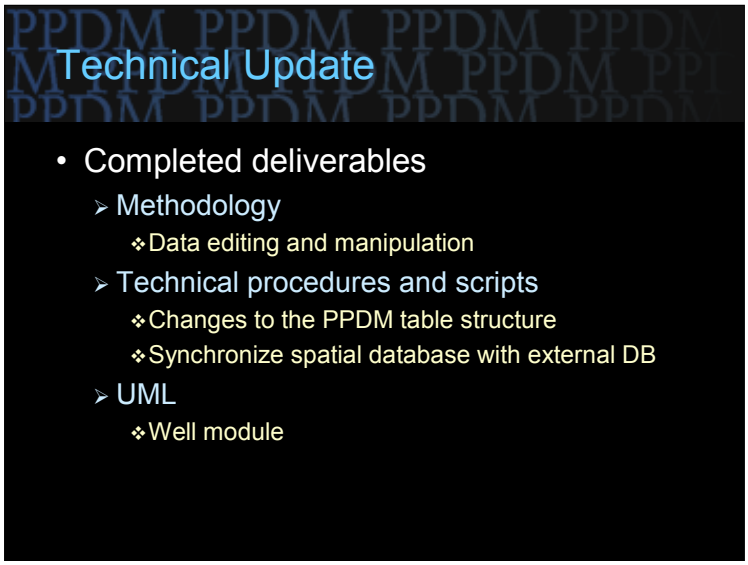
PPDM Spatial Project 2003

This presentation was given in Calgary on 6-March-2003 and Houston on 12-March-2003

Agenda

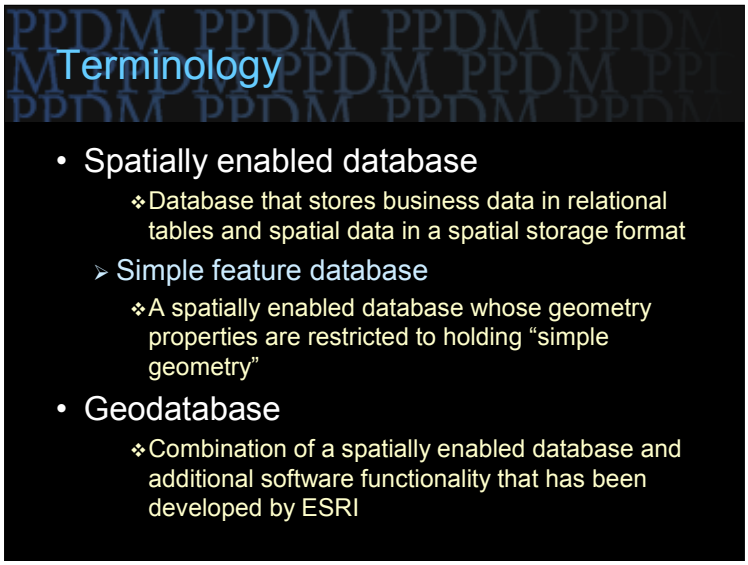
- Technical Update
  - Review completed deliverables
  - Work in progress
  - Next items
- ESRI PUG Conference
- Open Discussion



Completed Deliverables

The deliverables are available from the PPDM web site (www.ppdm.org). However, the deliverables MUST NOT be considered final. Changes can, and hopefully will, be made as a result of people downloading and trying to use the deliverables.

The methodology and technical documentation is made publicly available. The technical scripts and procedures are available to PPDM members and the UML is available to PPDM members who also are registered with the ESRI PUG.



Terminology

During the previous phase of the Spatially Enabling Project there was considerable misunderstanding of the applicability of the technology being delivered. As a result the deliverables are now being divided into clear technology tracks.

➢Spatially enabled database

A spatially enabled database stores data that a GIS can read. The term "spatially enabled database" is used when the deliverable attempts to be completely open. Many deliverables of this project are applicable to a spatially enabled database.

➢Simple feature database

A simple feature database stores the spatial data that follows OpenGIS Consortium standards for simple features.

