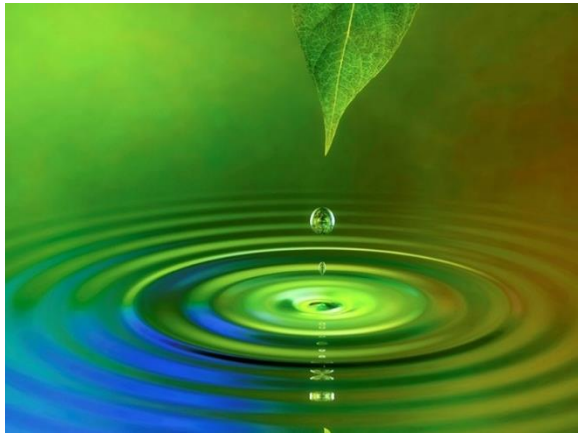



Implementing ArcGIS for Pipeline Referencing (APR) using the Pipeline Open Data Standard (PODS)

European PUG - November, 2017, London, UK

Introduction and Outline

- Peter Veenstra
  - GIS Technologist – TRC
  - PODS Board of Directors Member
  - Chairperson of PODS Next Gen Committee
- Outline
  - PODS Association & NG Initiative Update
  - Initial Findings on Implementing PODS Next Gen using APR
  - Thoughts on Data Management

PODS Association

- Best Practice
- Proven, Open, Neutral
- Advocacy Services for Member Organizations
- Deep Industry Knowledge Base

***The combined experience of* our Working Group volunteers represents ~120 Years Combined Experience**

2016/2017 – Status Update

- Key Deliverables
  - PODS Lite - 360 downloads and 1,043 views since April!
  - Next Gen
  - Operators Forum – A packed room & lively information exchange
  - Webinars – 8 webinars since Jan. 2016
  - PODS 6.1 – Available for member review October 2017

- Improved Communication and Visibility
  - Increased outreach to Members
  - Member Portal
  - More Events & Presentations
  - Streamlined Organizational Policies, Procedures and Tools

Status Report

January 2017

PODS Lite Geodatabase (APR)

August 2017

Enterprise Architect & ShapeChange Training

October 2017

PODS Lite for RDBMS

Test Plans for RDBMS and Geodatabase

Q3/4 2017

PODS Core Member Review

Q4 2017

Working Groups

- History
- ILI\Offline Inspection Storage

Q4 2017

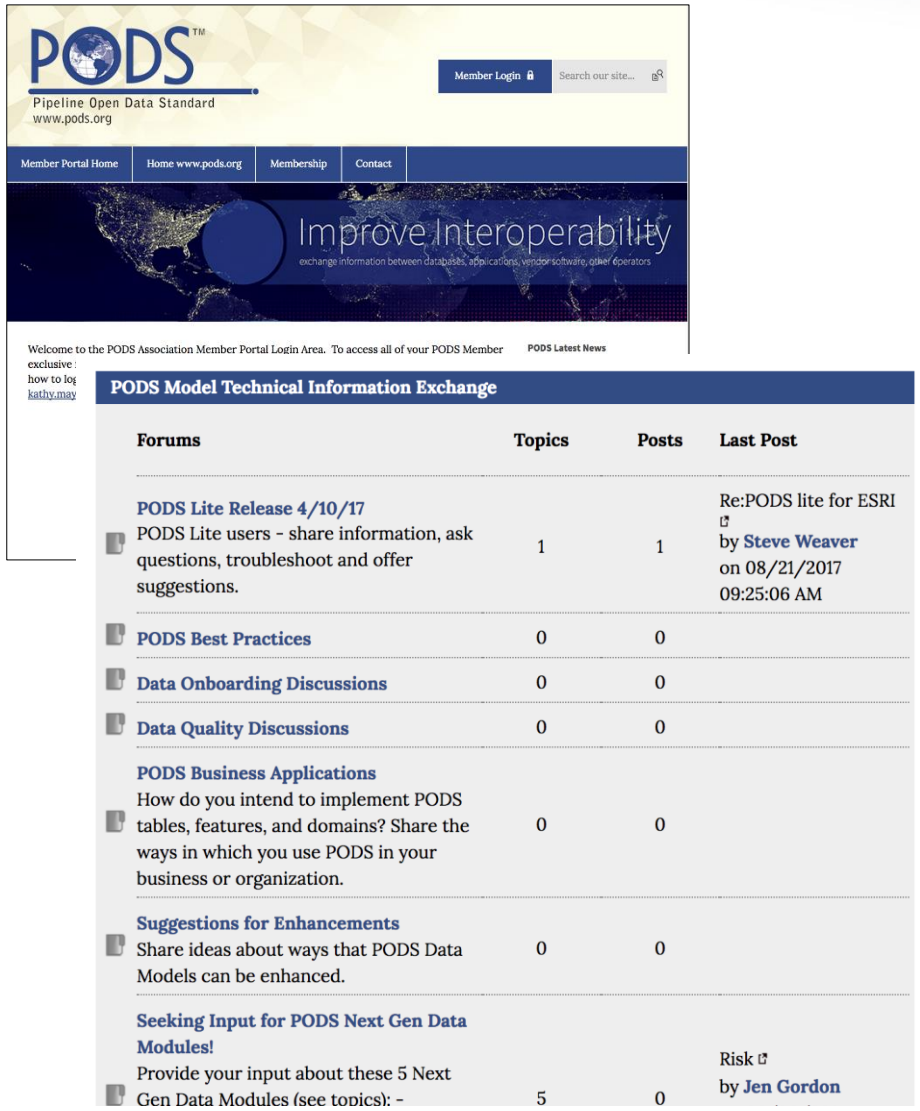
Updated Documentation

- Physical Model
- Logical Model
- Data Dictionary
- Design Principals
- Model Overview

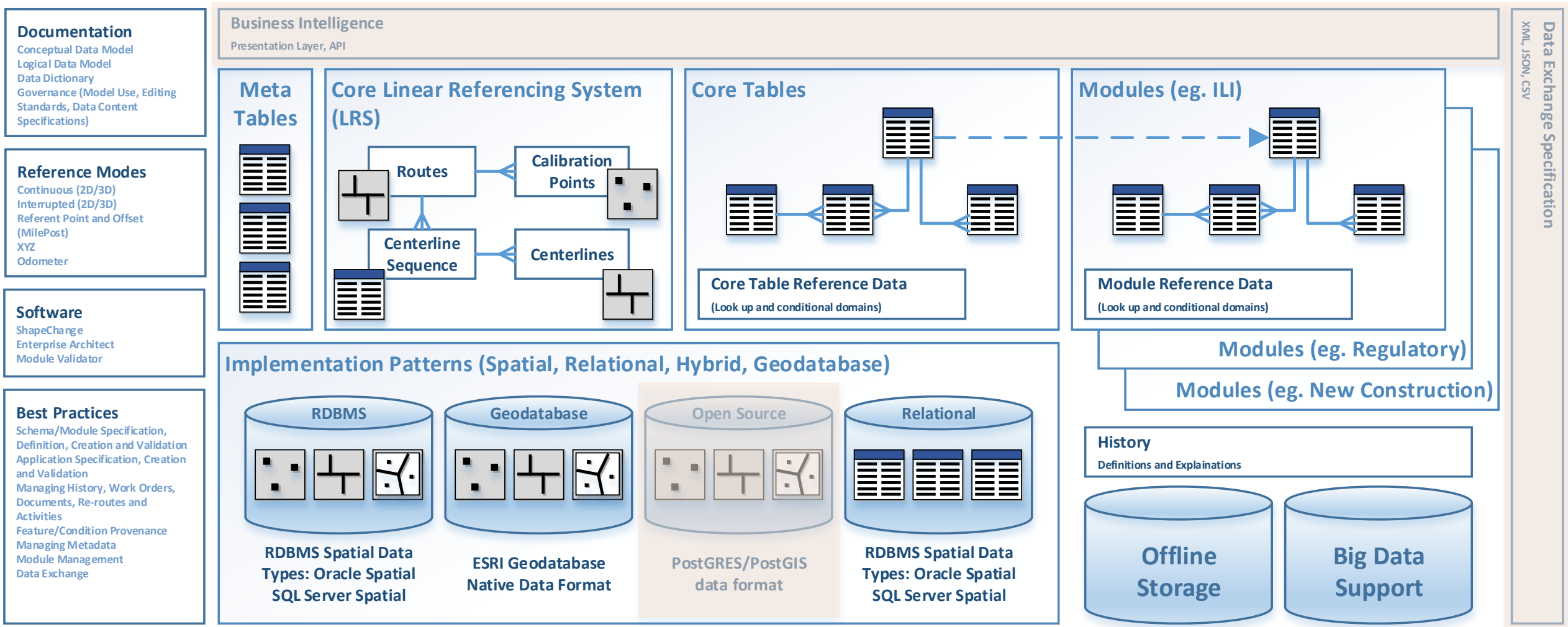
New PODS Member Portal

- Register for Events
- Download PODS Data Models
- Interact with other PODS Members
- Engage with PODS Volunteers

- "PODS Models Technical Information Exchange" in Community Forums – discussion/input
  - Next Gen Modules: Integrity/ILI; US Regulatory; Cathodic Protection/Inspection (CIS); Physical Inspections; Risk



What is PODS Next Generation (NG)?

