Completion Data-object Usage Guide

For use with WITSML version 1.4.1.1 and PRODML version 1.2.2

WITSML Overview	The Wellsite Information Transfer Standard Mark-up Language (WITSML) consists of XML data-object definitions and a Web services specification developed to promote the right-time, seamless flow of well data between operators and service companies to speed and enhance decision making.
Version	**1.0**
Abstract	This guide provides an overview of the Completion data-object, which provides a standard way to describe wellbore completions so that data may be transferred from one software application to another, accurately, reliably, and consistently. While the completion data-object is officially part of the WITSML specification, it may also be used with other Energistics specifications such as PRODML and RESQML.
Prepared by	Energistics and the Completions Data-object Work Group
Date published	9 August 2013
Document type	Usage guide
Keywords:	standards, energy, data, information, process, completion, drilling, production

energistics™ The Energy Standards Resource Centre

Document Information	
DOCUMENT VERSION	Version 1
DATE	9 August 2013
Language	US English

Acknowledgements

Thanks are due to the members of the WITSML, PRODML and RESQML SIGs who joined forces to develop the Completion data-object.

In particular, the efforts donated by members from the following companies must be acknowledged: Chevron, Weatherford, Peleton, Schlumberger, Paradigm, ExxonMobil, and BP.

Amendment History			
Version	**Date**	**Comment**	**By**
1.0	9 August 2013	Production Release. Finalize content following public review. Amendments to equipment properties/loader file description to clarify how this works.	Laurence Ormerod and Completion Object Work Group

Table of Contents

Executive Summary

The Completion data-object was designed to support the exchange of data describing the completion equipment inside a well. It is an XML data-object schema and part of the WITSML data-exchange standard. It includes:

- The physical description of the hardware
- The operation (or "events") by which the hardware was added to or removed from the well

With this information, it is possible to support these high-level use cases:

- "Snapshot": the completion equipment at a given instant in time.
- "Change Log": the events that change the completion between any two time instances.
- "Cumulative History": all of the completion equipment and all of the events that changed it over the life of the well.

In addition to equipment and events, the Completion data-object supports reporting flow at the level of the whole well, co-mingled flow within a wellbore, and flow through individual physical connections, such as perforations to the reservoir. The data-object also supports modeling complex and multi-lateral wells; for example, dual completion strings, gravel pack completions, and ESP or rod-pumped well completions.

Benefits. Use of the data-object is expected to dramatically improve data quality and reliability, by eliminating the manual processes currently used, such as export and import of spreadsheets, re-keying data into databases, and the use of paper records. Automated, more reliable data-exchange should benefit efficiency, safety, well operations, and planning.

This document introduces the reader to the Completion data-object in stages:

- Chapter 1 provides an overview and presents the business case and scope of the specification.
- Chapter 2 explains key concepts and main usage patterns.
- Chapter 3 provides XML code examples for each of the key use cases.
- Chapter 4 provides XML code examples for commonly used equipment, scenarios, and events.

1 Introduction

The Completion data-object is an XML specification for describing a well completion throughout its whole history. With a standardized description, the completion data can be accurately, reliably, and consistently exchanged between the many software applications used to plan, manage, and record work on a completion over the life of a well.

The Completion data-object is officially part of the Energistics WITSML data exchange specification (which defines wells) but has been designed to also work with Energistics PRODML (production) and RESQML (earth modeling/reservoir) data exchange specifications. Note that specific production and reservoir domain requirements have not been addressed in this first release, but are expected to be added in later releases.

This usage guide describes the business case and value of the Completion data-object, provides an overview of the key concepts underlying it, and then presents details and examples about how it can be used and implemented.

1.1 The Business Case

Over the life a well, the configuration of a completion is typically changed many times, a process that involves changes in operational "ownership" between operator and service companies, drilling & completions, production operations, and workover teams. Each time the completion is worked on and changes hands, the project team taking over needs to know the current configuration—information that is vital for safety and to optimize the efficiency of completing the workover.

A key assumption (based on experience) is that drilling and operations organizations use different software applications from different vendors. With these different systems, the transfer of completions data between organizations can be a very manual processing, involving reformatting or, in some cases, re-entering all data into the new system. This manual data transfer process is inefficient and introduces opportunities for errors, which can jeopardize safety and schedules.