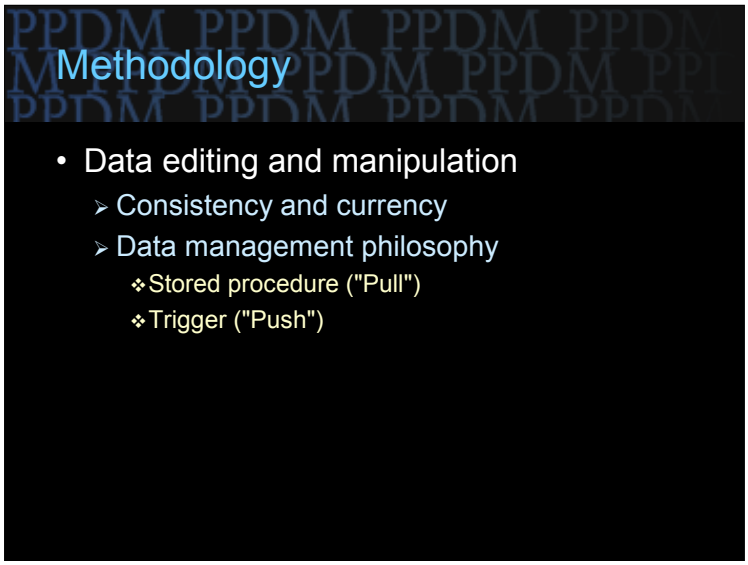
A simple feature is a geometric object constructed such that its interior is isotropic (all points have isomorphic neighbourhoods), and hence everywhere locally isomorphic to an open subset of a Euclidean coordinate space of the appropriate dimension (Can someone PLEASE explain to me what that means!). For further reading on what is a simple feature please consult http://www.opengis.org/techno/abstract/01-101.pdf

This project implements a simple feature database using Oracle Spatial as the spatial storage format. It uses Oracle PL/SQL code to create the geometries and synchronise the spatial and attribute data.

➢Geodatabase

A Geodatabase a term developed by ESRI to describe a suite of technology that provides a storage mechanism for spatial and attribute data that contains specific storage structures for features, collections of features, attributes, relationships between attributes and relationships between features. For more details on a Geodatabase please consult http://www.esri.com/library/whitepapers/pdfs/arcgis_geodb_multiuser.pdf

This project implements a Geodatabase using SDE Binary as the spatial storage format. It uses the SDE Java API to create the geometries and synchronize the spatial and attribute data.



Data editing and manipulation

In order gain wider acceptance of spatial data in needs to be incorporated into data management departments. Each data management department will have different business drivers. Rather than trying to meet all these unknown drivers we have tried to focus on meeting some obvious ones

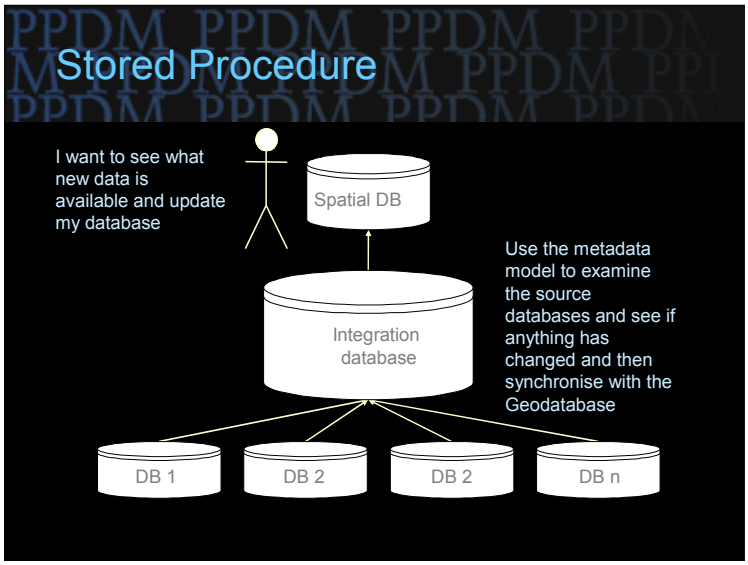
- Consistency and Currency

  Consistency requires that the value for any attribute stored in a database is the same throughout all the databases within the organisation

  Consistency and currency of data can normally be ensured through the correct application of technology. This project provides recommendations for how to ensure consistency and currency of data. However, in order to ensure quality of data, technology is a small part of the whole solution. Technology can only enable processes that require human intervention to create the quality of data. The project does not address any process issues- the scope is limited to the technology that can enable the processes

- Data Management philosophy

  This is a decision that each company will need to make: factors that can determine which philosophy to adopt include volume of data, quality of data and frequency of updates.