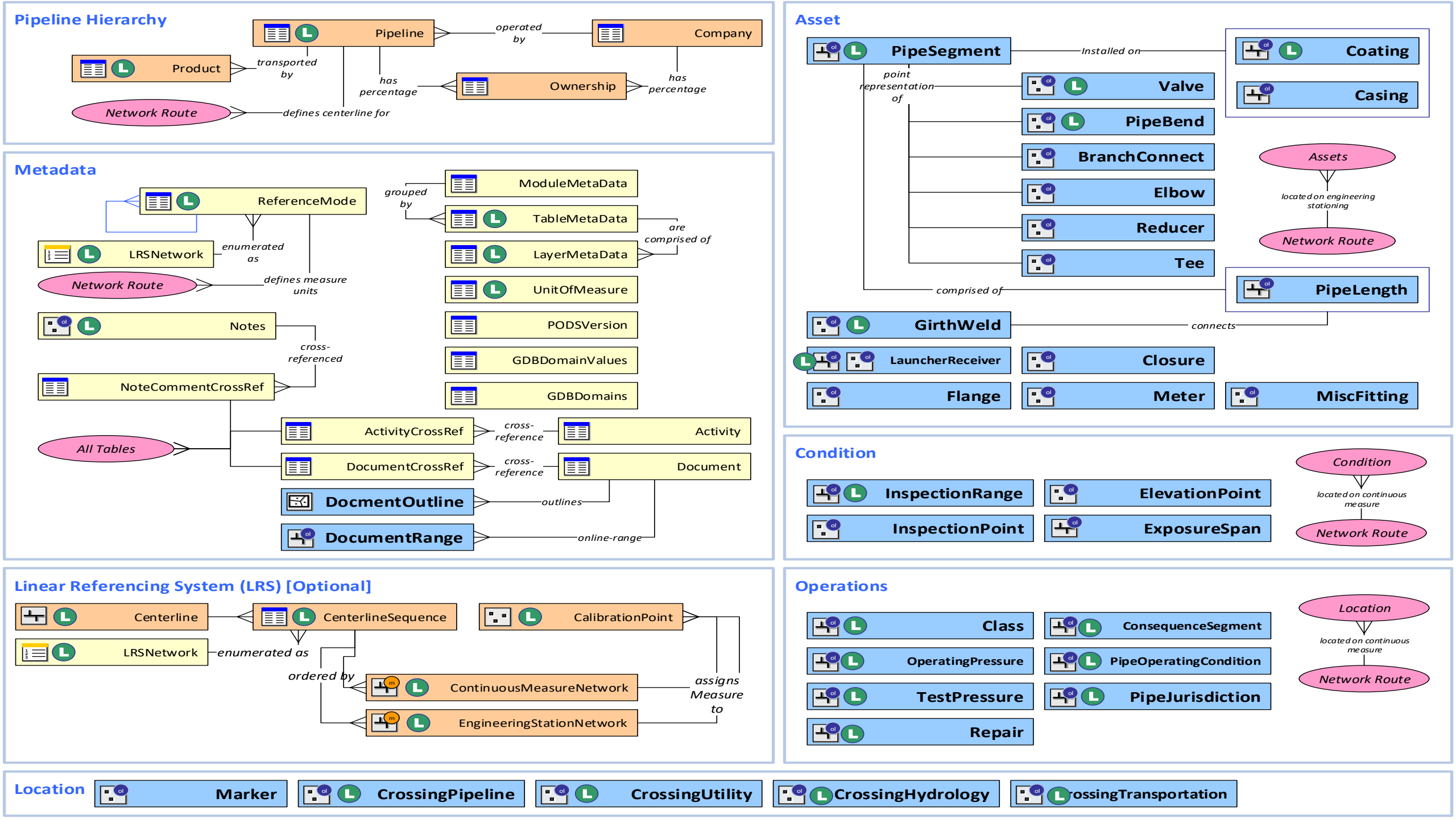


New to Next Generation of PODS Standards

PODS Lite

- Free for use by all from www.pods.org
  - 360 downloads and 1,043 views since April
  - Subset of the full data model (Next Gen Core)
  - Designed to provide look-and-feel for testing and evaluation purposes
  - Sufficient content to allow North American operators to test loading and generating annual regulatory NPMS reports
- Preview of PODS 7.0 Core (Next Gen)
- Preview of Location Model design
  - Moving location information to event tables
  - Supporting both Linear Referencing and Coordinate-based location and management (or neither, if desired)

Built for Pipelines.


Pipeline Hierarchy
Pipeline
operated by
Company
Product
transported by
has percentage
Ownership
has percentage
Network Route
defines centerline for
Metadata
ReferenceMode
grouped by
ModuleMetaData
TableMetaData
are comprised of
LRSNetwork
enumerated as
LayerMetaData
Network Route
defines measure units
UnitOfMeasure
Notes
PODSVersion
cross-referenced
GDBDomainValues
NoteCommentCrossRef
GDBDomains
All Tables
ActivityCrossRef
cross-reference
Activity
DocumentCrossRef
cross-reference
Document
DocmentOutline
outlines
DocumentRange
online-range
Asset
PipeSegment
Installed on
Coating
point representation of
Valve
Casing
PipeBend
BranchConnect
Assets
Elbow
located on engineering stationing
Reducer
Network Route
Tee
comprised of
PipeLength
GirthWeld
connects
LauncherReceiver
Closure
Flange
Meter
MiscFitting
Condition
InspectionRange
ElevationPoint
Condition
InspectionPoint
ExposureSpan
located on continuous measure
Network Route
Linear Referencing System (LRS) [Optional]
Centerline
CenterlineSequence
CalibrationPoint
LRSNetwork
enumerated as
ordered by
ContinuousMeasureNetwork
assigns Measure to
EngineeringStationNetwork
Operations
Class
ConsequenceSegment
Location
OperatingPressure
PipeOperatingCondition
located on continuous measure
TestPressure
PipeJurisdiction
Network Route
Repair
Location
Marker
CrossingPipeline
CrossingUtility
CrossingHydrology
rossingTransportation

NG Modeling Approach: Two Tools

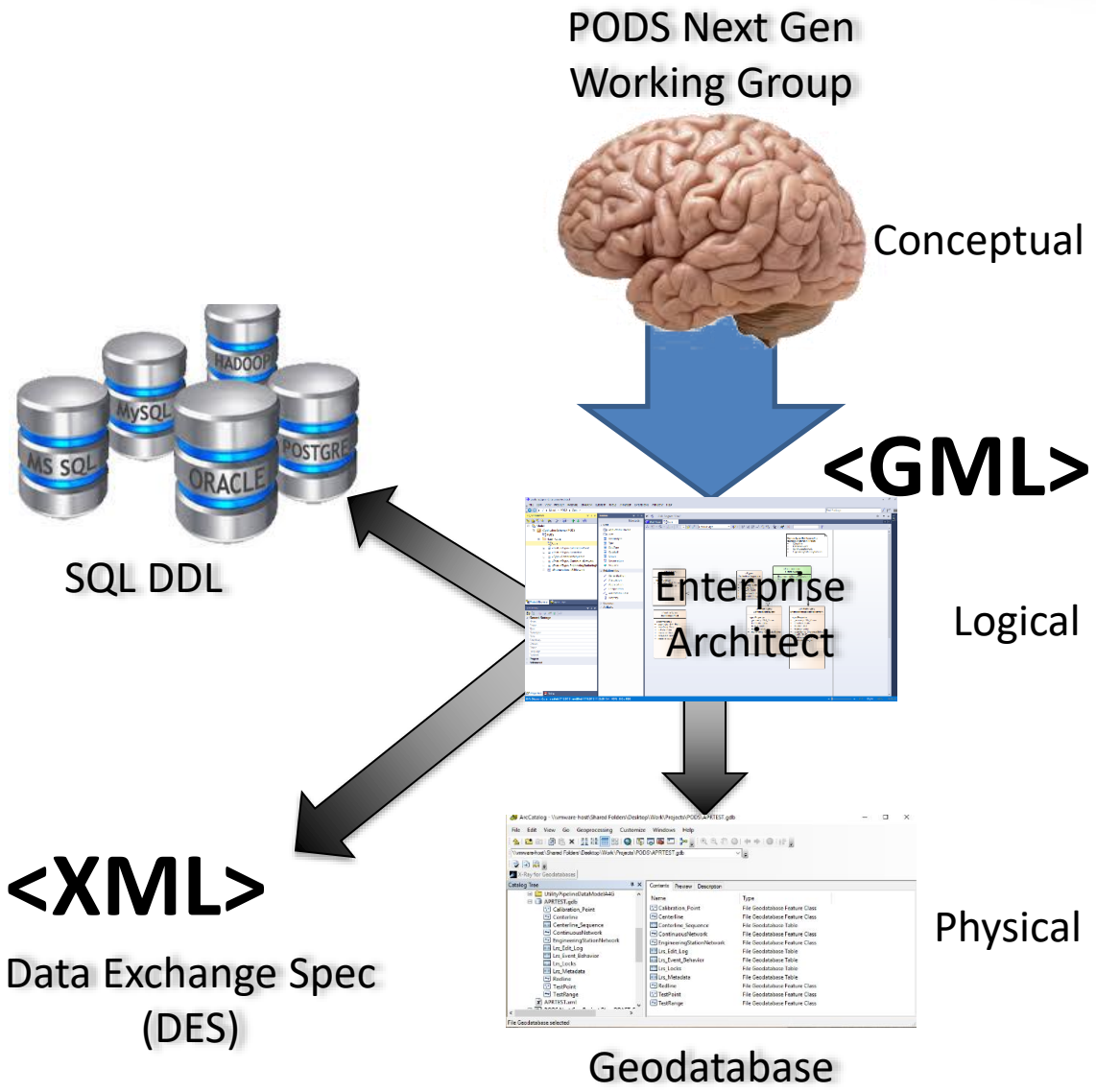
1. Enterprise Architect
   (with GML and ArcGIS Extensions)