1.1.1 Business Drivers and Benefits

The goal of the Completion data-object is to allow completion data to be transferred from one software application to another accurately, reliably, consistently. The business drivers for developing this data-object include:

- Eliminate multiple data entry requirements (when transferring from different systems) thereby improving accuracy and saving time.
- Improve data consistency, accuracy, quality.
- Improve safety. Knowledge of what is done hole will improve operations.
- Faster access to required data.
- Improve planning and scheduling (through faster access to better quality data and by eliminating manual re-entry of data).
- Reduce reliance on paper or spread sheet well files.

1.2 Scope and Use Cases

The initial scope for this version of the Completion data-object includes the completion equipment from the sandface connection to the reservoir, up to and including the tubing spool/tubing hanger, for use with:

- Use cases as defined in this guide.
- WITSML v1.4.1.1 and PRODML schema v1.2.2.

The Completion data-object can be used to transfer or exchange data between software applications or databases for these high-level workflows:

- Drilling and initial installation of the completion.

- Handover from drilling to production.
- Transfer of data between engineering applications.
- Well services through the life of an asset (which may involve links to other WITSML or similar data-objects, for example, for the details of drilling, cementing, fracturing and other “job” types yet to be supported).

The three use cases addressed in the current release that support these workflows include the following (For use case details, see Section 2.1, page 10 and for code examples, see Chapter 3):

- **Snap shot** of a completion transferred at a point in time, which supports the "what is the current status" request, developing a completion diagram, or performing nodal analysis to troubleshoot a new production problem.
- **Change Log** of events that change a completion between one time and another. If an asset team is doing some longer term analysis, they might want to examine well performance over a specific period of time, and how those changes to the completion have impacted performance. This use case also supports a well services contractor supplying a digital version of their service job back to the operator (for example, “Between times A and B, Service Company performed the following operations…”).
- **Cumulative History**, which includes all data over the life of a completion. An operator working with multiple service companies or purchasing a new asset may have a need to synchronize various databases and need the complete historical information for a well’s completion. All of this data, if retrieved, should allow you to create the data for either of the two use cases listed above.

1.3 Audience and Purpose of this Document

1.3.1 Audience

- **Domain/business professionals and managers**. The first two chapters provide high-level information about business value, benefits, and key concepts.
- **Information technology (IT) professionals.** Intended for IT people, such as software developers and product architects, who develop WITSML-enabled and PRODML-enabled products and need to understand the details of how the completion data-object works.

1.3.2 Purpose

This guide was developed to:

- Give an overview of Completion data-object and explain the business value and benefits of its use.
- Explain key concepts about the data-object design to help guide how it can be used in software applications to facilitate the transfer of completion data.
- Provide use cases and related code examples to help developers implement the completion data-object.

1.4 Resources to Get You Started with the Completion Data-object

The following files comprise the release of the Completion data-object, and all may be found in the download zip file available on the Energistics website:

- Usage guide (this document).
- Power Point (WITSML-PRODML Completion Data-object Overview FINAL.pptx). This provides a quick overview of the Completion data-object purpose, use cases, and high level concepts.
- Completion data-object schemas. The schemas are documented with all element definitions.
- Example files. The Completion data-object download contains example XML files, which you can use as a basis for and modify to create your own data-objects.

Help is available from Energistics on a limited basis. Energistics is a not-for-profit organization and not a commercial supplier of consultancy. Energistics can put you in touch with contractors who may be able to help on a commercial basis. Contact Energistics via its website, http://www.energistics.org/contact-us.

Note that the best way to become familiar with any of the Energistics standards is by joining the organization as a member company. The PRODML and WITSML Special Interest Groups (SIGs) provide a good way to learn about the standards and a way to participate in their further development.

1.4.1 Not Familiar with WITSML or PRODML?

Additionally, these related resources may be helpful for people who are not familiar with Energistics standards. For the latest versions of these related specifications, including schemas, APIs, and other documentation, visit the Energistics website, http://www.energistics.org.

- WITSML, Wellsite Information Transfer Standard Mark-up Language (http://www.energistics.org/drilling-completions-interventions/witsml-standards)
- PRODML (http://www.energistics.org/production/prodml-standards)

1.5 Background and History

The work to develop the Completion data-object began with a company-internal project by Chevron in conjunction with several service/software companies. As an Energistics member and user of the WITSML, PRODML and RESQML specifications, Chevron recognized the need for standardization of completion data across the industry and proposed the Completion data-object as an Energistics project.

