hash: 4281564849
SELECT  1 SHAPE, PARCEL.OBJECTID_1,  PARCEL.SHAPE.points,PARCEL.SHAPE.numpts,
  PARCEL.SHAPE.entity,PARCEL.SHAPE.minx,PARCEL.SHAPE.miny,PARCEL.SHAPE.maxx,
  PARCEL.SHAPE.maxy,PARCEL.rowid
FROM
  GDB.PARCEL PARCEL WHERE SDE.ST_EnvIntersects(PARCEL.SHAPE,:1,:2,:3,:4) = 1



call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.14       0.33          2       1004          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch       23     22.46      24.65      46014      48237          0        2268
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total       25     22.60      24.98      46016      49241          0        2268

Misses in library cache during parse: 1
Optimizer mode: ALL_ROWS
Parsing user id: 100

Rows     Row Source Operation
-------  ---------------------------------------------------
   2268  TABLE ACCESS FULL PARCEL (cr=48632 pr=46016 pw=0 time=0 us cost=12677
  size=18358624 card=7859)
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query with the longest time

SQL ID: b8xarztp3ncmb
Plan Hash: 4281564849
SELECT  1 SHAPE, PARCEL.OBJECTID_1,  PARCEL.SHAPE.points,PARCEL.SHAPE.numpts,
  PARCEL.SHAPE.entity,PARCEL.SHAPE.minx,PARCEL.SHAPE.miny,PARCEL.SHAPE.maxx,
  PARCEL.SHAPE.maxy,PARCEL.rowid
FROM
  GDB.PARCEL PARCEL WHERE SDE.ST_EnvIntersects(PARCEL.SHAPE,:1,:2,:3,:4) = 1



call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.14       0.33          2       1004          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch       23     22.46      24.65      46014      48237          0        2268
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total       25     22.60      24.98      46016      49241          0        2268

Misses in library cache during parse: 1
Optimizer mode: ALL_ROWS
Parsing user id: 100

Rows     Row Source Operation
-------  ---------------------------------------------------
   2268  TABLE ACCESS FULL PARCEL (cr=48632 pr=46016 pw=0 time=0 us cost=12677
  size=18358624 card=7859)
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query with the longest time

SQL ID: b8xarztp3ncmb
Plan Hash: 4281564849
SELECT  1 SHAPE, PARCEL.OBJECTID_1,  PARCEL.SHAPE.points,PARCEL.SHAPE.numpts,
  PARCEL.SHAPE.entity,PARCEL.SHAPE.minx,PARCEL.SHAPE.miny,PARCEL.SHAPE.maxx,
  PARCEL.SHAPE.maxy,PARCEL.rowid
FROM
  GDB.PARCEL PARCEL WHERE SDE.ST_EnvIntersects(PARCEL.SHAPE,:1,:2,:3,:4) = 1



call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.14       0.33          2       1004          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch       23     22.46      24.65      46014      48237          0        2268
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total       25     22.60      24.98      46016      49241          0        2268

Misses in library cache during parse: 1
Optimizer mode: ALL_ROWS
Parsing user id: 100

Rows     Row Source Operation
-------  ---------------------------------------------------
   2268  TABLE ACCESS FULL PARCEL (cr=48632 pr=46016 pw=0 time=0 us cost=12677
  size=18358624 card=7859)
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

- Full table scan, no spatial index
- Need to add spatial index
- ST Domain Index
  - Index used for ST_Geometry in Oracle
  - Contains statistics about the spatial index
- If index exists, but no statistics present:
  - `DOMAIN INDEX (Sel: Default - No Stats)`

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

- Add Spatial Index
  - ArcGIS or ArcObjects IClass.AddIndex Method
  - sdelayer -o normal_io command

- Gather statistics

```
SQL> exec dbms_stats.gather_table_stats(user,'PARCEL');
                PL/SQL procedure successfully completed.
```

