WITSML v2.0
Release Candidate Overview

Jay Hollingsworth
CTO
This morning:

» Energistics
» WITSML
» Standards v2.0
» WITSML 2.0
  • Data objects
  • API (ETP)
» Release Candidate contents
Who are we? (Hint: we are not a vendor...)

» Energistics is a global, non-profit, membership consortium focused on developing open data exchange standards in the upstream oil and gas industry. We have served the industry for more than 25 years.

» Our membership consists of E&P companies, oilfield service companies, software vendors, system integrators, regulatory agencies and the global standards community

» Our standards are developed by workgroups (known as Special Interest Groups, or SIGs) made up of industry experts from our member companies

» In short, the standards are created by the industry for the industry
Energistics Family of Standards

UNIVERSAL INTEROPERABILITY

DRILLING

<WITSML/™>

PRODUCTION

<PRODML/™>

RESERVOIR

<RESQML/™>

COMMON TECHNICAL ARCHITECTURE
Version 1.X Use Cases

» Consistent high-quality transfer of wellbore and drilling-related data
  • Data transfer to real-time operations centers
    ▪ Reference objects – Well and Wellbore
    ▪ Growing objects – Log (time, depth), Trajectory, Mudlog
    ▪ Snapshots in time – with “report” information
  • Move well-related data between applications
  • Real-time availability of drilling operations
WITSML V1.4.1 Overview

» Set of schemas defining 27 primary objects
  • Well, wellbore, logs, etc. used in drilling operations
  • Enumerations file (enumValues.xml)

» API defining server (and client) behavior
  • Simple methods (AddToStore, GetFromStore)
New Requirements on Transfer Standards

» Big data/analytics
  • Analytics on data in motion

» High-performance transfer standards

» Broader workflows – not just wellsite to office
  • Application to application
  • File-based transfers
  • Archival workflows
  • Expanded metadata
How Are Standards v2.0 Better

» Integrated
» Programmer oriented
» New workflow support
» Training available
» New underlying technology
Integrated

» Among the MLs
» Between the standards bodies
  • SEGY/SEGD in epc
  • HDF use
  • OGC in MLs
  • IEP/ISO 19115
  • MathML coming
Programmer Oriented

» More convenient XML style
» Better documented
» Open source code
» Devkits
New Workflow Support

» Server and serverless

» Data management workflows
  • Data quality/assurance
  • Archival workflows
  • Data heritage
  • Metadata
Training

» Training first conducted in 2015
  • Three private classes already delivered/scheduled

» Up to a 3-day class on each ML
  • WITSML public class earlier this year
  • Further public classes to be scheduled

» Webinars
WITSML v2.0

» Continues to provide XML “data objects”
» Is based on the Common Technical Architecture
» Deprecates the legacy SOAP API, replaced by ETP v.1.1+
» Has a simplified XML schema structure & fewer files
» Data object documentation
What Can v2.0 Do I Couldn’t Do Before?

» True, secure, low-latency data streaming
» Data quality assurance
» Wellbore Geology, Stimulation and Cementing design and execution

» Unlimited types and organizations of channel data
» Tracing through multiple generations of aggregating servers

» Elimination of polling traffic
WITSML 1.4.1 Data Objects

» attachment
» bhaRun
» cementJob
» changeLog
» convCore
» coordinateRefSystem
» drillReport
» fluidsReport
» formationMarker
» log
» message
» mudLog
» objectGroup
» opsReport
» rig
» risk
» sidewallCore
» stimJob
» surveyProgram
» target
» toolErrorModel
» toolErrorTermSet
» trajectory
» tubular
» wbGeometry
» well
» wellbore
WITSML 1.4.1 Data Objects vs 2.0

- attachment
- bhaRun
- **cementJob**
- changeLog
- convCore
- coordinateRefSystem
- drillReport
- fluidsReport
- **formationMarker**
- log
- message
- **mudLog**
- objectGroup
- opsReport
- rig
- risk
- **sidewallCore**
- **stimJob**
- surveyProgram
- target
- toolErrorModel
- toolErrorTermSet
- **trajectory**
- tubular
- wbGeometry
- well
- wellbore

Removed – Moved to Common – Completely Redesigned – Largely unchanged
WITSML 2.0 Data Objects

- Attachment
- BhaRun
- CementJob
- CementJobEvaluation
- Channel
- ChannelSet
- CuttingsGeology
- DepthRegImage
- DrillReport
- FluidsReport
- InterpretedGeology
- Log
- OpsReport
- Rig
- Risk
- ShowEvaluation
- StimJob
- StimJobStage
- ToolErrorModel
- ToolErrorTermSet
- Trajectory
- TrajectoryStation
- Tubular
- Well
- Wellbore
- WellboreGeology
- WellboreGeometry
- WellboreMarker
- WellboreMarkerSet
Data Objects:

» New naming style – Pascal style
» Focus on better attribute names
» More documentation for each element
» Substitution groups are just for the Aggregate object
» More use of inheritance
Data Objects: Simpler Structure