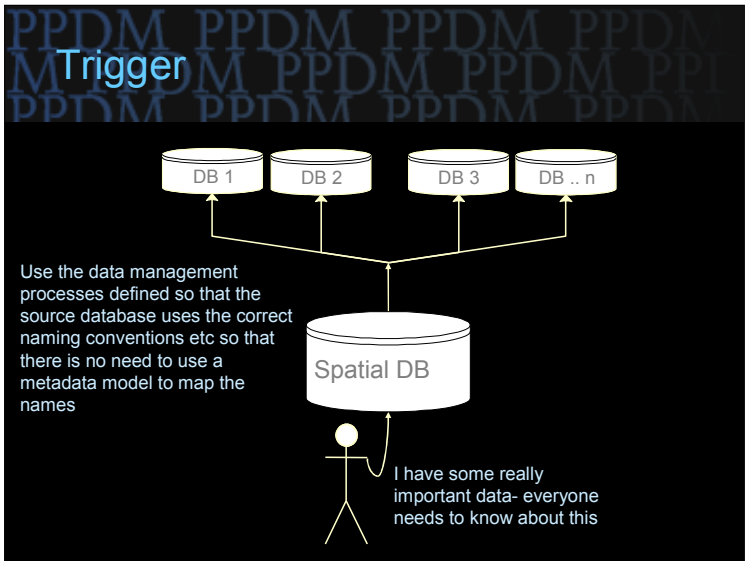
Stored Procedure

The integration database contains all the information to map different entities in different databases and ensure consistency.

Stored procedures are generally used when the end user requires control over the timing of synchronization and does not have control over the quality of the data that is available to him. It is also used where the volume of data being transferred is large or when the synchronization occurs at regular intervals and can be scheduled to occur out of office hours.

The advantage of using stored procedures is that it can run once and can synchronise many databases. It also uses code that is centralized.

The disadvantage is the currency of the data depends on the frequency of running the stored procedure.



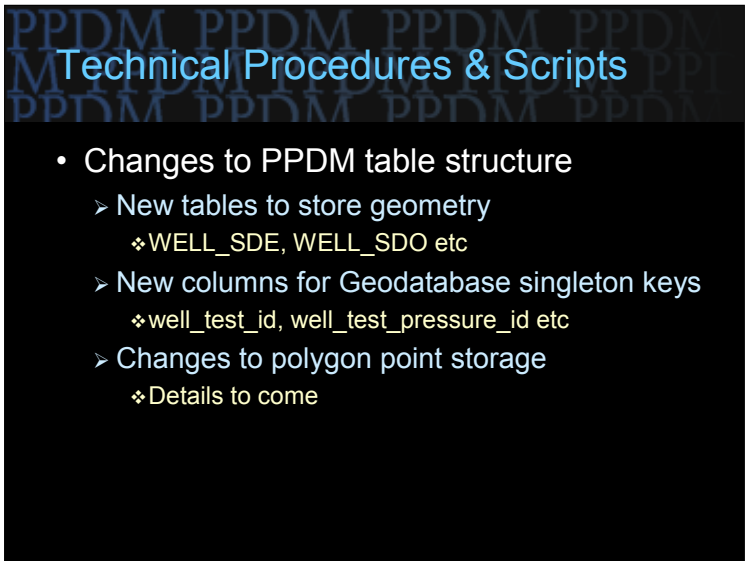
Trigger

The user uses the data management processes defined by the data management group. This means that the source database uses the correct naming conventions etc so that there is no need to use a metadata model to map the names

Triggers can be used when the data is of known (i.e good) quality, or when the volume of data is small.

The advantage of using triggers is that currency of the data is guaranteed: as soon as the change is made in the source database it is replicated to all other databases.

The disadvantage of using triggers is that the user may feel that they have lost control over their data. In addition the triggers have to be implemented on each database that is a source.



Changes to PPDM table structure

This is a work in progress and the table names, column names and particularly column types may change considerably.