- Check if statistics are present for ST Domain Index

```
SQL> select
BLEVEL,LEAF_BLOCKS,CLUSTERING_FACTOR,DENSITY,NUM_ROWS,LAST_ANALYZED from
SDE.ST_GEOMETRY_INDEX where table_name='PARCEL';

BLEVEL LEAF_BLOCKS CLUSTERING_FACTOR DENSITY  NUM_ROWS LAST_ANALYZED
------------ ---------------------- -------------------------------- ------------- ------------------- ------------------------
         2              15098                    622400      22.68        783159         25-MAR-09
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query AFTER Spatial Index added AND Stats Gathered
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query AFTER Spatial Index added AND Stats Gathered

SQL ID: b8xarztp3ncmb
Plan Hash: 1656122025
SELECT  1 SHAPE, PARCEL.OBJECTID_1,  PARCEL.SHAPE.points,PARCEL.SHAPE.numpts,
  PARCEL.SHAPE.entity,PARCEL.SHAPE.minx,PARCEL.SHAPE.miny,PARCEL.SHAPE.maxx,
  PARCEL.SHAPE.maxy,PARCEL.rowid
FROM
  GDB.PARCEL PARCEL WHERE SDE.ST_EnvIntersects(PARCEL.SHAPE,:1,:2,:3,:4) = 1


call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.00          0          0          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch       23      0.18       0.14          0       2851          0        2268
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total       25      0.18       0.15          0       2851          0        2268

Misses in library cache during parse: 0
Optimizer mode: ALL_ROWS
Parsing user id: 100

Rows     Row Source Operation
-------  ---------------------------------------------------
   2268  TABLE ACCESS BY INDEX ROWID PARCEL (cr=3112 pr=0 pw=0 time=0 us cost=9198
  size=5655532 card=11266)
   2268   DOMAIN INDEX (Sel: 1.4385563) A4_IX1 (cr=261 pr=0 pw=0 time=0 us cost=9198
  size=0 card=0)
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query AFTER Spatial Index added AND Stats Gathered

SQL ID: b8xarztp3ncmb
Plan Hash: 1656122025
SELECT  1 SHAPE, PARCEL.OBJECTID_1,  PARCEL.SHAPE.points,PARCEL.SHAPE.numpts,
  PARCEL.SHAPE.entity,PARCEL.SHAPE.minx,PARCEL.SHAPE.miny,PARCEL.SHAPE.maxx,
  PARCEL.SHAPE.maxy,PARCEL.rowid
FROM
  GDB.PARCEL PARCEL WHERE SDE.ST_EnvIntersects(PARCEL.SHAPE,:1,:2,:3,:4) = 1


call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.00          0          0          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch       23      0.18       0.14          0       2851          0        2268
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total       25      0.18       0.15          0       2851          0        2268

Misses in library cache during parse: 0
Optimizer mode: ALL_ROWS
Parsing user id: 100

Rows     Row Source Operation
-------  ---------------------------------------------------
   2268  TABLE ACCESS BY INDEX ROWID PARCEL (cr=3112 pr=0 pw=0 time=0 us cost=9198
  size=5655532 card=11266)
   2268   DOMAIN INDEX (Sel: 1.4385563) A4_IX1 (cr=261 pr=0 pw=0 time=0 us cost=9198
  size=0 card=0)
```

Example: DBMS trace output on Oracle

Database Trace: e.g. findParcels function

```
//ST_EnvIntersects Query AFTER Spatial Index added AND Stats Gathered

SQL ID: b8xarztp3ncmb
Plan Hash: 1656122025
SELECT  1 SHAPE, PARCEL.OBJECTID_1,  PARCEL.SHAPE.points,PARCEL.SHAPE.numpts,
  PARCEL.SHAPE.entity,PARCEL.SHAPE.minx,PARCEL.SHAPE.miny,PARCEL.SHAPE.maxx,
  PARCEL.SHAPE.maxy,PARCEL.rowid
FROM
  GDB.PARCEL PARCEL WHERE SDE.ST_EnvIntersects(PARCEL.SHAPE,:1,:2,:3,:4) = 1


call     count       cpu    elapsed       disk      query    current        rows
------- ------  -------- ---------- ---------- ---------- ----------  ----------
Parse        1      0.00       0.00          0          0          0           0
Execute      1      0.00       0.00          0          0          0           0
Fetch       23      0.18       0.14          0       2851          0        2268
------- ------  -------- ---------- ---------- ---------- ----------  ----------
total       25      0.18       0.15          0       2851          0        2268

Misses in library cache during parse: 0
Optimizer mode: ALL_ROWS
Parsing user id: 100

Rows     Row Source Operation
-------  ---------------------------------------------------
   2268  TABLE ACCESS BY INDEX ROWID PARCEL (cr=3112 pr=0 pw=0 time=0 us cost=9198
  size=5655532 card=11266)
   2268   DOMAIN INDEX (Sel: 1.4385563) A4_IX1 (cr=261 pr=0 pw=0 time=0 us cost=9198
  size=0 card=0)
```

Example: DBMS trace output on Oracle

Database Trace: ST Domain index statistics summary

ST_EnvIntersects Query	NO Spatial Index	Domain Index WITH Stats
Total count	2271	2271
Total cpu	22.60	0.18
Total elapsed time	24.98	0.15
Total disk	46016	0
Total query	49241	2851
Total current	0	0
Total rows	2268	2268

Example: DBMS trace output on Oracle

Database Trace: Performance Issue

- One step further
  - Big improvement
  - HOWEVER, is it possible to improve?
- Go back to application or examine trace more
  - Profile again if needed.