   Creates a GML Logical model (one model to rule them all)

2. ShapeChange

   Generates Physical Models

   - SQL DDL: Oracle, SQL Server, PostgreSQL with and without spatial types
   - Geodatabase: ArcGIS EA model and then from there to Workspace XML
   - DES: GML 3.2/3.3 + Schematron

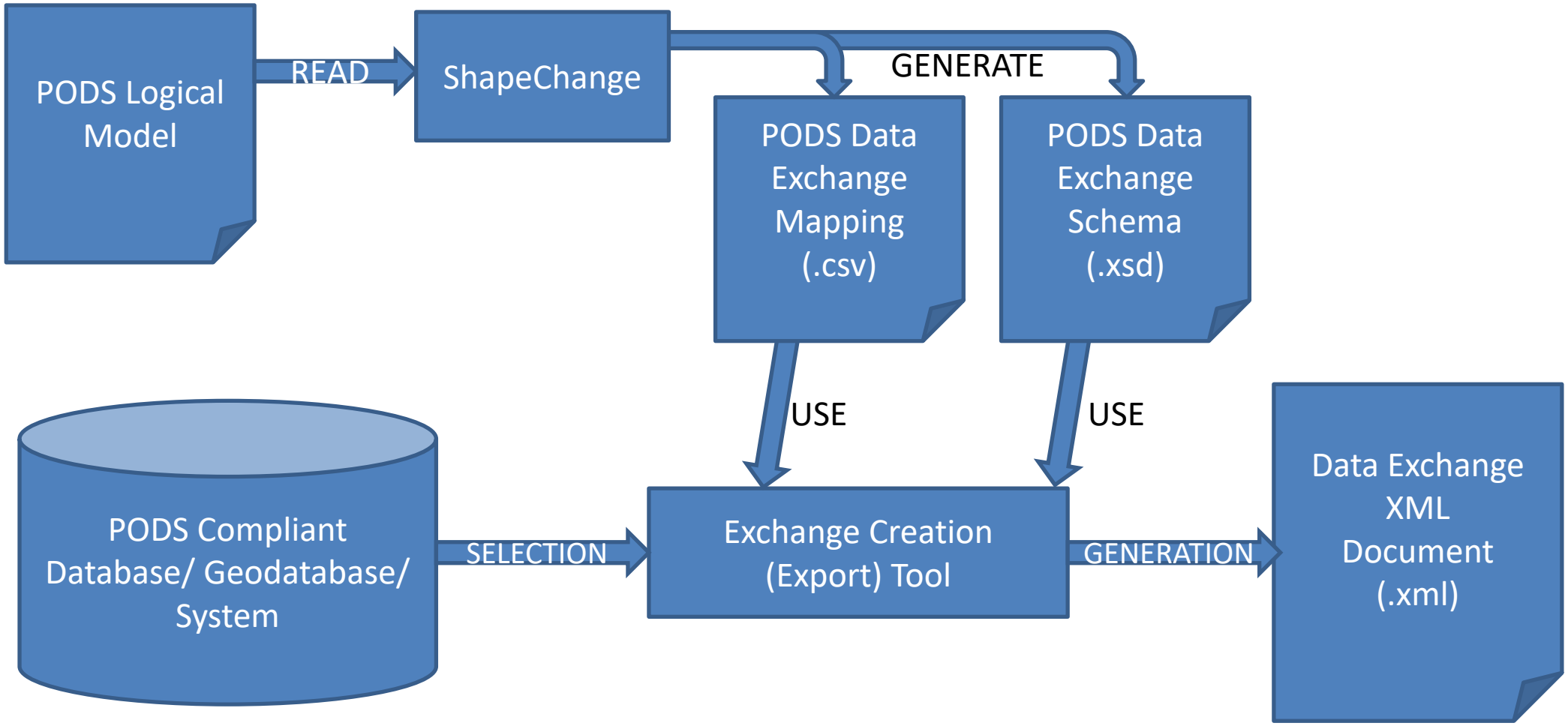


Data Exchange Specification (DES)

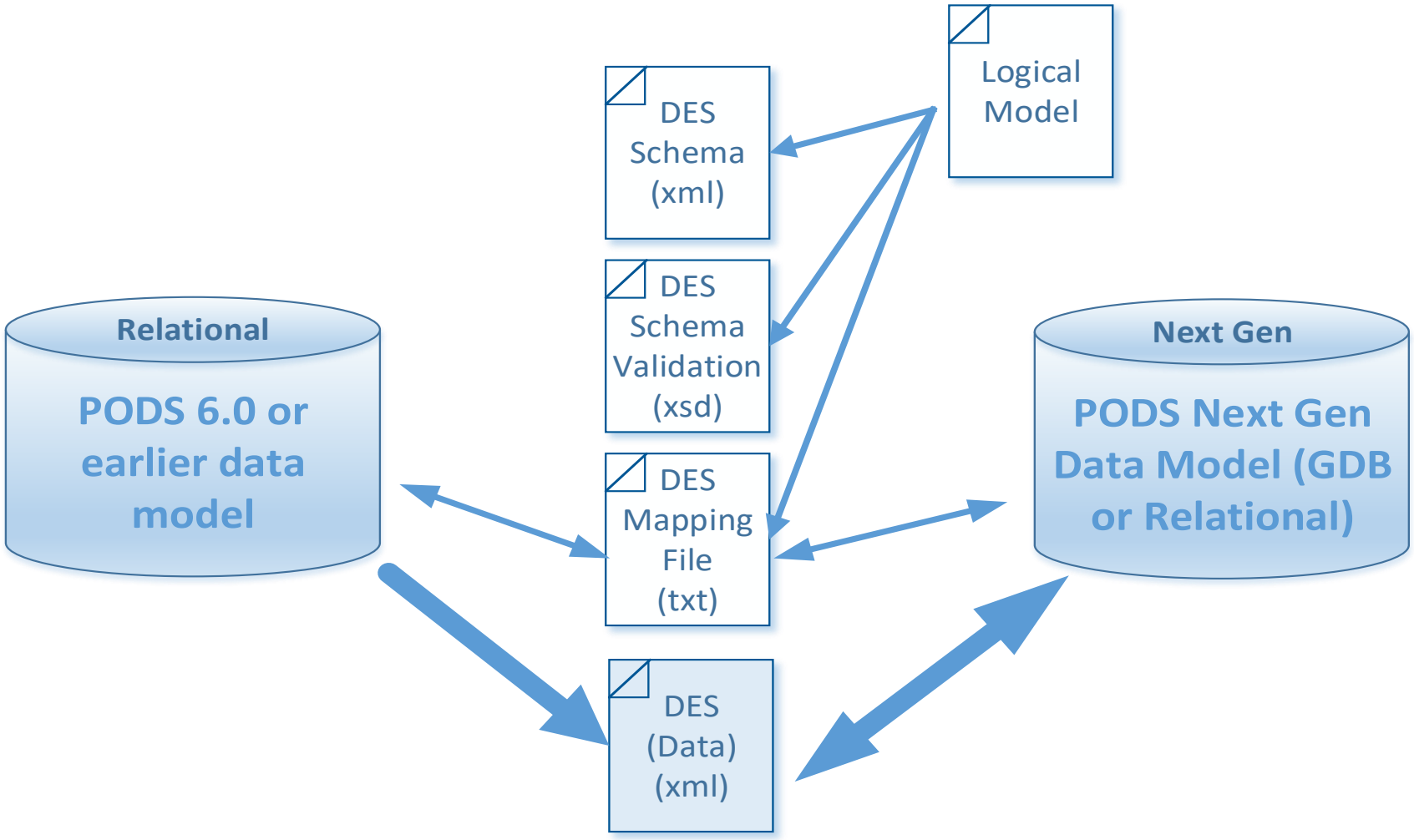
- Three XML Files: Schema Definition/Rules; Data; Schema Mapping File
- Used as a transfer file format between databases and software systems
- Will be a standard schema, with consistent attribute names and datatypes
- ‘Transfers’ data between systems – becomes a ‘PODS Standard Data Format’
- Why XML?
  - Industry Standard File Format
  - Used for data exchanged and is ‘extensible’
  - Has built in schema and content validation protocols
  - Machine and human-readable
  - Structured, consistent and defensible