The Energistics Board and members of the three major Energistics special interest groups (SIGs) (WITSML, PRODML and RESQML) recognized the need and value of the project and expressed strong interest in the work. The Completion Object Work Group (COWG) was formed to perform the work of defining requirements, with the Chevron project team a significant contributor.

The WITSML specification defines wells, so the resulting Completion data-object schema and COWG are “officially” part of WITSML. However, a wellbore completion impacts work flows along the entire E&P value chain and Energistics (like the industry) is moving towards better integration of the key domain workflows and specifications, so the Completion data-object has been designed with these goals in mind.

1.6 Future Plans

Future plans for further developing the Completion data-object include addressing other use cases, for example:

- Tree and surface equipment associated with the well.
- Surveillance, flow path definitions, for example, what are the paths through the hardware of flows that we report and measure and where are the sensors relative to these?—especially important in complex completions.
- Optimization, including addition of inflow performance and description, link to reservoir units made explicit and vertical lift performance.
- Further “xxxJob” data-objects, for example, perfJob, gravelPackJob, which can contain details of operations (“*how it was done?*”) as opposed to the resultant change to the completion (“*what was the end result?*”).

2 Key Concepts and Overview of Use Cases

This chapter describes important key concepts about the organization and structure of the Completion data-object and the use cases that it addresses.

For detailed XML examples of how to implement these use cases, see Chapter 3, page 23.

2.1 Use Cases and Related Concepts

The current version of the Completion data-object has been designed to address these use cases:

- "Snapshot" of a completion at one instant in time
- "Change Log" of events, capturing the changes to a completion between two moments in time
- "Cumulative History" of the well over its whole life

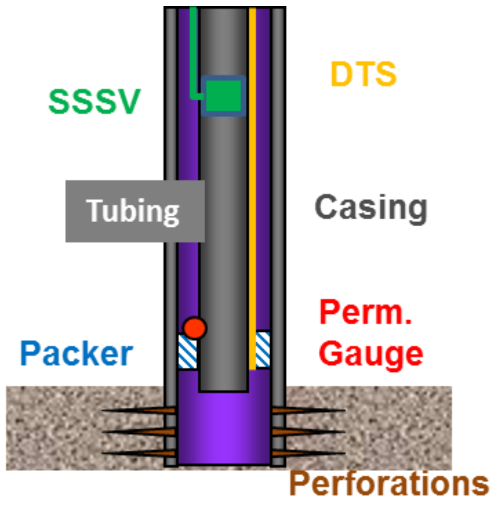
These use cases relate to each other to enable systems to track progress and changes made to the completion, and to transfer the actual configuration at any instant in time. However, while the Completion data-object supports these use cases, the ability of individual software applications to support them may vary.

These concepts, which are consistent with how WITSML works, are explained below with diagrams to relate them to real-world assets and activities.

Note that while the well completion hardware is modeled explicitly in detail and operations are handled with as much detail as existing data-objects allow, flow paths of fluids through the whole completion are modeled in a limited sense. The connections to the reservoir are modeled, and these are associated with ports (or flows) at the wellhead. However, currently the detailed path in between is not explicitly described.

2.1.1 Snapshot Use Case

The Snapshot use case refers to exchanging data that represents the well completion at a given instant in time. Everything in the data transfer represents equipment in the completion at the instant specified by the user. Conceptually, a snapshot corresponds to a completion diagram (Figure 1). However, a Snapshot does NOT include any information about how components were installed or any service-related information.



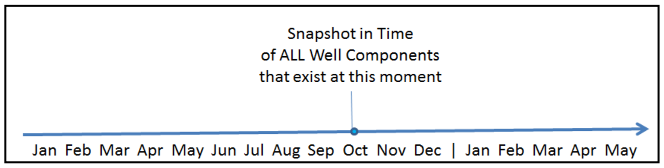

Snapshot in Time
of ALL Well Components
that exist at this moment
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec | Jan Feb Mar Apr May

Figure 1. The Snapshot use case is effectively the same as a completion diagram.