- New tables to store geometry

  In order to store the spatial data there needs to be new tables. The table naming convention results in the names WELL_SDE or WELL_SDO depending on the particular spatial storage format that you use. This naming convention will be used for other datatypes as the geometry scripts are created

- New columns for Geodatabase singleton keys

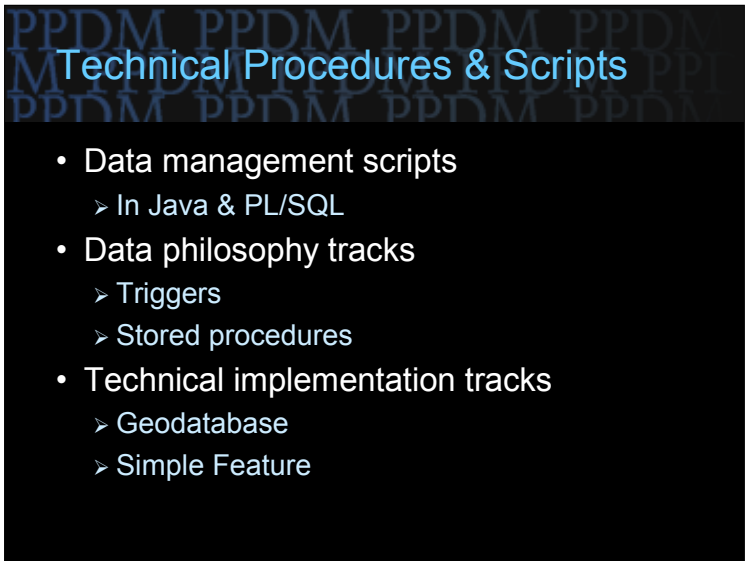
  A Geodatabase requires singleton keys to enforce relationship classes. The standard PPDM DDL uses compound keys. Therefore, in order to use relationship classes in a Geodatabase singleton keys must be created. The columns are currently defined as integer and are named according to the following convention:

  <table_name>_id

  NB Even if the table already has a column that can be used a singleton key the new column is still added.

- Changes to polygon point storage

  The relational tables that store the data that is used to create polygon geometries need to be redesigned. However, this process is still in the pre-alpha stage. The intention is to use the techniques employed by the spatial storage formats and duplicate functionality into the relational tables

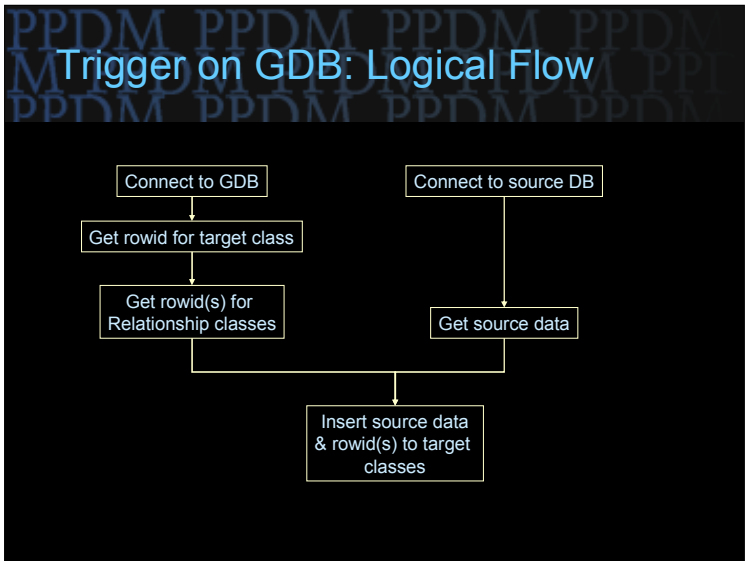


Data Management Scripts

- Code Tracks

  The code is available in four tracks

  - Trigger on simple feature database
  - Stored procedure on simple feature database
  - Trigger on Geodatabase
  - Stored procedure on Geodatabase.