<xml />

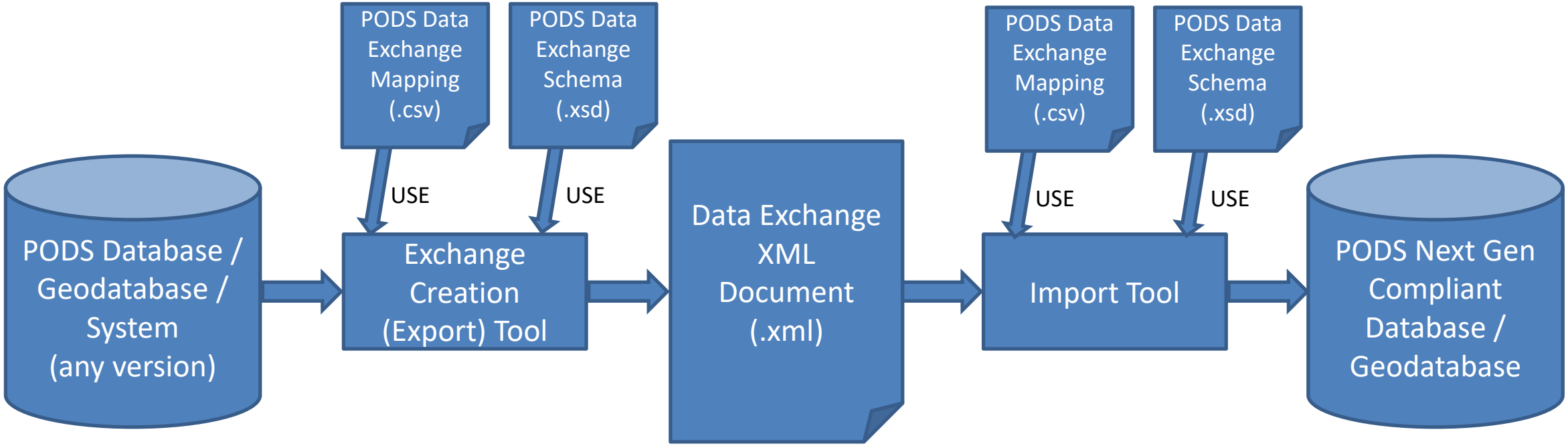
DES Generation Process



DES: Mapping Process Source to Target



DES Transfer Process



Future - Modules

- ‘Added’ on to PODS Core
- Contain tables, attributes, relationships and domains describing a particular topic.
- Provide operators the option of what they want to include in their module beyond the core
- Depend solely on the core
- Can be developed by operators but there will be official PODS Modules available from the PODS website
- Priority ranked by the TCDM, NG, and TCG teams

MODULES
Integrity/ILI
US Regulatory
Cathodic Protection/Inspection (CIS)
Physical Inspections
Risk
New Construction
Offshore
Gathering
Leak
Offline/Site
Storage
Operating Measures
SCC
Compressor

FAQ: PODS Lite

What is the difference between PODS Lite and PODS Core?

PODS Lite	PODS CORE
First glimpse at Next Generation PODS - released for testing and R&D and to provide a flavor of the Core	PODS Lite + additional tables = Next Generation PODS Core
All the tables in PODS Lite are part of PODS CORE	An expanded set of tables, attributes, domains and relationships that builds on the PODS Lite offering
The Basis of PODS CORE	Contains ALL of PODS LITE plus more
FREE!	Requires current PODS Membership to access and use

FAQ: Terms

The one thing that confuses me is the terms – PODS Next Generation, PODS Lite, PODS CORE, PODS 7.0 Model – Help!

Next Generation = the initiative to transform the data model
PODS Lite = the first release, as a POC and to support APR (strategic)
PODS CORE = the basis of all future PODS model work and is the initial release of the PODS 7.0 Model
PODS 7.0 Model = the CORE tables, the Data Exchange Specification and any additional tables that are developed in future, to be determined and yet to be released MODULES

The reason for these different terms is because data models and supporting documentation aren't created instantly – we needed to move forward and show progress

FAQ: Do I need ESRI APR?

Do I need ESRI APR to use PODS Lite or PODS 7.0

- No
- PODS Lite and the core of PODS 7.0 utilize the same table structure for managing centerlines and linear referenced networks as ESRI APR but they do not require ESRI APR to function
- PODS Service Providers and operators can develop their own processes, routines and software for managing data in PODS 7.0 format if they choose.

FAQ: How does this align with ESRI APR?

- Implements ESRI ArcGIS for Pipeline Referencing (APR) core for management of linear referenced network route systems
- This does not mean that PODS Next Gen will only work with APR
  It does mean that APR can be easily implemented with a PODS Next Gen data model
- APR can work with any data model – not just Esri's UPDM – as long as the data model meets certain minimal requirements
- PODS Next Gen data model will meet those requirements
- PODS Association Members can therefore use APR to manage information in a ESRI Geodatabase

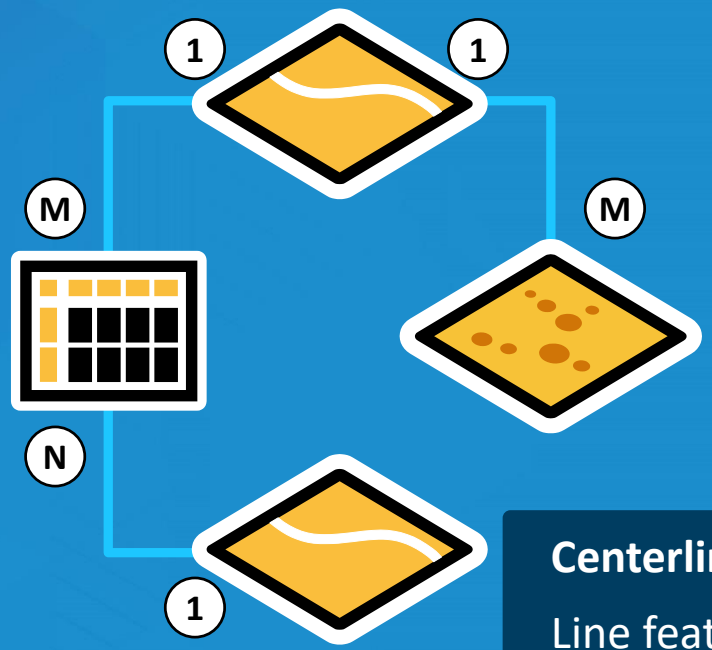
Location Model

Schema for route centerline management

Routes *(Network)*

Route features

Separate feature class for each LRM



Centerline Sequence

Key table for M-N relationship between Centerline and Route

Calibration Points

Point feature class that stores route measures

Centerline

Line feature class that stores route geometry

…with support for Engineering Stationing

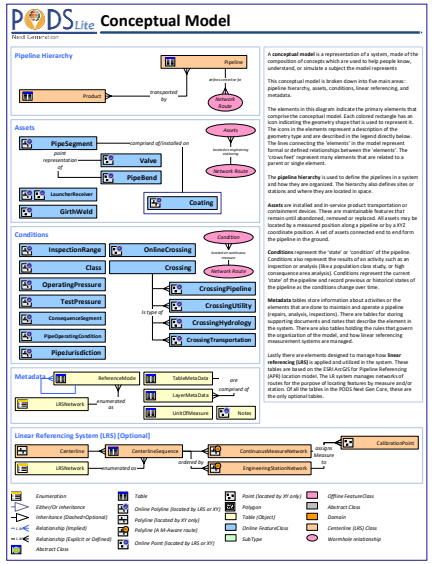
What is ArcGIS for Pipeline Referencing (APR)

- ESRI Software Product
- Schema and Data Structure within Geodatabase
  - Networks and events are stored as ‘features’
- Tools for managing networks using coordinate position or linear referenced position
  - Software manages LRS location
- A location service bus for locating things on or along a network
- Resources for further learning (ESRI/PODS Websites)

PODS Core Conceptual Model

PODS Next Generation

TRC Results you can rely on



A **conceptual model** is a representation of a system, made of the composition of concepts which are used to help people know, understand, or simulate a subject the model represents

This conceptual model is broken down into five main areas: pipeline hierarchy, assets, conditions, linear referencing, and metadata.

The elements in this diagram indicate the primary elements that comprise the conceptual model. Each colored rectangle has an icon indicating the geometry shape that is used to represent it. The icons in the elements represent a description of the geometry type and are described in the legend directly below. The lines connecting the 'elements' in the model represent formal or defined relationships between the 'elements'. The 'crows feet' represent many elements that are related to a parent or single element.

The **pipeline hierarchy** is used to define the pipelines in a system and how they are organized. The hierarchy also defines sites or stations and where they are located in space.

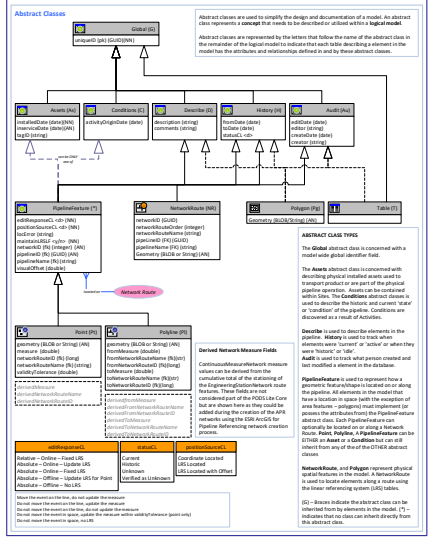
Assets are installed and in-service product transportation or containment devices. These are maintainable features that remain until abandoned, removed or replaced. All assets may be located by a measured position along a pipeline or by a XYZ coordinate position. A set of assets connected end to end form the pipeline in the ground.

Conditions represent the 'state' or 'condition' of the pipeline. Conditions also represent the results of an activity such as an inspection or analysis (like a population class study, or high consequence area analysis). Conditions represent the current 'state' of the pipeline and record previous or historical states of the pipeline as the conditions change over time.

Metadata tables store information about activities or the elements that are done to maintain and operate a pipeline (repairs, analysis, inspections). There are tables for storing supporting documents and notes that describe the element in the system. There are also tables holding the rules that govern the organization of the model, and how linear referencing measurement systems are managed.

Lastly there are elements designed to manage how **linear referencing (LRS)** is applied and utilized in the system. These tables are based on the ESRI ArcGIS for Pipeline Referencing (APR) location model. The LR system manages networks of routes for the purpose of locating features by measure and/or station. Of all the tables in the PODS Next Gen Core, these are the only optional tables.