Note that the snapshot can include flow paths of fluids through the whole completion, modeled in a limited sense. The connections to the reservoir are modeled, and these are associated with ports (or flows) at the wellhead. However, the detailed path in between is not explicitly described.

2.1.2 Change Log Use Case

The Change Log use case is about exchanging ONLY the data that has changed over a specified time period. Components that did not change are NOT exchanged. This capability is useful to allow repeated synchronization of applications or databases.

Well and completion configurations change by performing or executing a "job", a single completion or well service (For more information, see Section 2.1.4, page 13). Information about the job and resulting changes are captured in an "event ledger" called the Well CM Ledger (CM stands for Construction and Maintenance), which captures necessary data about both the job that created the change and the result (e.g., equipment change) of the job. (For more information about the Well CM Ledger, see Section 2.2.2, page 19.)

Figure 2 shows an example perforation job (in a well that has already been cased, tubing has been run, and is ready to produce).

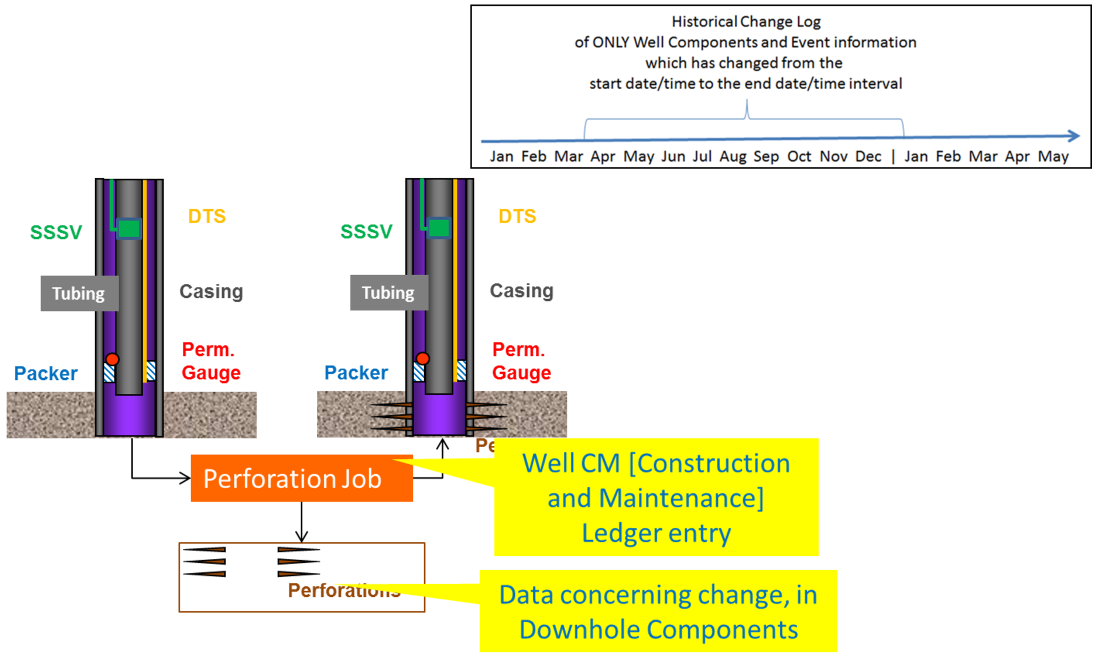

Historical Change Log
of ONLY Well Components and Event information
which has changed from the
start date/time to the end date/time interval
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec | Jan Feb Mar Apr May
SSSV
DTS
Tubing
Casing
Packer
Perm. Gauge
SSSV
DTS
Tubing
Casing
Packer
Perm. Gauge
Pe
Perforation Job
Perforations
Well CM [Construction and Maintenance] Ledger entry
Data concerning change, in Downhole Components

Figure 2. Example Well CM Ledger entry for a single “job”.

The ledger in this case would have a single entry for this perforation job. The entry contains data about:

- **Job as a process.** In our perforation job example, job data would include which company performed the job, when it was run, equipment used, etc.
- **Result of the job.** The equipment or change associated with the well. In this example, it would be the addition of the perforations, but can also include installation, removal or replacement of equipment.

For this perforation example, the time at which the perforations become active is recorded in the equipment data-object. Thus a snapshot for a time after the perforation job would show the perforations; whereas, a snapshot for a time before the job would not show the perforations.