Example: Summary

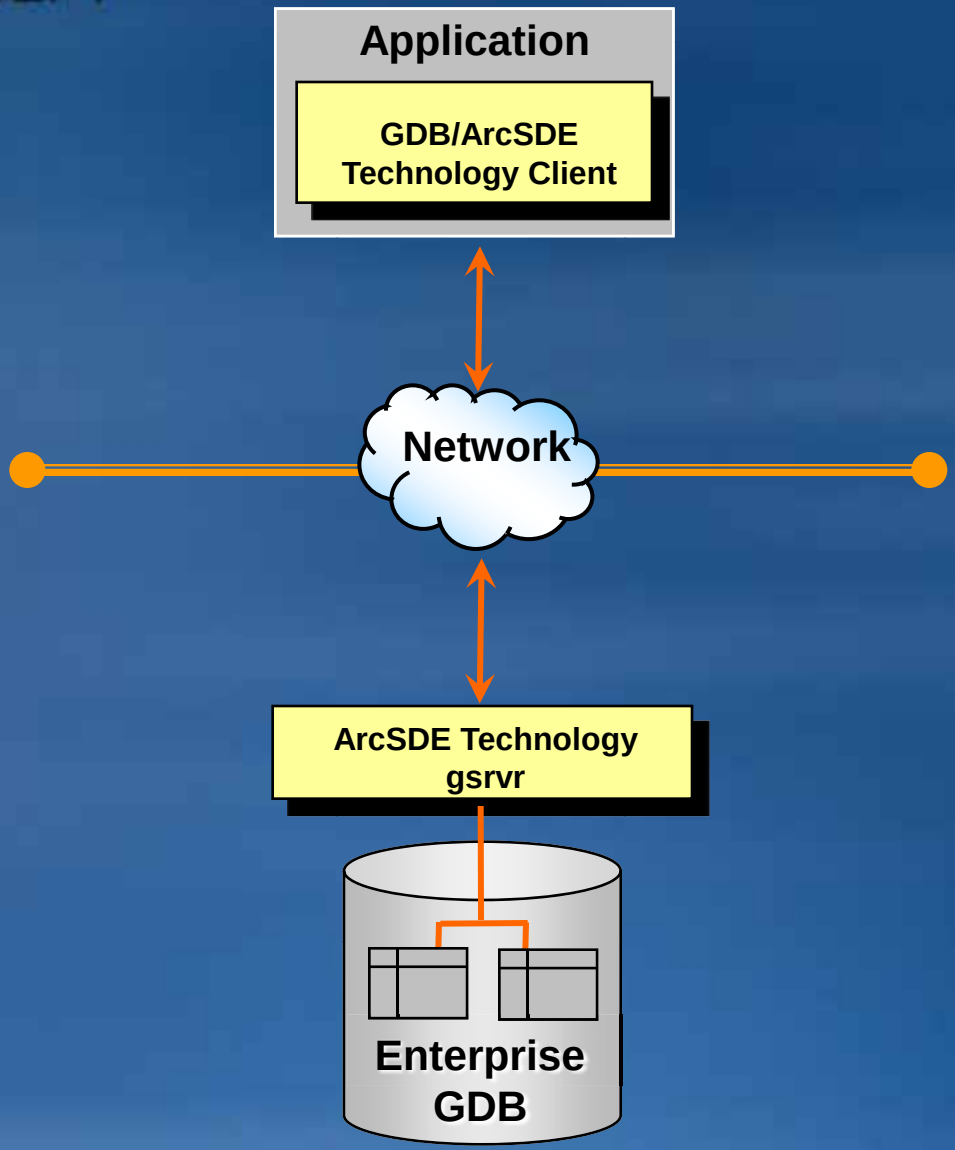
- Using application timer information
  - Determine what to trace in DBMS
- Fix issues
- Repeat
  - This is an iterative process
- Look at the entire application/system
  - Determine where bottlenecks are