Logical Model Design – Abstract Classes (I)



Abstract classes are used to simplify the design and documentation of a model. An abstract class represents a **concept** that needs to be described or utilized within a **logical model**.

Abstract classes are represented by the letters that follow the name of the abstract class in the remainder of the logical model to indicate that each table describing a element in the model has the attributes and relationships defined in and by these abstract classes.

ABSTRACT CLASS TYPES

The **Global** abstract class is concerned with a model-wide global identifier field.

The **Assets** abstract class is concerned with describing physical installed assets used to transport product or are part of the physical pipeline operation. Assets can be contained within Sites. The **Conditions** abstract classes is used to describe the historic and current 'state' or 'condition' of the pipeline. Conditions are discovered as a result of Activities.

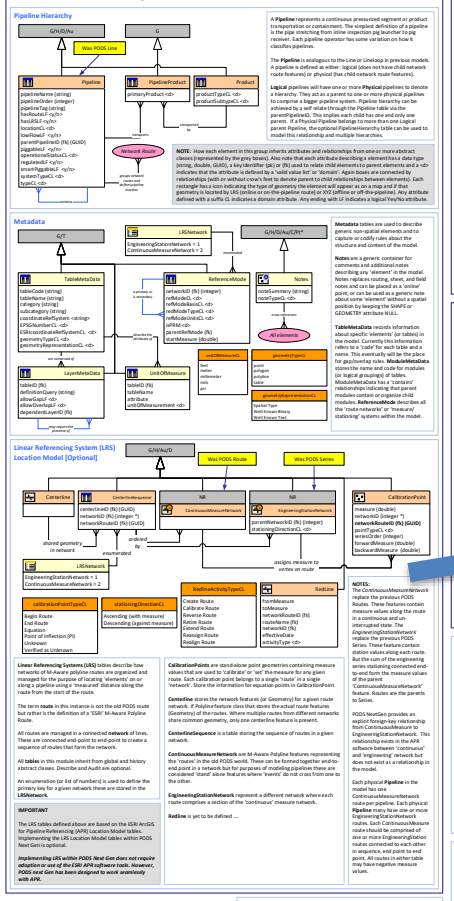
Describe is used to describe elements in the pipeline. **History** is used to track when elements were 'current' or 'active' or when they were 'historic' or 'idle'.
Audit is used to track what person created and last modified a element in the database.

PipelineFeature is used to represent how a geometric feature/shape is located on or along the pipeline. All elements in the model that have a location in space (with the exception of area features – polygons) must implement (or possess the attributes from) the PipelineFeature abstract class. Each PipelineFeature can optionally be located on or along a Network Route. **Point**, **Polyline**, **A PipelineFeature** can be EITHER an **Asset** or a **Condition** but can still inherit from any of the of the OTHER abstract classes

NetworkRoute, and **Polygon** represent physical spatial features in the model. A NetworkRoute is used to locate elements along a route using the linear referencing system (LRS) tables.

(G) – Blocks indicate the abstract class can be inherited from by elements in the model. (*) – indicates that no class can inherit directly from this abstract class.

Logical Model Design – Pipeline Hierarchy, MetaData and Linear Referencing (II)



A **Pipeline** represents a continuous pressurized segment or product transportation or containment. The simplest definition of a pipeline is the pipe stretching from inline inspection pig launcher to pig receiver. Each pipeline operator has some variation on how it classifies pipelines.

The **Pipeline** is analogous to the Line or Linelosp in previous models. A pipeline is defined as either: logical (does not have child network route features) or physical (has child network route features).

Logical pipelines will have one or more **Physical** pipelines to denote a hierarchy. They act as a parent to one or more physical pipelines to comprise a bigger pipeline system. Pipeline hierarchy can be achieved by a self relate through the Pipeline table via the parentPipelineID. This implies each child has one and only one parent. If a Physical Pipeline belongs to more than one Logical parent Pipeline, the optional PipelineHierarchy table can be used to model this relationship and multiple hierarchies.

NOTE: How each element in this group inherits attributes and relationships from one or more abstract classes (represented by the grey boxes). Also note that each attribute describing a element has a data type (string, double, GUID), a key identifier (pk) or (fk) used to relate child elements to parent elements and a <d> indicates that the attribute is defined by a 'solid value list' or 'domain'. Again boxes are connected by relationships (with or without crow's feet to denote parent to child relationships between elements). Each rectangle has a icon indicating the type of geometry the element will appear as on a map and if that geometry is located by LRS (online or on-the-pipeline route) or XYZ (offline or off-the-pipeline). Any attribute defined with a suffix CL indicates a domain attribute. Any ending with LF indicates a logical Yes/No attribute.

Metadata tables are used to describe generic non-spatial elements and to capture or codify rules about the structure and content of the model.

Notes are a generic container for comments and additional notes describing any 'element' in the model. Notes replaces routing, sheet, and field notes and can be placed as a 'online' point, or can be used as a generic note about some 'element' without a spatial position by keeping the SHAPE or GEOMETRY attribute NULL.

TableMetaData records information about specific 'elements' (or tables) in the model. Currently this information refers to a 'code' for each table and a name. This eventually will be the place for geo/overlap rules. **ModuleMetaData** stores the name and code for modules (or logical groupings) of tables. ModuleMetaData has a 'contains' relationships indicating that parent modules contain or organize child modules. **ReferenceMode** describes all the 'route network' or 'measure/stationing' systems within the model.

Linear Referencing System (LRS) tables describe how networks of M-Aware polyline routes are organized and managed for the purpose of locating 'elements' on or along a pipeline using a 'measured' distance along the route from the start of the route.

The term **route** in this instance is not the old PODS route but rather is the definition of a 'ESRI' M-Aware Polyline Route.

All routes are managed in a connected **network** of lines. These are connected end-point to end-point to create a sequence of routes that form the network.

All **tables** in this module inherit from global and history abstract classes. Describe and Audit are optional.

An enumeration (or list of numbers) is used to define the primary key for a given network these are stored in the **LRSNetwork**.

IMPORTANT

The LRS tables defined above are based on the ESRI ArcGIS for Pipeline Referencing (APR) Location Model tables. Implementing the LRS Location Model tables within PODS Next Gen is optional.

Implementing LRS within PODS Next Gen does not require adoption or use of the ESRI APR software tools. However, PODS next Gen has been designed to work seamlessly with APR.

CalibrationPoints are stand-alone point geometries containing measure values that are used to 'calibrate' or 'set' the measure for any given route. Each calibration point belongs to a single 'route' in a single 'network'. Store the information for equation points in CalibrationPoint.

Centerline stores the network features (or Geometry) for a given route network. If Polyline feature class that stores the actual route features (Geometry) of the routes. Where multiple routes from different networks share common geometry, only one centerline feature is present.

CenterlineSequence is a table storing the sequence of routes in a given network.

ContinuousMeasureNetwork are M-Aware Polyline features representing the 'routes' in the old PODS world. These can be formed together end-to-end point in a network but for purposes of modeling pipelines these are considered 'stand' alone features where 'events' do not cross from one to the other.

EngineeringStationNetwork represent a different network where each route comprises a section of the 'continuous' measure network.

Redline is yet to be defined ....

NOTES:
The *ContinuousMeasureNetwork* replace the previous PODS Routes. These features contain measure values along the route in a continuous and un-interrupted state. The *EngineeringStationNetwork* replace the previous PODS Series. These feature contain station values along each route. But the sum of the engineering series stationing connected end-to-end form the measure values of the parent 'ContinuousMeasureNetwork' feature. Routes are the parents to Series.

PODS NextGen provides an explicit foreign-key relationship from ContinuousMeasure to EngineeringStationNetwork. This relationship exists in the APR software between 'continuous' and 'engineering' network but does not exist as a relationship in the model.

Each physical **Pipeline** in the model has one ContinuousMeasureNetwork route per pipeline. Each physical **Pipeline** may have one or more EngineeringStationNetwork routes. Each ContinuousMeasure route should be comprised of one or more EngineeringStation routes connected to each other in sequence, end point to end point. All routes in either table may have negative measure values.

Version 1.0 – 2017/04/07

www.pods.org

The poster is designed to be read from upper left, downwards and the from upper right and downwards. The sections are enumerated by numbers in the title blocks (I-II-III)

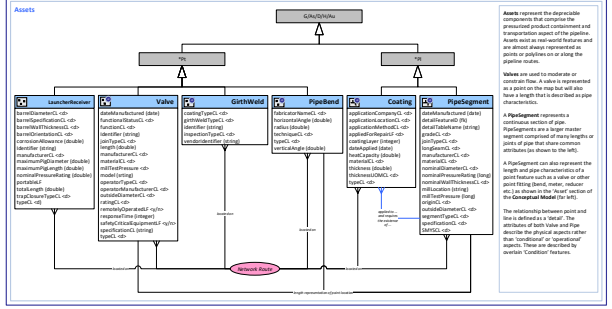
Design Notes

This poster comprises both a conceptual and logical model design notes. The poster is designed to be read from upper left, downwards and the from upper right and downwards. The sections are enumerated by numbers in the title blocks (I-II-III). For the designer it is difficult to perform and document a conceptual model and logical design without reverting back to the terms 'tables' and 'attributes' and 'relationships'. In some cases, the design is best explained in these terms.

The principles for design were to utilize what worked in previous PODS models including the relational and spatial model. The design uses existing patterns if they still make sense or there was not a better way. Every attempt was made to remove any element that was not necessary, to keep things simple, and to use common 'pipeline' terms rather than data modeler terminology. Every attempt was made to keep the abstract classes as straight forward as possible.