2.1.3 Cumulative History

The Cumulative History use case refers to the transfer of data corresponding to the whole history of the completion.

The ledger described in the previous section is also used to determine all events and equipment changes (additions, replacements, and removals) as shown in Figure 3.

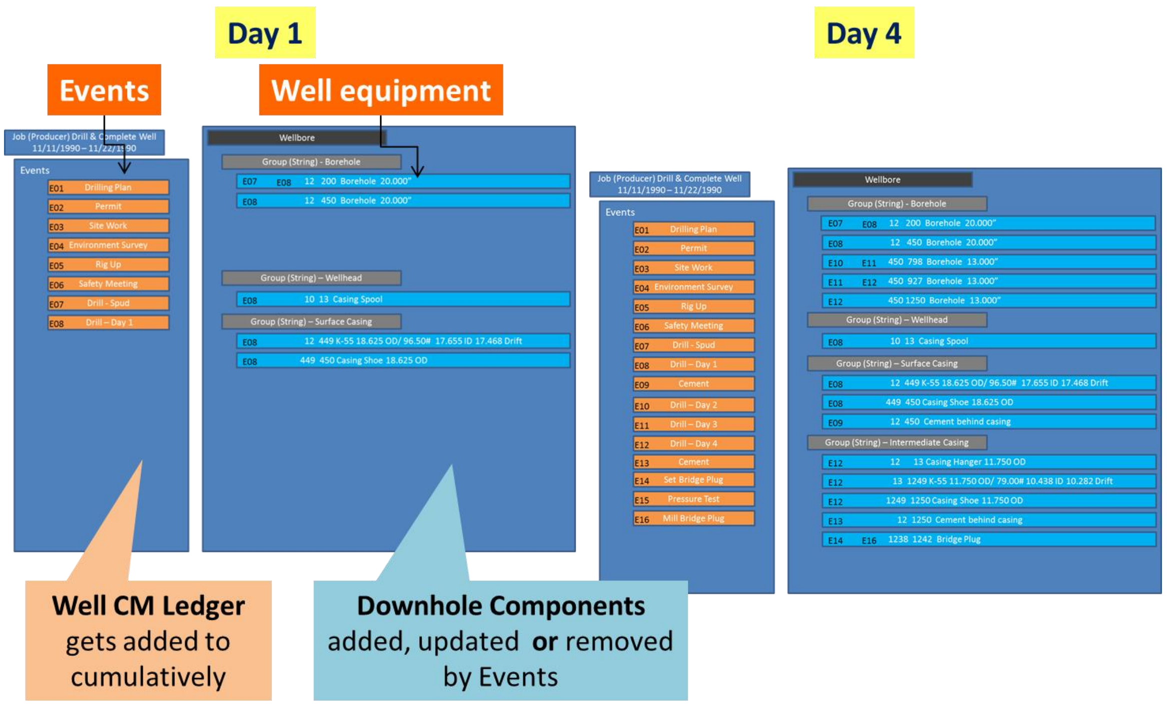

Day 1
Day 4
Events
Well equipment
Job (Producer) Drill & Complete Well 11/11/1990 – 11/22/1990
Events
E01 Drilling Plan
E02 Permit
E03 Site Work
E04 Environment Survey
E05 Rig Up
E06 Safety Meeting
E07 Drill - Spud
E08 Drill – Day 1
Wellbore
Group (String) - Borehole
E07 E08 12 200 Borehole 20.000"
E08 12 450 Borehole 20.000"
Group (String) – Wellhead
E08 10 13 Casing Spool
Group (String) – Surface Casing
E08 12 449 K-55 18.625 OD/ 96.50# 17.655 ID 17.468 Drift
E08 449 450 Casing Shoe 18.625 OD
Job (Producer) Drill & Complete Well 11/11/1990 – 11/22/1990
Events
E01 Drilling Plan
E02 Permit
E03 Site Work
E04 Environment Survey
E05 Rig Up
E06 Safety Meeting
E07 Drill - Spud
E08 Drill – Day 1
E09 Cement
E10 Drill – Day 2
E11 Drill – Day 3
E12 Drill – Day 4
E13 Cement
E14 Set Bridge Plug
E15 Pressure Test
E16 Mill Bridge Plug
Wellbore
Group (String) - Borehole
E07 E08 12 200 Borehole 20.000"
E08 12 450 Borehole 20.000"
E10 E11 450 798 Borehole 13.000"
E11 E12 450 927 Borehole 13.000"
E12 450 1250 Borehole 13.000"
Group (String) – Wellhead
E08 10 13 Casing Spool
Group (String) – Surface Casing
E08 12 449 K-55 18.625 OD/ 96.50# 17.655 ID 17.468 Drift
E08 449 450 Casing Shoe 18.625 OD
E09 12 450 Cement behind casing
Group (String) – Intermediate Casing
E12 12 13 Casing Hanger 11.750 OD
E12 13 1249 K-55 11.750 OD/ 79.00# 10.438 ID 10.282 Drift
E12 1249 1250 Casing Shoe 11.750 OD
E13 12 1250 Cement behind casing
E14 E16 1238 1242 Bridge Plug
Well CM Ledger gets added to cumulatively
Downhole Components added, updated or removed by Events