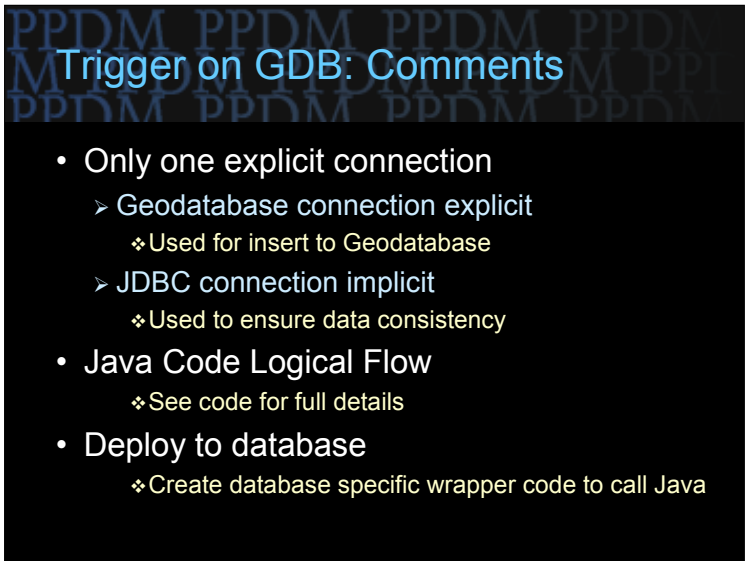
Logical Flow

In order to insert to a Geodatabase the trigger needs to know the "next" singleton key to insert. There are a number of ways that this can be undertaken, including using sequences, but the general principle remains: it is not sufficient to just insert the data, the singleton key must be inserted as well. Somehow the trigger needs to create this singleton key

In addition, if the table is a dependent on a foreign key in another table, then the singleton key for the parent row must also be inserted. NB This process needs to be repeated for all potential parent tables.

The source data is typically passed to the program as a series of parameters. If the trigger is on a table that creates a geometry then the source data may require additional queries to ensure that the "best" geometry value is created in the spatial database.

Finally, the data can be inserted to the Geodatabase. This requires that the source database user has insert/ update/ delete privilege on the Geodatabase and may require the use of database links if the Geodatabase is in a different database instance.



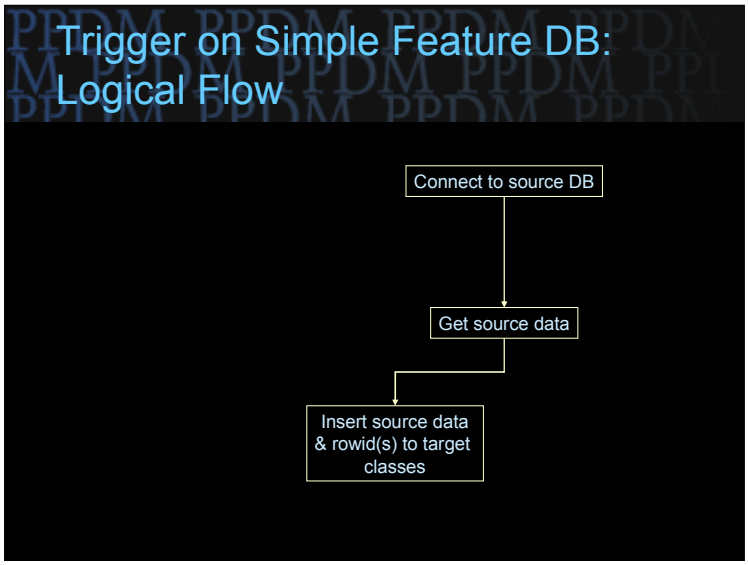
Comments

- Connections

  The trigger runs inside a database. Therefore, the source connection is automatically established. However, the connection to the Geodatabase needs to be explicitly made using an SDE connection (SeConnection).

- Deploy to database

  The trigger MUST be written in a non-native database language (i.e. Java) and so the code must be embedded in the database and then called by a wrapper in the native database language.



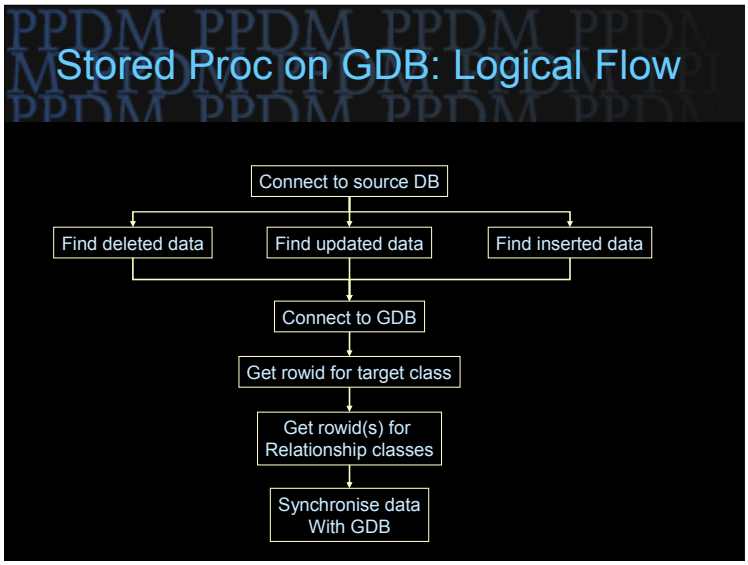
Logical Flow

This process can use the native database language or be written in another language. As for the flow to a Geodatabase, if the source data forms part of the geometry then the trigger needs to check that the complete geometry creation logic is followed before overwriting a "better" geometry.