At this stage of the design no decision about feature, table, attribute 'SourceCL' information has been finalized. Part of 'SourceCL' construct is covered in the PipelineFeature abstract class. The final decision has not been made regarding this. Domains and Code-Lookup tables are considered to be synonymous with each other containing, at a minimum, a code and description value.

Logical Model Design – Assets (III)



Assets represent the depreciable components that comprise the pressurized product containment and transportation aspect of the pipeline. Assets exist as real-world features and are almost always represented as points or polylines on or along the pipeline routes.

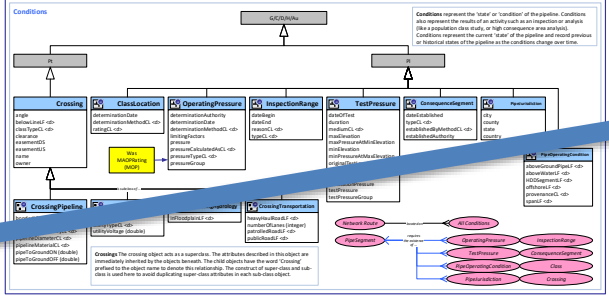
Values are used to moderate or constrain flow. A valve is represented as a point on the map but will also have a length that is described as pipe characteristic.

A **Pipelegment** represents a continuous section of pipe. PipeSegments are a larger master segment comprised of many lengths or joints of pipe that share common attributes (as shown to the left).

A PipeSegment can also represent the length and pipe characteristics of a point feature such as a valve or other point fitting (bend, meter, reducer etc.) as shown in the 'Asset' section of the **Conceptual Model** (the left).

The relationship between point and line is defined as a 'detail'. The attributes of both Valve and Pipe describe the physical aspects rather than 'conditional' or 'operational' aspects. These are described by overlain 'Condition' features.

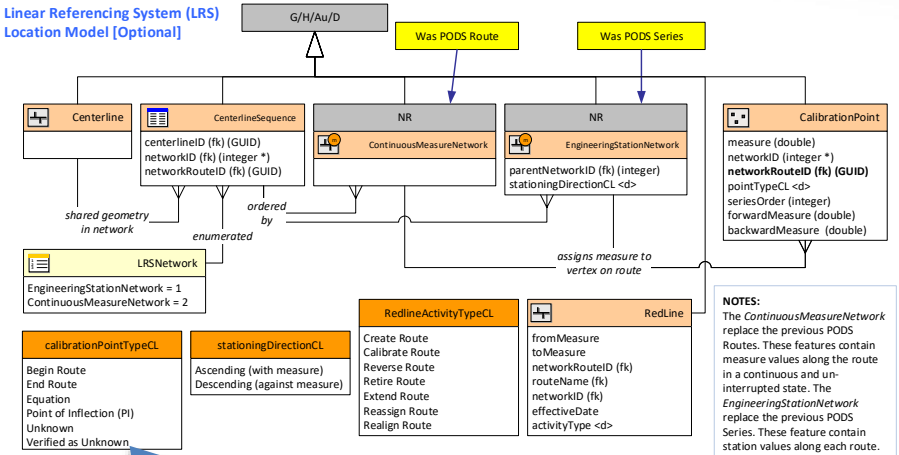
Logical Model Design – Conditions (IV)



Conditions represent the 'state' or 'condition' of the pipeline. Conditions also represent the results of an activity such as an inspection or analysis (like a population class study, or high consequence area analysis). Conditions represent the current 'state' of the pipeline and record previous or historical states of the pipeline as the conditions change over time.

Crossings The crossing object acts as a superclass. The attributes described in this object are immediately inherited by the objects beneath. The child objects have the word 'Crossing' prefixed to the object name to denote this relationship. The construct of super-class and sub-class is used here to avoid duplicating super-class attributes in each sub-class object.

Linear Referencing System (LRS) Location Model [Optional]



Linear Referencing System (LRS) tables describe how networks of M-Aware polyline routes are organized and managed for the purpose of locating 'elements' on or along a pipeline using a 'measured' distance along the route from the start of the route.

The term **route** in this instance is not the old PODS route but rather is the definition of a 'ESRI' M-Aware Polyline Route.

All routes are managed in a connected **network** of lines. These are connected end-point to end-point to create a sequence of routes that form the network.

All **tables** in this module inherit from global and history abstract classes. Describe and Audit are optional.

An enumeration (or list of numbers) is used to define the primary key for a given network these are stored in the **LRSNetwork**.

IMPORTANT

The LRS tables defined above are based on the ESRI ArcGIS for Pipeline Referencing (APR) Location Model tables. Implementing the LRS Location Model tables within PODS Next Gen is optional.

Implementing LRS within PODS Next Gen does not require adoption or use of the ESRI APR software tools. However, PODS next Gen has been designed to work seamlessly with APR.

CalibrationPoints are stand-alone point geometries containing measure values that are used to 'calibrate' or 'set' the measure for any given route. Each calibration point belongs to a single 'route' in a single 'network'. Store the information for equation points in CalibrationPoint.

Centerline stores the network features (or Geometry) for a given route network. If Polyline feature class that stores the actual route features (Geometry) of the routes. Where multiple routes from different networks share common geometry, only one centerline feature is present.

CenterlineSequence is a table storing the sequence of routes in a given network.

ContinuousMeasureNetwork are M-Aware Polyline features representing the 'routes' in the old PODS world. These can be formed together end-to-end point in a network but for purposes of modeling pipelines these are considered 'stand' alone features where 'events' do not cross from one to the other.

EngineeringStationNetwork represent a different network where each route comprises a section of the 'continuous' measure network.

Redline is yet to be defined ....

NOTES:
The *ContinuousMeasureNetwork* replace the previous PODS Routes. These features contain measure values along the route in a continuous and un-interrupted state. The *EngineeringStationNetwork* replace the previous PODS Series. These feature contain station values along each route. But the sum of the engineering series stationing connected end-to-end form the measure values of the parent 'ContinuousMeasureNetwork' feature. Routes are the parents to Series.

PODS NextGen provides an explicit foreign-key relationship from ContinuousMeasure to EngineeringStationNetwork. This relationship exists in the APR software between 'continuous' and 'engineering' network but does not exist as a relationship in the model.

Each physical **Pipeline** in the model has one ContinuousMeasureNetwork route per pipeline. Each physical **Pipeline** may have one or more EngineeringStationNetwork routes. Each ContinuousMeasure route should be comprised of one or more EngineeringStation routes connected to each other in sequence, end point to end point. All routes in either table may have negative measure values.

Implementing APR with PODS Next Gen

- Working with two clients implementing APR (one on UPDM, one on PODS Lite)
  - One purely transmission (tons of ILI data)
  - One mixture of gathering and transmission
  - Data modeling and mapping exercise
    - Utilized PODS Lite and extended the schema to match existing data model (ahead of the curve of PODS for releasing modules)

- PODS Lite installer builds the required schema
  - PODS Lite was missing a PipelineName (had PipelineID)

- APR Provides multiple network support – continuous, engineering
  - Implemented everything on engineering network and allow ESRI to add derived network attributes
  - Decision on what network to utilize for locating events on

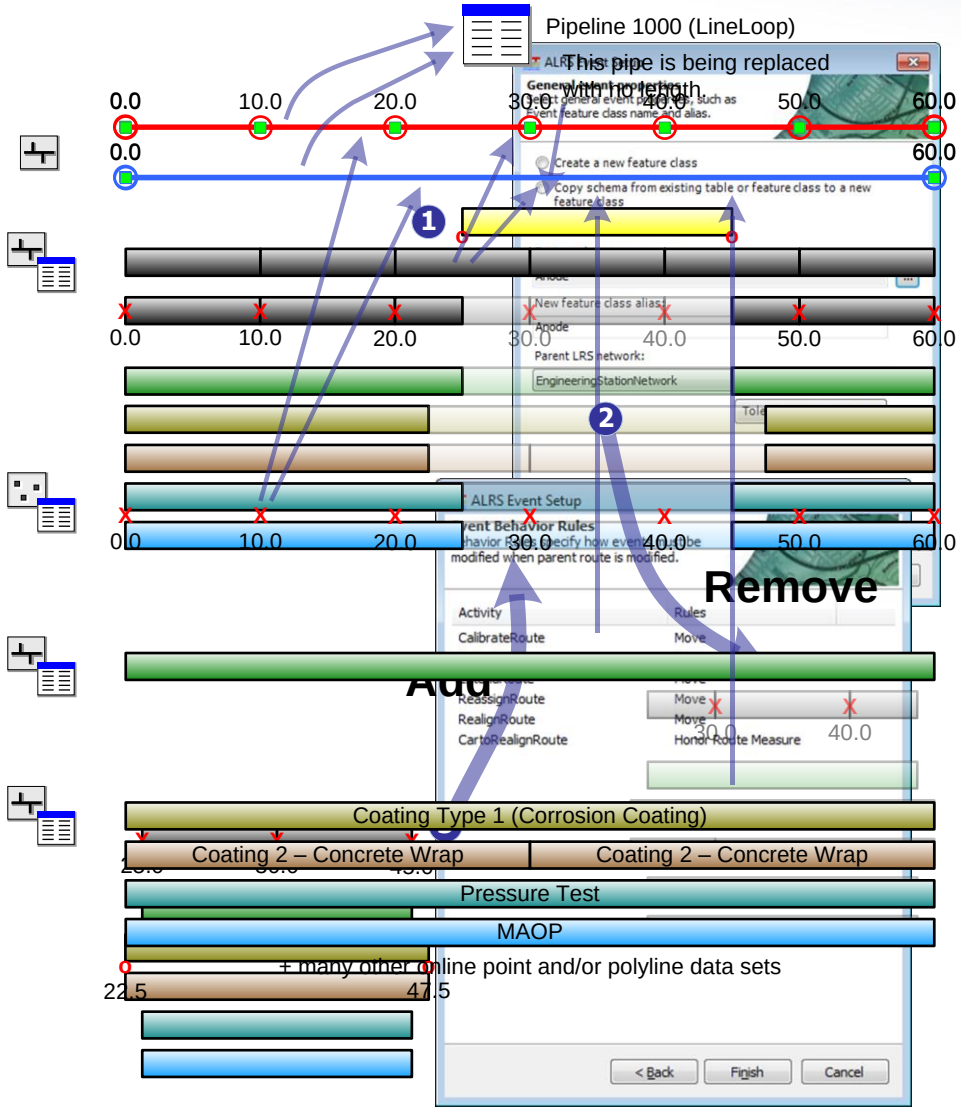
APR Software Components

- APR 10.5 (ArcGIS Pro 1.1+, ArcGIS Desktop/Server 10.5)