Figure 3. Example well cumulative history.

Figure 3 shows the data for the first few days of a well being drilled and completed. The events are shown as sequentially numbered and this ledger continues to grow through the life of the well. The “Wellbore Equipment” shows the physical installation of the completion. This installation includes the wellbore itself (i.e., the hole drilled in the earth) and the hardware (casing, tubing, etc.) installed in it. Note that each item of equipment contains the ID (the “Exx” in the figure) of the event that installed it, allowing the details (including date and time) of each piece’s installation and removal to be discovered. When an element of equipment is removed, a second ID is added, which has the event that resulted in its removal.

For example, the last element of equipment is a bridge plug used to pressure test the casing; it was installed by event E14 and then later removed by event E16. The pressure test, E15, is between the installation and drilling out. The equipment entry shows both install and remove events at the time of the day 4 cumulative history.

Note also that the borehole equipment entries for each day get “removed” and replaced by a new deeper borehole when appropriate during the drilling phase, with the day’s drilling report being the event that triggers this new equipment entry. This can be seen by looking at, e.g., the second borehole size running on from 450 depth units, which is drilled to 798 by event E10 (“drilling day 2”). Then this borehole data-object is removed by event E11 (“drilling day 3”) and replaced by a deeper borehole to 927, and finally is established by E12 as drilled to 1250.

The Snapshot use case can be thought of as a query on a cumulative history at a given instant in time. The Change Log use case can be thought of as a query on a cumulative history between two instants in time giving the “delta” events between those times.

2.1.4 Jobs, Events, and Service Company Data

2.1.4.1 Job vs. Event

The notion of a *job* is an important one when considering a well completion; it is through a series of jobs that the completion is actually built, modified, or removed.

However, exactly what constitutes or defines a *job* can vary widely from company to company. Figure 4 shows several examples of common jobs—from both the operator/producer and service company perspectives—related to wellbore completions.