Code instrumentation and tracing

SDEINTERCEPT

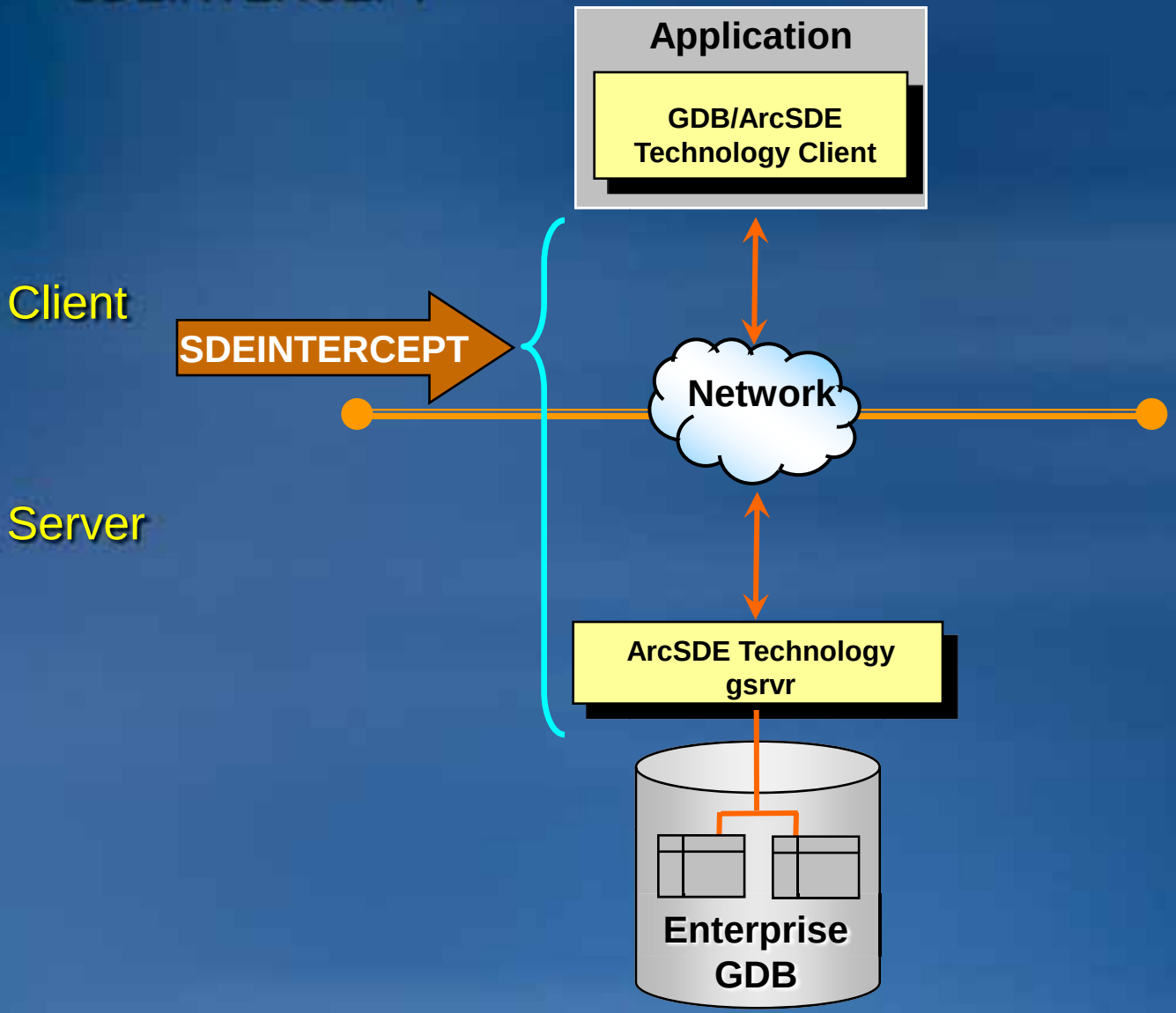
Client

Server



Code instrumentation and tracing

SDEINTERCEPT



Code instrumentation and tracing

SDEINTERCEPT

- Built-in ArcSDE functionality
- Logs the ArcSDE client calls to the ArcSDE server
- Useful to profile how many and what type of calls are made

Code instrumentation and tracing

SDEINTERCEPT

- Client side - only client session
- Server side - all sessions
- Based on connection to the SDE geodatabase

Code instrumentation and tracing

SDEINTERCEPT

- Client side - only client session
- Server side - all sessions
- Based on connection to the SDE geodatabase

- set SDEINTERCEPTLOC=<file location>
- set SDEINTERCEPT=<flags>

Code instrumentation and tracing

SDEINTERCEPT

- **Intercept flags**
  - **set SDEINTERCEPT=<flags>**
  - **Flags to intercept network broadcasts**
  - **Sets type and amount of information written to intercept file**
  - **SDEINTERCEPT flags not set, defaults to “crwf”**

Flag	Description
c	Intercept the API command name
r	Intercept the Channel broadcasts read-only
w	Intercept the Channel broadcasts write-only
t	Intercept log time (minute:second)
T	Intercept log time (hour:minute:second)
f	Intercept flush immediate

Example: SDEINTERCEPT setup

SDEINTERCEPT

- set SDEINTERCEPTLOC
- set SDEINTERCEPT
- Start application

1

```
REM enabling SDEINTERCEPT
REM Batch File to enable SDEINTERCEPT and Start ArcMap

set SDEINTERCEPTLOC=c:\temp\client_sdeintercept
```