The big difference to note is that the simple feature database does not require the singleton key logic.

Trigger on Simple Feature DB: Comments

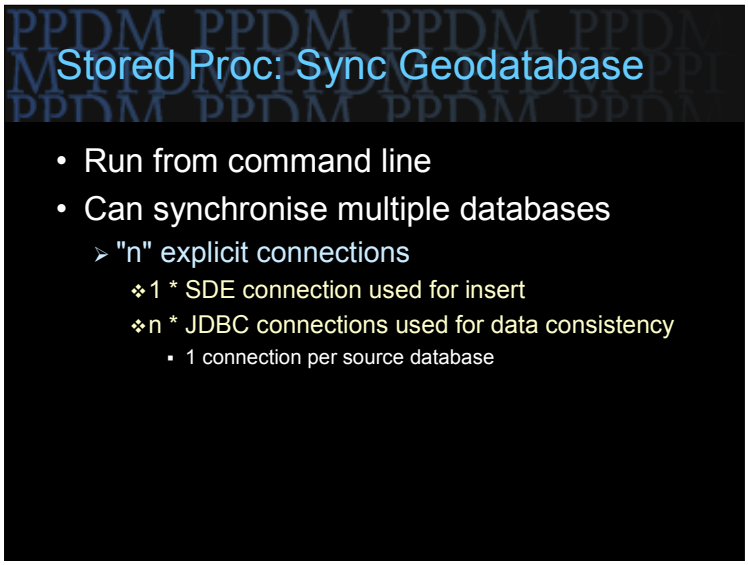
- Database specific code
  - Exactly the same as any other synchronization



Logical Flow

The stored procedure can connect to multiple sources and perform multiple actions in a single script. This means that it is not necessary to ensure that a single user has read/ write privilege across multiple schema as each connection can uses different userid/ passwords. It also has the advantage that it is non-intrusive on the source database(s).

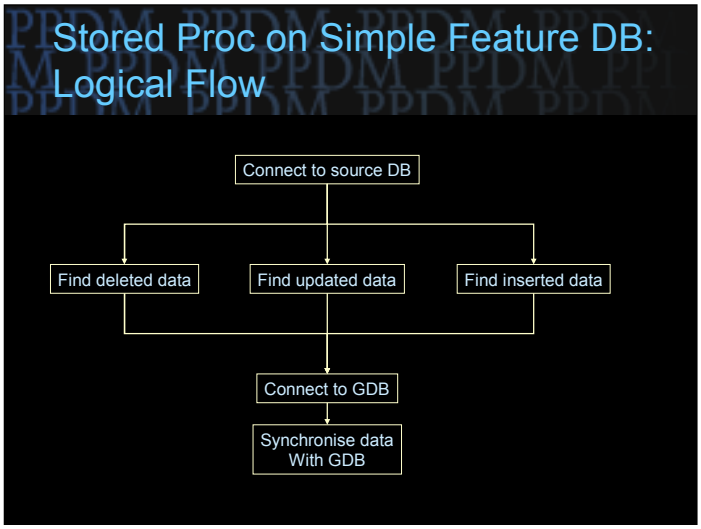
A potential problem can exist when updating data- it is possible to update a row in the Geodatabase and change the primary key. As long as both the new and old values exist in the parent table that enforces referential integrity then the update occurs without violating database integrity. However, the singleton key used to validate **Geodatabase** integrity is not. Therefore when implementing an update process the code must check to see if the primary key has been updated and if so, update the singleton key as well.