» There are two kinds of schema file structures
  • The obj_ schema file has a single global element
    ▪ This is an individually addressable “top-level” element
    ▪ Top-level elements inherit from AbstractObject
  • The other named .xsd files only contain types

» The earlier structure is deprecated
  • No plural root, no docInfo
  • No repetition of inherited elements
Data Objects: Fewer Files example

» Before: 200 files in schema folder

» WITSML 2.0 will have
  • ~27 top level objects
  • 1 common base across all MLs
  • fewer component elements

» Top level “obj_” files stay

» Component “cs_” naming goes

» Global equipt “ges_” naming goes

» Group file “grp_” naming goes

» Add one xsd file per UML package
Data Objects: Common

- One file for each typ_style
- Base types - “abstractString”
- Enum types - “well fluid”
- Data types - “timestamp”
  - Old – everything ML-defined
  - New – xs types used
- Measure types - “angle Measure”
- Quantity classes - length in “m”
  - Old – different types per ML
  - New – common across all MLs
- Shared schemas – CRS, root
New Underlying Technology - CTA

- ETP
- UML->XML (->JSON?)
- EPC/OPC & Breaking XML
- UoMs
- PWLS
- HDF
- EIP
UML: Unified Modeling Language

» Energistics uses UML to generate XML schemas
  • and other artifacts including documentation

» UML is an OMG specification, the latest is 2.4.1 at
  ○ [http://www.omg.org/spec/UML/2.4.1/](http://www.omg.org/spec/UML/2.4.1/)

» Enterprise Architect currently supports UML 2.4.1

» Energistics uses and delivers class diagrams
  • Other UML diagram types may or may not be used
Use of XML

» What Energistics’ members have always used
» Energistics standards use XML v 1.0, not 1.1
» XML is W3C recommendation - latest ed. of v 1.0 is
  o http://www.w3.org/TR/2008/REC-xml-20081126/
» Energistics does not require other XML standards
  • Like XPath, XQuery, XLink, etc.
  • Simplicity aids in uptake
ETP: Energistics Transfer Protocol

» ETP is a new data exchange specification
» Enables real-time data transfer between applications
» Is delivered as a specification and as sample code
» Works by sending pre-defined messages
  • The messages are grouped together into “protocols”
  • The description of these protocols make up the standard
» No server required, just sender and receiver
ETP: Use Across Energistics MLs

» ETP was developed initially for WITSML™
  • Since WITSML is not a truly real-time transfer

» The other MLs will use it as well

» ETP can be used
  • For any kind of data transfer
  • From the field to the office
  • Between applications in the office
  • For any sensor-based M2M application (IIoT)
ETP: Protocols

ETP currently consists of eight child protocols:

0: Core
1: ChannelStreaming
2: ChannelDataFrame
3: Discovery
4: Store
5: StoreNotification
6: GrowingObject
7: DataArray
ETP: Protocols

ETP currently consists of eight child protocols:

0: Core – Creates and manages ETP sessions
1: ChannelStreaming – Exchanges channel-oriented data
2: ChannelDataFrame – Exchanges frame-based data
3: Discovery – Understand the contents of a data store
4: Store – Perform CRUD operations on data in a store
5: StoreNotification – Receive notification of data changes
6: GrowingObject – Manage growing parts of data objects
7: DataArray – Transfer large, binary arrays
ETP: WebSocket

» ETP is itself a sub-protocol of WebSocket
» ETP uses web ports to reduce connectivity problems
» The messages are payload data in Websocket frames
» Messages can travel in both directions
  • Used for discovery and later for query
ETP: Avro

ETP uses a subset of the Avro 1.7.5 functionality

- ETP defines all messages using the Avro schema file format
  - The Avro schemas are managed in and produced from UML by EA
- All messages on the wire are serialized per the Avro rules
- ETP uses Avro additional schema attributes
- ETP does not use Avro RPC and container file facilities

ETP supports Avro use of both binary and JSON data
ETP: JSON

» The Avro schemas are created in JSON for Avro use
» ETP also supports JSON encoding of data via Avro
ETP: git

» ETP is issued as a formal specification
» For developers, IT artifacts are also available via git
» git is a widely-used version control system
  • The commercial version used for ETP is Bitucket
» The Bitbucket repository also holds the documents
» The IT artifacts are UML and Avro schemas in JSON
» Source code in several languages is also available
ETP: Distribution Methods

» The availability of source depends on the language
  • C# code is delivered as a nuget package
  • Javascript (node.js) is distributed via mpm
  • Java and C++ are in the Energistics Bitbucket repository

» Source could include a full sample implementation
  • Or might only contain proxy classes

» All code is contributed and maintained by members
WITSML v2.0 RC contents

» XML schemas – the standard
» XSLT transforms
  • From both WITSML 1.3.1 and WITSML 1.4.1.1
» Sample XML files
  • Additional files available soon from Statoil/Kongsberg
» Documentation
  • Improving as the review period continues
» Feedback Form in zip file
  • Comments due June 30
Wrap Up

ANY QUESTIONS?