Example: SDEINTERCEPT setup

SDEINTERCEPT

- set SDEINTERCEPTLOC
- set SDEINTERCEPT
- Start application

```
REM enabling SDEINTERCEPT
REM Batch File to enable SDEINTERCEPT and Start ArcMap

1 set SDEINTERCEPTLOC=c:\temp\client_sdeintercept
2 set SDEINTERCEPT=cwrfT
```

Example: SDEINTERCEPT setup

SDEINTERCEPT

- set SDEINTERCEPTLOC
- set SDEINTERCEPT
- Start application

```
REM enabling SDEINTERCEPT
REM Batch File to enable SDEINTERCEPT and Start ArcMap

set SDEINTERCEPTLOC=c:\temp\client_sdeintercept
set SDEINTERCEPT=cwrfT
C:\arcgis\bin\arcmap.exe
```

Example: SDEINTERCEPT Output

SDEINTERCEPT

```
========================================
[W 18:26:20.312] Command:      QueryWithInfo
[W 18:26:20.312] Long:         1
[W 18:26:20.312] Query Info:
  Num Columns:   1
  Columns:       "*"
  SQL_Construct: [1]
  Tables:        "GDB.TEST_NUM2"
  WhereClause:   "OBJECTID in (1)"
  Query Type:    4
  ByClause:      <null>
  Num Hints:     0
  Num Parameter markers: 0
  Logfile:       <null>
[R 18:26:20.312] Long:         0
[R 18:26:20.312] Col_Defines:  [2]
  Name                              Type     Width nDec  NULL?    RowID
  ------------------------------ ---------- ----- ---- -------- -----
  CALCU                          SE_INT32        10    0      NULL
  OBJECTID                       SE_INT32        10    0  NOT NULL  SDE
  ------------------------------ ---------- ----- ---- -------- -----
========================================
```

Example: SDEINTERCEPT Output

SDEINTERCEPT

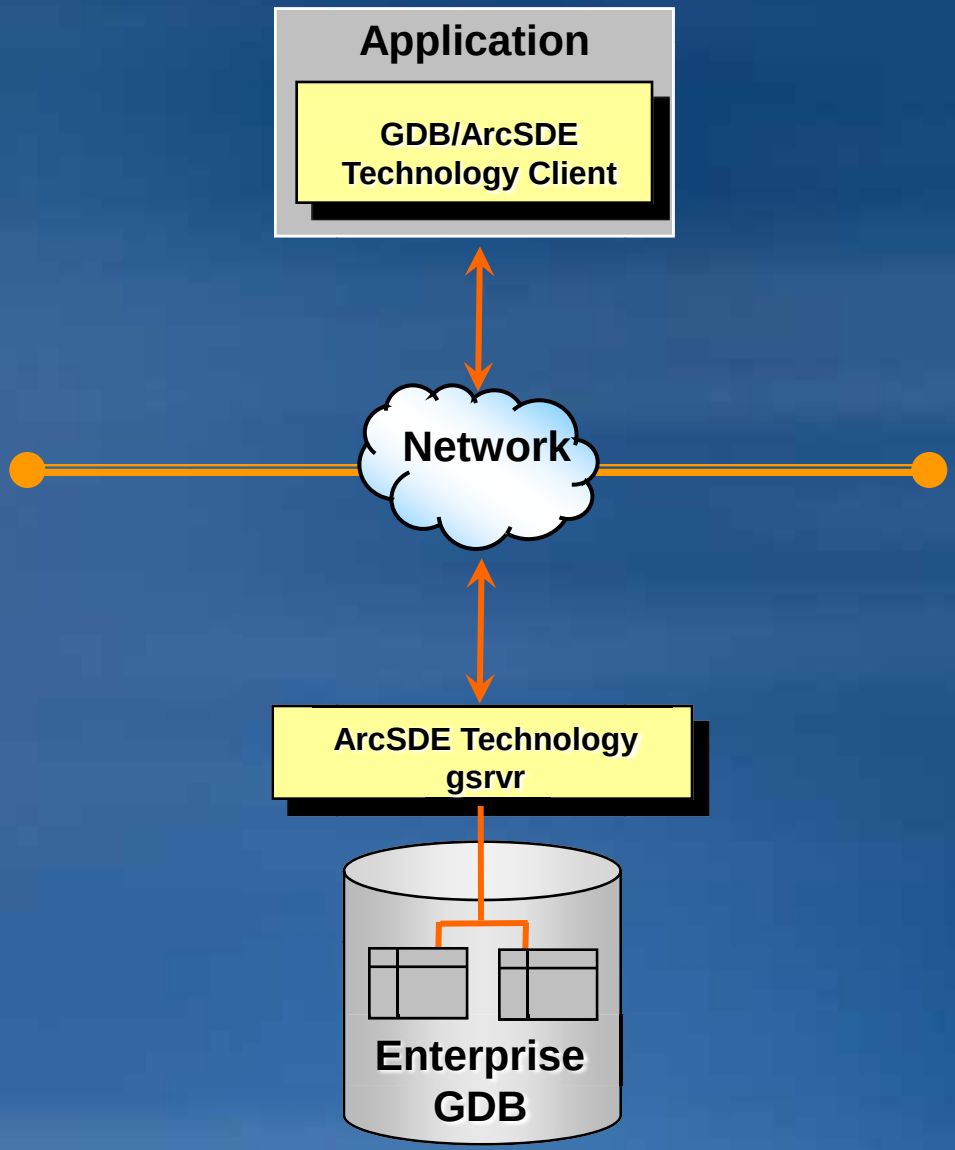
```
========================================
[W 18:26:20.312] Command:      ExecuteSpatialQuery
[W 18:26:20.312] Long:         1
[R 18:26:20.312] Long:         0
========================================
[W 18:26:20.312] Command:      NextBuffer
[W 18:26:20.328] Long:         1
[R 18:26:20.328] Long:         -51
========================================
[W 18:26:20.328] Command:      GetstreamError
[W 18:26:20.328] Long:         1
[R 18:26:20.328] Long:         -51
[R 18:26:20.328] Long:         1455
[R 18:26:20.328] NString:      "ORA-01455: converting column overflows
  integer datatype
"
[R 18:26:20.328] NString:      ""
========================================
[W 18:26:28.140] Command:      CloseStream
[W 18:26:28.140] Long:         1
[W 18:26:28.140] Long:         1
[R 18:26:28.140] Long:         0
========================================
```