Comments

- Run from command line

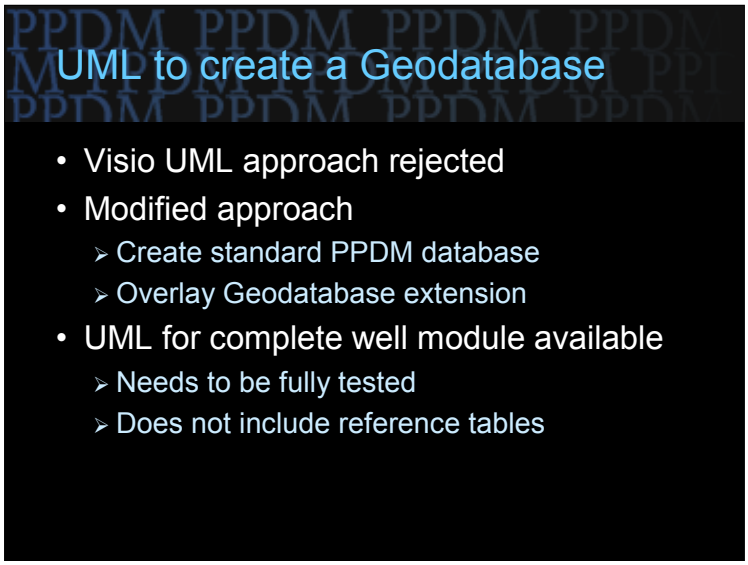
  The synchronization process can be as simple as a simple script that is run from the command line. This allows it to be incorporated into a cron job or other scheduled event.



Logical Flow

The stored procedure for a simple feature DB can also connect to multiple sources and perform multiple actions.

Because there are no singleton keys the additional code to ensure intra-row consistency is not required



UML

- Original approach

  The previous phase of the Spatial Project created a detailed UML in Microsoft Visio 2000 for the well header and well test module of PPDM. The only way to create that UML is to type in each column name, column type and any additional column constraints. This phase of the project is attempting to spatially enable the complete data model. This has ~17,000 columns in 900 tables and >5,000 constraints.

  The deliverable created by typing in the data model would be to view UML diagrams for the data model, and export the information to an XMI file for import to ArcCatalog in order to create a Geodatabase.

  However, the UML diagrams would duplicate the content of the data diagrams currently available from PPDM, but instead of using entity relational symbols for the objects it would use UML symbols.

  Given the amount of effort required to create the Visio UML and the debatable value obtained from it the "type it all in" approach was rejected.

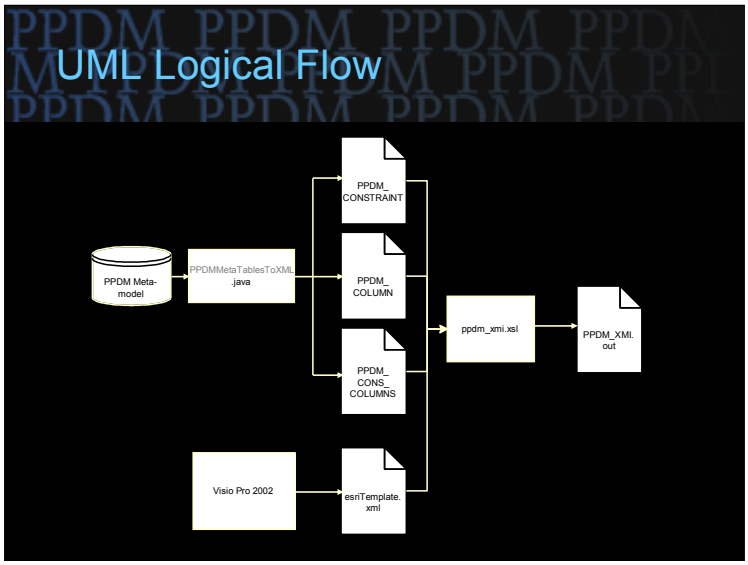
- Modified approach

  The modified approach uses the existing PPDM DDL to create a standard PPDM database. The user then uses ArcCatalog to read the XMI provided by PPDM to create a Geodatabase.

  The XMI file contains the singleton keys and relationship classes used by the Geodatabase but does not include any of the columns defined in the standard PPDM DDL. This approach should meet all the requirements of creating a Geodatabase.

- UML available now

  The XMI for the complete well module is available. However, it does not include the reference tables (GDB_CODEDDOMAINS) as the methodology for synchronizing between the source database and GDB reference tables has not been completed



UML Logical Flow

This diagram is included to show how the XMI file is created. This workflow should only be used if you are running a version of PPDM that has not had the XMI created.