- Utilizes ArcGIS Desktop, Arc PRO and ArcGIS Server
  - ESRI wants users to utilize ArcPRO
  - ArcPRO offers full support for GUID datatype (for primary keys)
  - Some tools are still in ArcGIS Desktop (don't support GUIDS)
    - Loading Routes
    - Being able to configure LRS events for derived networks
  - Python script to configure LRS networks and event responses (Configuration is 100% scriptable)
  - All tools need to be moved to ArcPRO

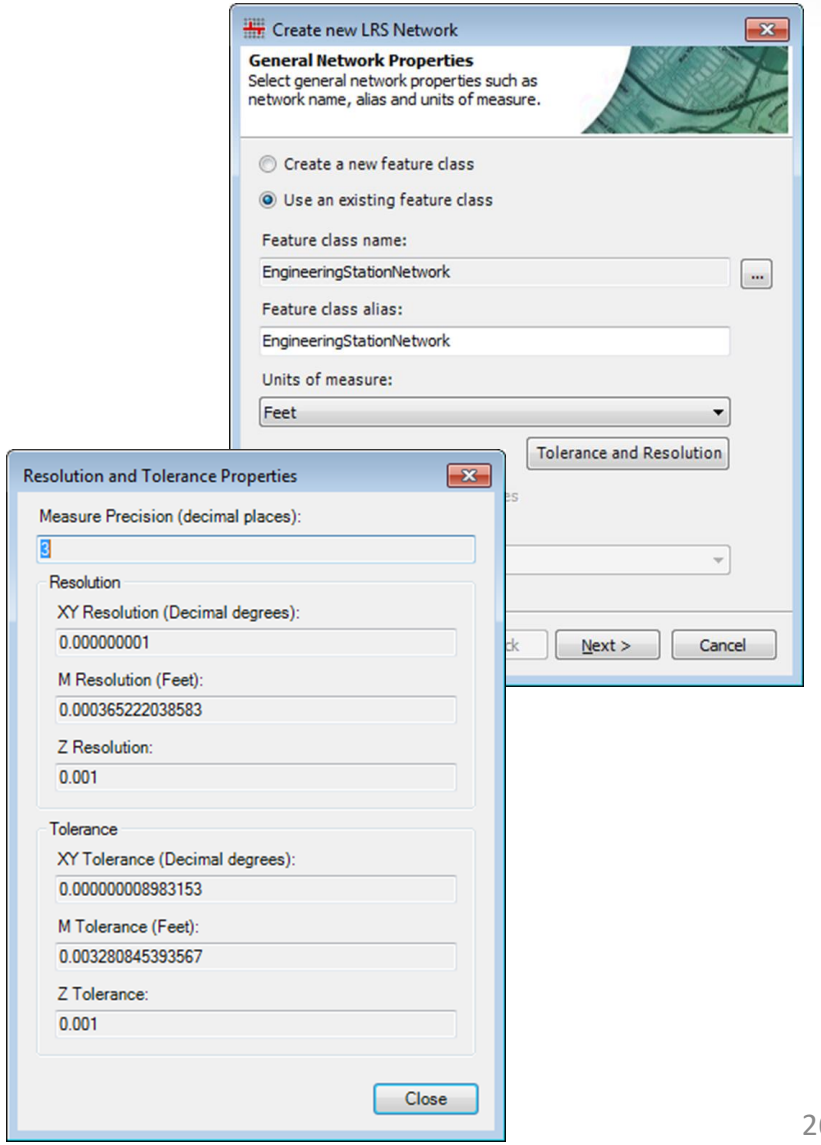
Event Configuration

- "Retire" in APR may not be the same as a RETIRE in pipeline terminology
  - Use Case – Want to cut out or isolate all events between KP 2.5 and 5.0
    - APR will not cut or split these events, needs to be performed manually before-hand OTHERWISE …
    - Any linear event spanning or crossing over or passing-thru this section will be retired
- "Stay-Put" – may not be the same as in APDM/PODS data models
  - CLEditResponse=Relative or Absolute
  - The geometry may change it's position during a re-route (or measure update) but the measure stay's put
  - Change the work flow from 'edit route to update events' to 'update events and then edit route'



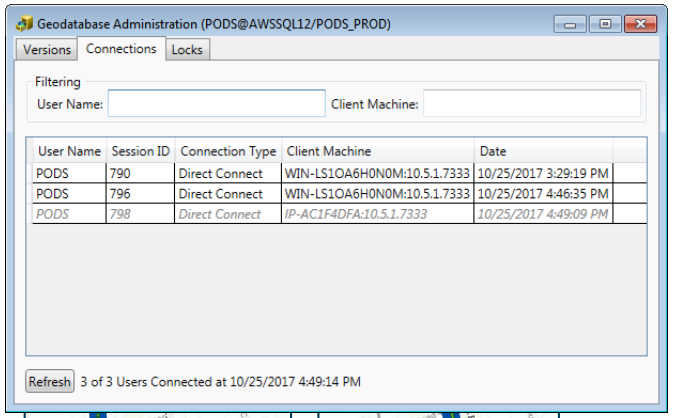
Loading Routes and Spatial Tolerance/Precision

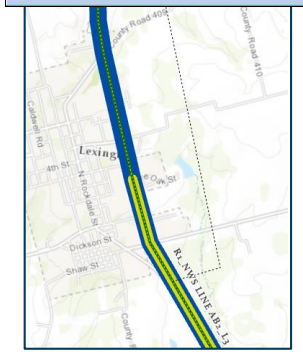
- If you utilize existing schema to configure the ALRS network ensure that the data and the spatial resolution and tolerance are congruous
- Required some testing and configuration trial-and-error to ensure that the existing schema was supported for the data being loaded into the create the LRS network features



Event Editor Extension

- Web-based server technology used for editing event information
- Metadata Attributes
  - Utilize the geodatabase editor extension tool to update CreatedBy/ModifiedBy data
  - PODS Lite offers more pipeline specific metadata attributes then core APR so these must be updated manually (not performed out of the box by APR tools)
- Understanding web-based feature layers, feature services and map services through the PORTAL interface for managing editing permissions important
  - Event editor and map service creates a lock on the geodatabase
- Understand how the ESRI time-slicing attributes works
  - Splitting a pipe segment creates two segments, one which maintains the original PK attribute

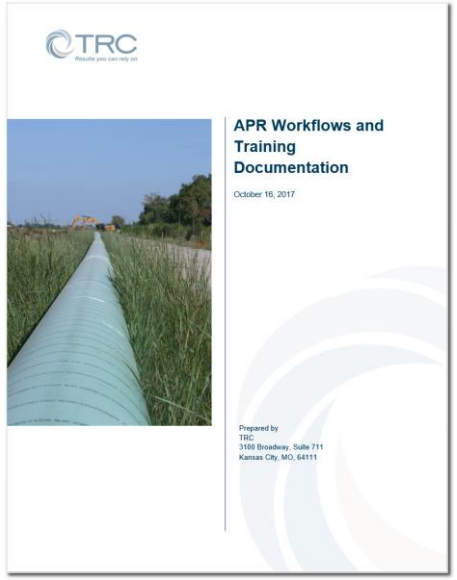




Opportunity for Training and Customization

- Documentation of Workflows

- Customization of APR to make it more ‘pipeline specific’ in terms of functionality