Code instrumentation and tracing

Summary

Client

Server

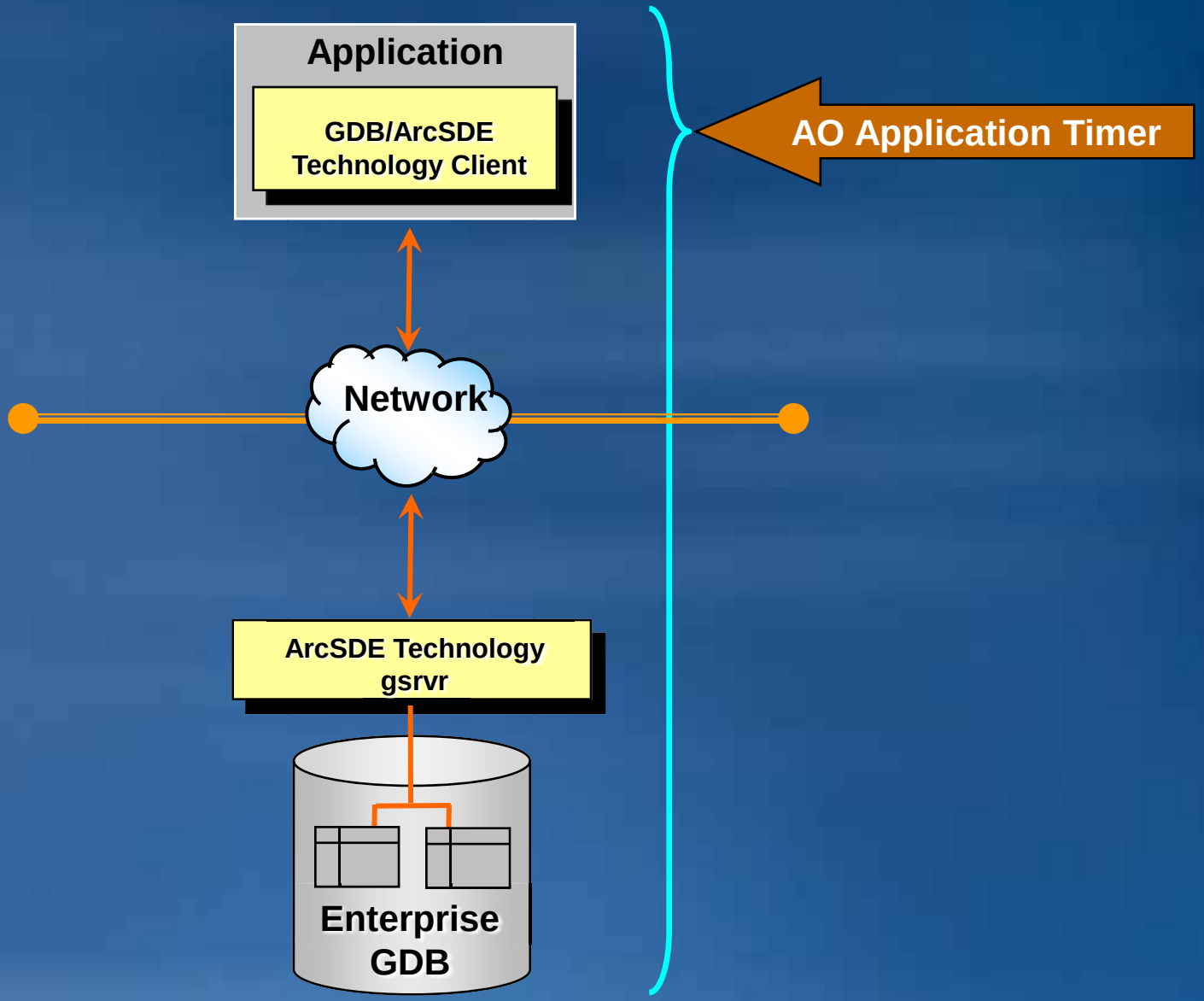


Code instrumentation and tracing

Summary

Client

Server

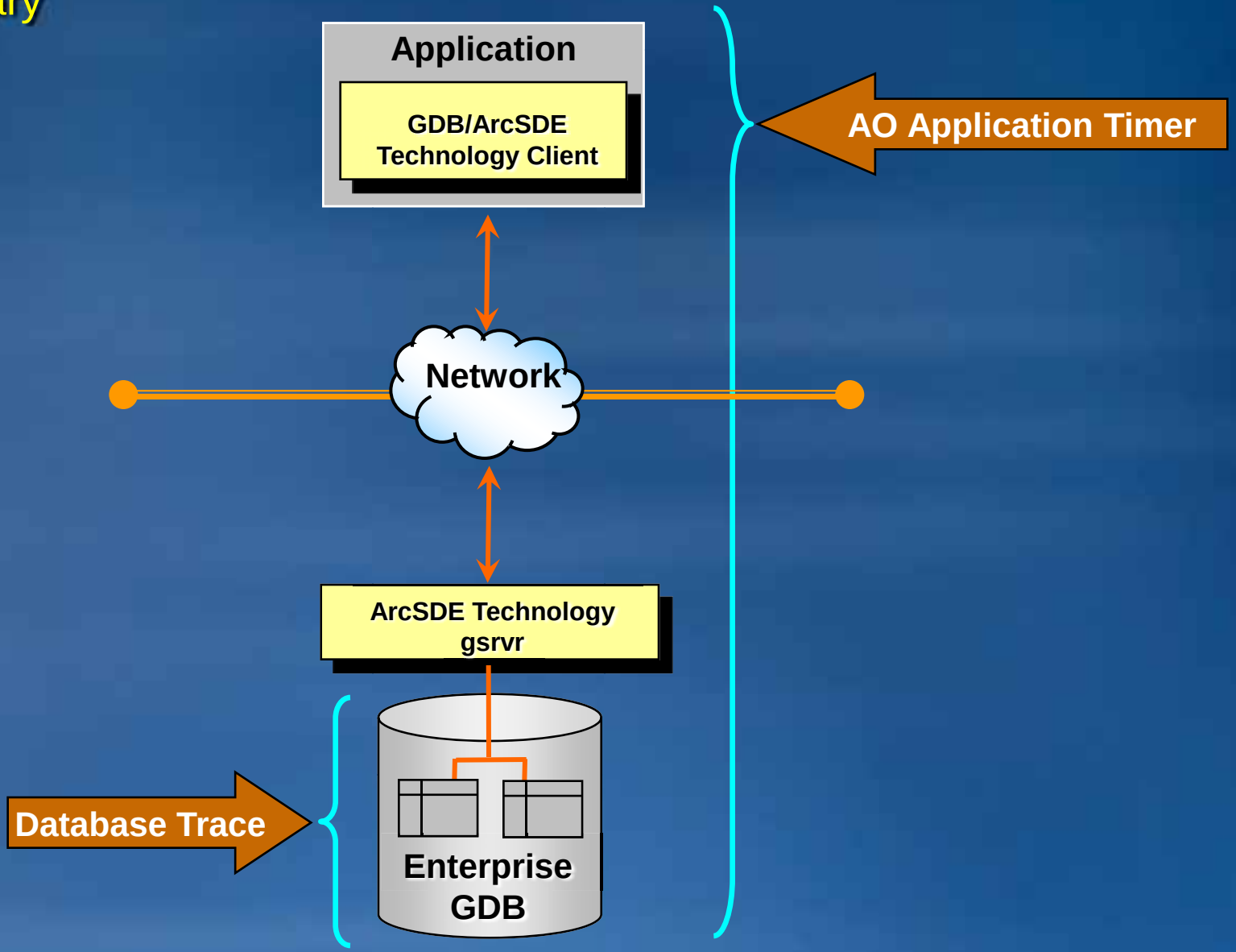


Code instrumentation and tracing

Summary

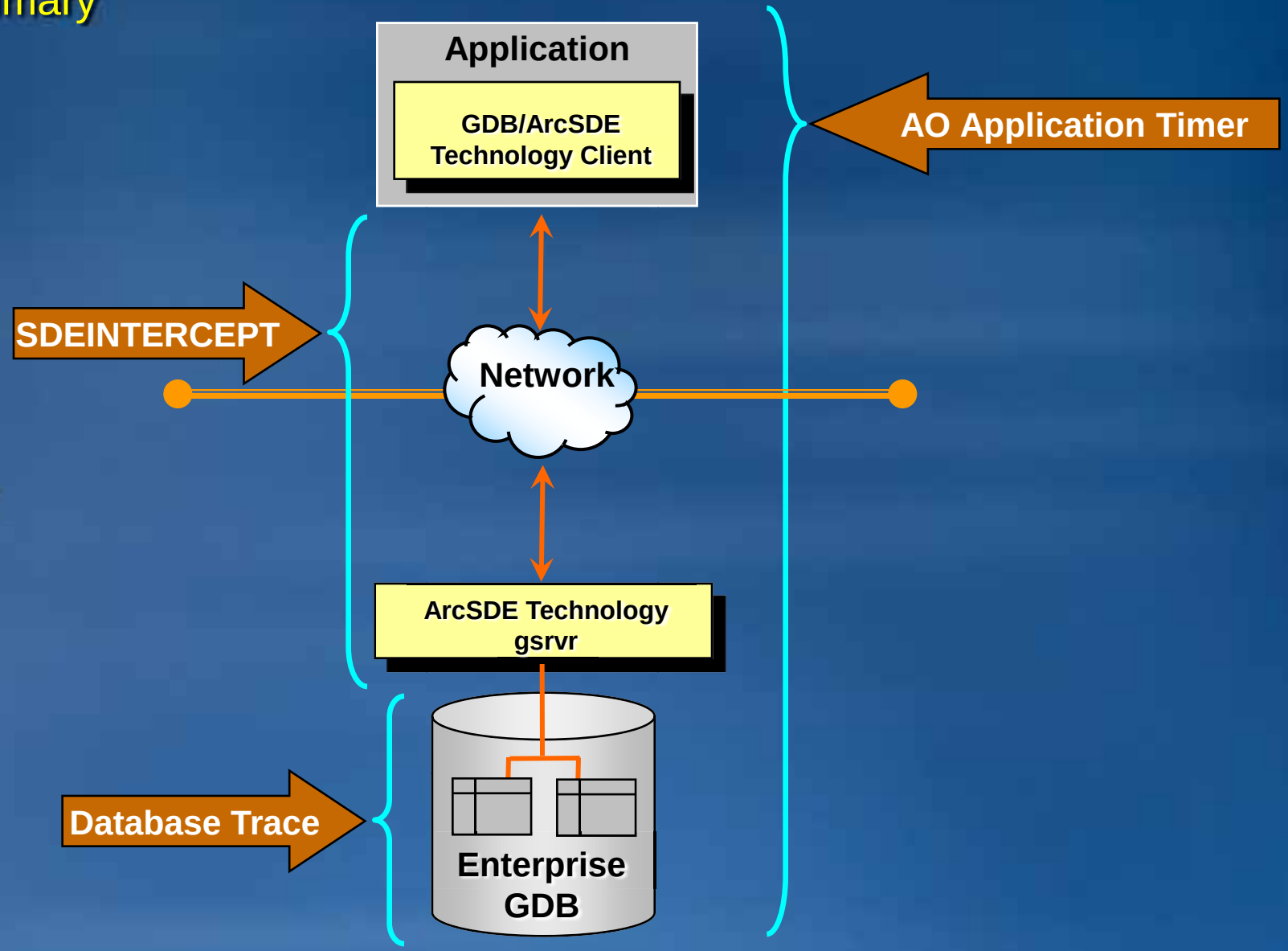
Client

Server



Code instrumentation and tracing

Summary



Code instrumentation and tracing

Summary

- Several instrumentation and tracing methods
- Look at entire application stack
- Application performance
  - Iterative process throughout development and release
  - Tools to determine where performance problems are
  - Determine where to look next
  - i.e. start at application level, or maybe the initial information is a DBMS trace, use built-in tools
- Build tracing and logging into app

Other Recommended Sessions

- Working with the Geodatabase Effectively using SQL
  - Thursday 8:30 – 9:45

In Conclusion…

- All sessions are recorded and will be available on EDN
  - Slides and code will also be available
- Please fill out session surveys!
- Still have questions?
  1. Tech talk, Demo Theatres, Meet the Team
  2. "Ask a Developer" link on web page
     - www.esri.com/devsummit/techquestions
- Instructor-Led Training
  - Managing Editing Workflows in a Multiuser Geodatabase
- Free Web Training Seminar
  - Introduction to Geodatabase Replication