- PPDM Meta-model

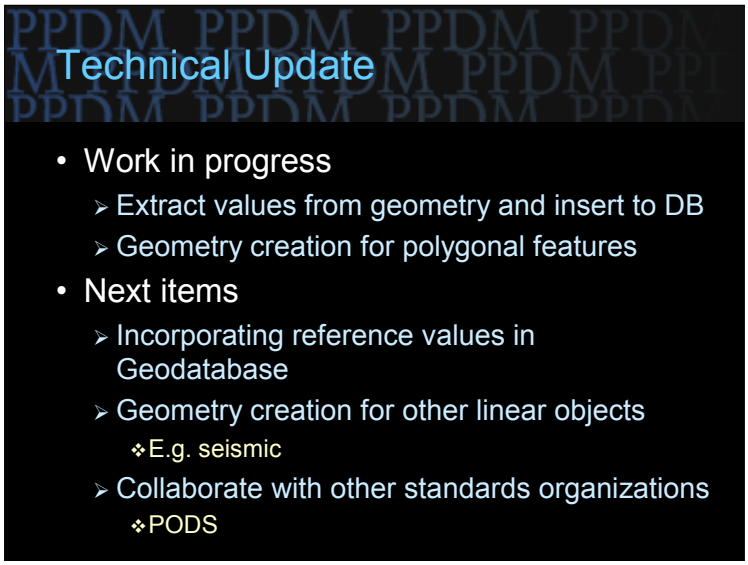
  The meta-model is included as part of version 3.6. It contains all the table names, column names etc for the entire data model. A Java script (PPDMMetaTablesToXML.java) reads these files and creates 3 XML files: ppdm_constraint.xml, ppdm_column.xml, ppdm_cons_columns.xml.

- ESRI XMI

  In order to create the Geodatabase XMI these files need to be merged with a standard ESRI XMI file created by exporting the ESRI UML template from Visio 2002 to XMI.

- Merging it all together

  ppdm_xmi.xsl then merges all 4 files together to create the final XMI file that can be read by ArcCatalog



Technical Update

- Work in progress

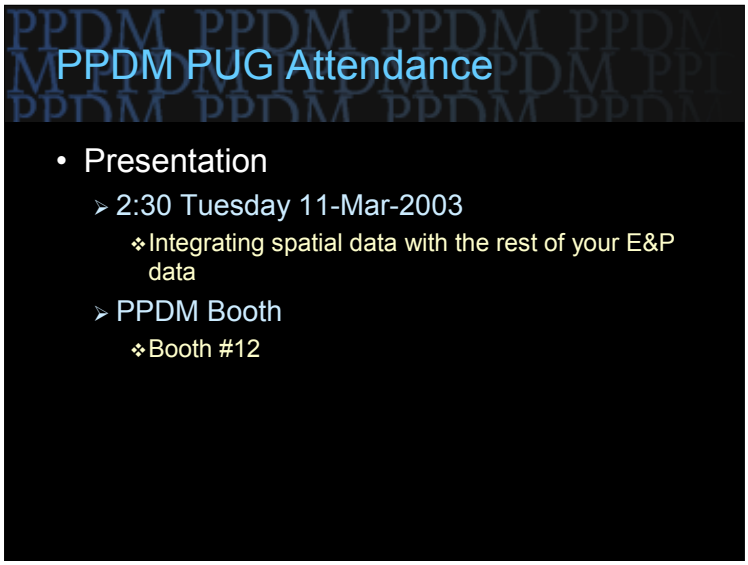
  PPDM has received a dataset from the Yukon Territorial Government. This dataset is in shapefiles. These files have been loaded into SDE and from there the geometry data is being extracted to populate the relational tables in PPDM.

  Once the relational tables are populated, the values can be read back and a new geometry created.

  This process provides a nice import/ export facility to SDE and ensures that data can complete the loop without losing any accuracy.

- Next Items

  Synchronizing the reference values held in the the R_% tables with the GDB_CODEDDOMAINS is a major gap in being able to incorporate a Geodatabase into the rest of a data management data flow. This needs to be addressed for complete acceptance of a Geodatabase



PUG Attendance

Thank you to everyone who visited the PPDM booth.

Please see the presentation available on this web site for full details of what was said in Houston.

Contact Information

- PPDM
  - Web
    - www.ppdm.org/products/spatial/index.html
    - www.ppdm.org/development/projects/spatial.htm
  - E-mail
    - ian@ppdm.org
    - spatial-l@ppdm.org

Open Discussion

- Any questions?