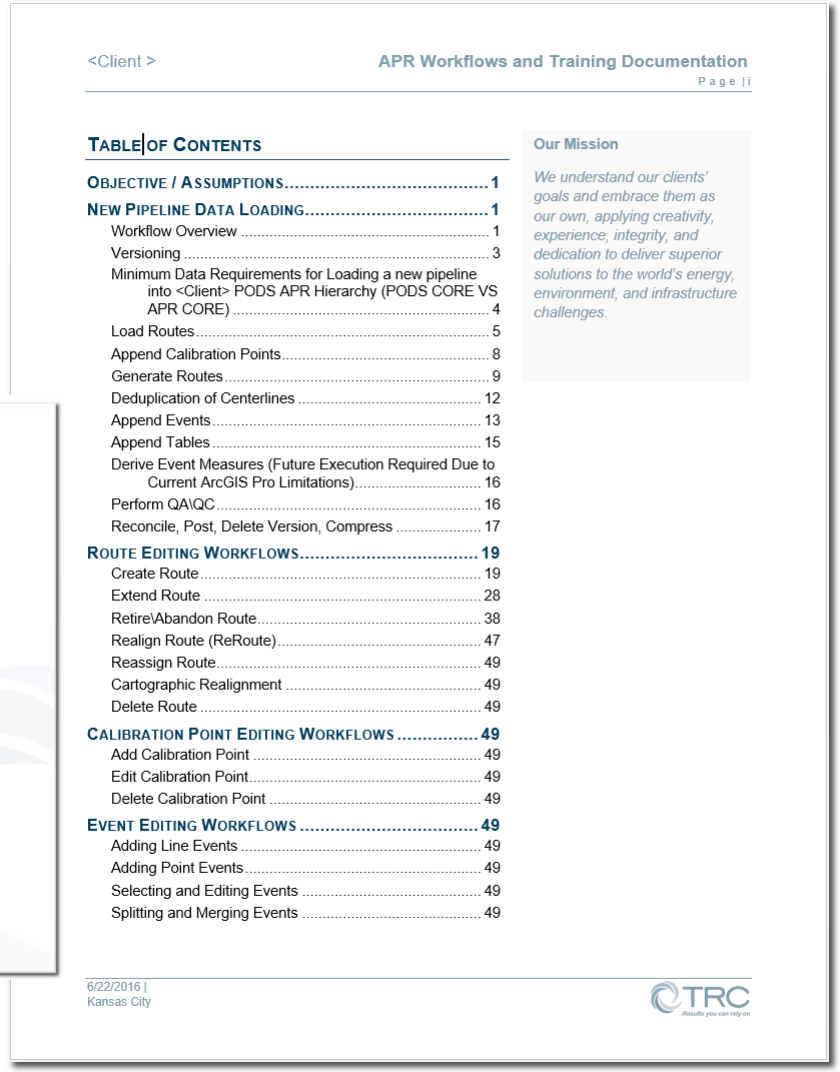


TRC

APR Workflows and Training Documentation

October 16, 2017

Prepared by
TRC
3100 Broadway, Suite 711
Kansas City, MO, 64111



<Client >

APR Workflows and Training Documentation

Page | i

TABLE OF CONTENTS

OBJECTIVE / ASSUMPTIONS 1
NEW PIPELINE DATA LOADING 1
Workflow Overview 1
Versioning 3
Minimum Data Requirements for Loading a new pipeline into <Client> PODS APR Hierarchy (PODS CORE VS APR CORE) 4
Load Routes 5
Append Calibration Points 8
Generate Routes 9
Deduplication of Centerlines 12
Append Events 13
Append Tables 15
Derive Event Measures (Future Execution Required Due to Current ArcGIS Pro Limitations) 16
Perform QA\QC 16
Reconcile, Post, Delete Version, Compress 17
ROUTE EDITING WORKFLOWS 19
Create Route 19
Extend Route 28
Retire\Abandon Route 38
Realign Route (ReRoute) 47
Reassign Route 49
Cartographic Realignment 49
Delete Route 49
CALIBRATION POINT EDITING WORKFLOWS 49
Add Calibration Point 49
Edit Calibration Point 49
Delete Calibration Point 49
EVENT EDITING WORKFLOWS 49
Adding Line Events 49
Adding Point Events 49
Selecting and Editing Events 49
Splitting and Merging Events 49

Our Mission

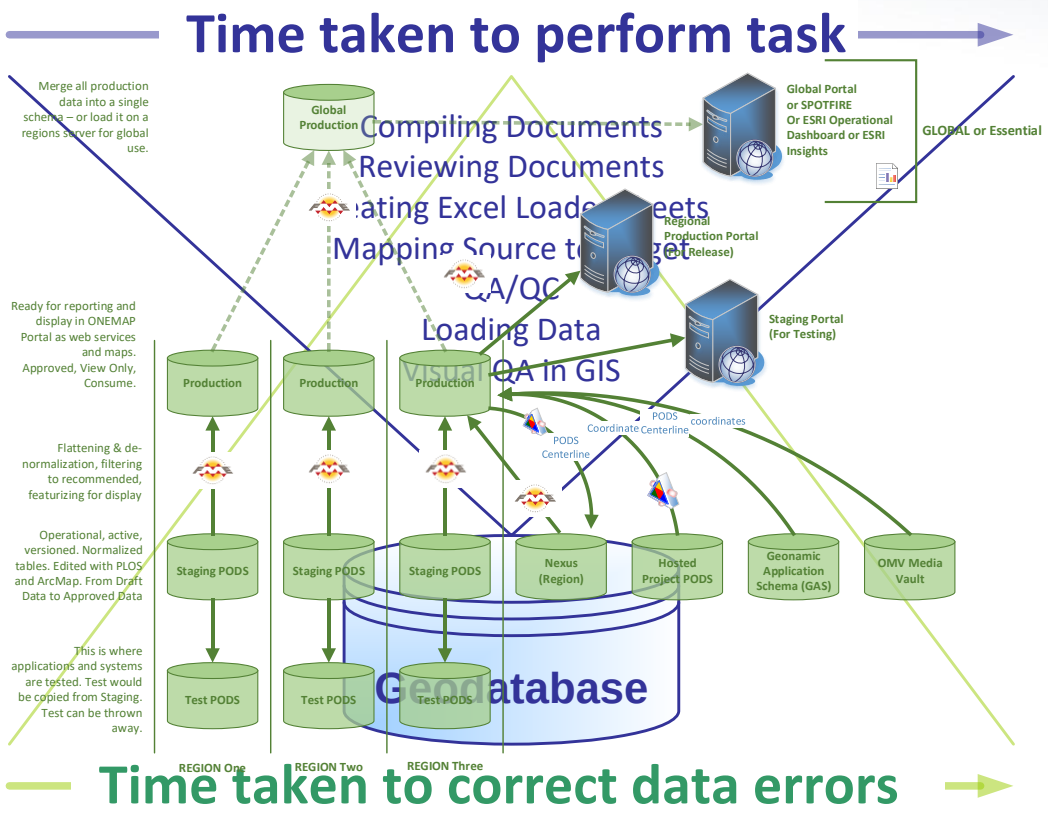
We understand our clients' goals and embrace them as our own, applying creativity, experience, integrity, and dedication to deliver superior solutions to the world's energy, environment, and infrastructure challenges.

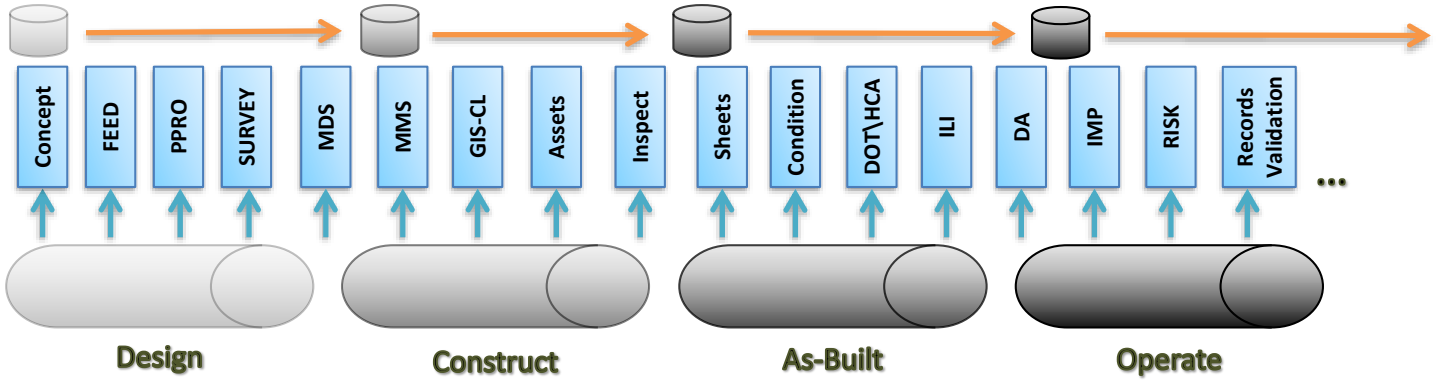
6/22/2016 |
Kansas City

TRC

Concluding Thoughts

- Value of PODS Association and PODS Data Model
- Pipeline Data Management
  - System of Record for Location of Assets
- Integrate data at the query/publication layer
  - Integrate systems by ID, Coordinate, Tag, Keywords
  - Let systems do what they do well
  - Let users build data they way they know how
- Offer a consistent technology platform for integration
- Cradle to grave digital mind-set





Thank you

Questions?

Peter Veenstra, TRC/PODS

P: 011-816-820-7841

E: pveenstra@trcsolutions.com

TRC's Guiding Principles

Our Mission

We understand our clients' goals and embrace them as our own, applying creativity, experience, integrity and dedication to deliver superior solutions to the world's energy, environment and infrastructure challenges.

Our Vision

We will solve the challenges of making the Earth a better place to live – community by community and project by project.

Our Values

We commit to these values to guide our decisions and our behaviors:

Safety: We create a working environment that promotes safe performance.

Quality: We always strive for excellence in the services we provide and in the results we produce for our clients.

Integrity: We are committed to the highest ethical standards.

Creativity: We believe in looking at challenges and opportunities from new angles and in exercising our curiosity.

Accountability: We take responsibility for all of our decisions and actions.

Teamwork: We work together to succeed.

Passion: We deliver superior results because we care deeply about what we do.