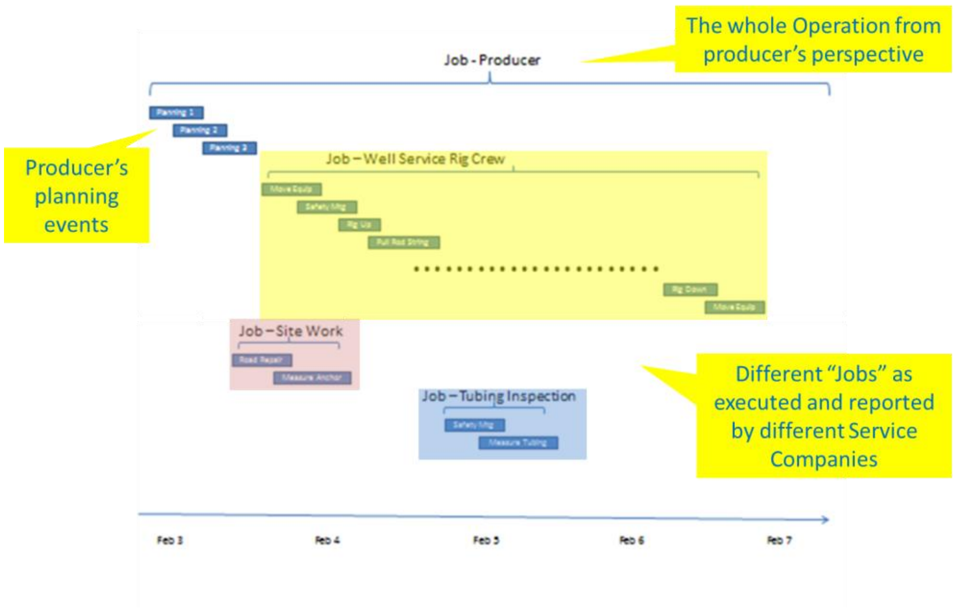


Figure 4. Example well services jobs from both the producer and service company perspective.

The term *job* can be used to refer to everything from the entire scope of work required to install the completion (from the operator's perspective), to one single task of that overall job, for example, perforating the well. There are typically jobs within jobs, which form a nested or hierarchical relationship.

To avoid the varying definitions of the common industry term *job*, the Completion data-object instead uses the concept of an *event*.

2.1.4.2 Event: Definition

An *Event* used in the context of the Completion data-object is any action or scope of work associated with the work on completion. Events are transferred in the Well CM Ledger data-object (see Section 2.2.2, page 19).

Note that Events may include activities that do NOT result in a physical change to the wellbore equipment, for example, reports, inspections, and meetings. However, the Events of primary interest for the Completion data-object are those that add to, modify, or remove components from a completion.

To accommodate the nested or hierarchical nature of the work, Events may be nested within Events, by referring to a Parent Event. Thus, an Event such as "complete the well" may have no Parent, but subsequent Events which were part of the completion operation may show the "complete the well" Event as their parent.

Any WITSML data-object can be referenced in an Event, in addition to the downhole component data-objects, as explained below. In addition, specific "Event Extension" data-objects can be referenced, where these are specific to the kind of Event, e.g., the stimJob data-object for a stimulation event. These data-objects supply the operational details of an Event to supplement the changes to the completion itself transferred using the Completion data-object. In the future, further well services schemas will be added to the “ML” family, enabling the full operational details of an increasing proportion of the well’s history to be transferred between software applications and databases.

2.1.4.3 Service Company Data

The Completion data-object includes the ability for service companies to submit data records to their operator clients corresponding to individual jobs (events) that they perform. Figure 4 shows an example workflow for this process. Within the overall job/event or complete operation from the well producer’s (operator’s) perspective, there may be multiple service providers, each of whom carries out some of the stages of work. These are shown as colored bands in the figure. The concept is that each of these providers may supply Change Log type data-objects containing the events and wellbore equipment elements for which they were responsible.

2.2 Data-Object Organization

The table here lists and describes the top-level data-objects that define the Completion data-object. Details of each are explained further in this section. Figure 5, (page 15) shows the organizational structure and relationships between the key components that make up the Completion data-object. The text throughout this section explains details about the data-objects, their purpose, and use.

Because the Completion data-object is part of the WITSML specification, the same data model concepts (e.g., about parent well and wellbore, nested strings, etc.) are used.

Data-object	Parent Object	Description
Downhole Component (downholeComponent)	Well	Contains all the wellbore equipment in a well. Components are identified as to which wellbore they are in. This data-object covers the openhole boreholes, the strings of equipment, the components in each string and the details of these components, and the connections from these elements to the reservoir.
Well CM Ledger (wellCMLedger)	Well and Wellbore	This name is short for Well Construction and Maintenance Ledger. Contains details of events (which were defined in Section 2.1.4.1, page 13). Each event also has a pointer to elements of downhole components that the event changes. It can reference UID well and UID wellbore for each event, making this data-object able to represent events on multiple wells.
Well Completion (wellCompletion)	Well	The Well Completion data-object represents a “flow” or “stream” from the well (e.g., from a wellhead port) that is associated with a set of Wellbore Completions. When there is more than one such Wellbore Completion, the flows from them commingle in the well (the Wellbore Completions may be located in multiple wellbores). The Well Completion represents this commingled flow. It can be associated with business concept such as field, rights, etc.
Wellbore Completion	Well, Wellbore and Well Completion	Each Wellbore Completion represents a flowing connection between wellbore and reservoir. It contains Contact Intervals, which reference the *physical* aspects of these connections detailed in downholeComponents.

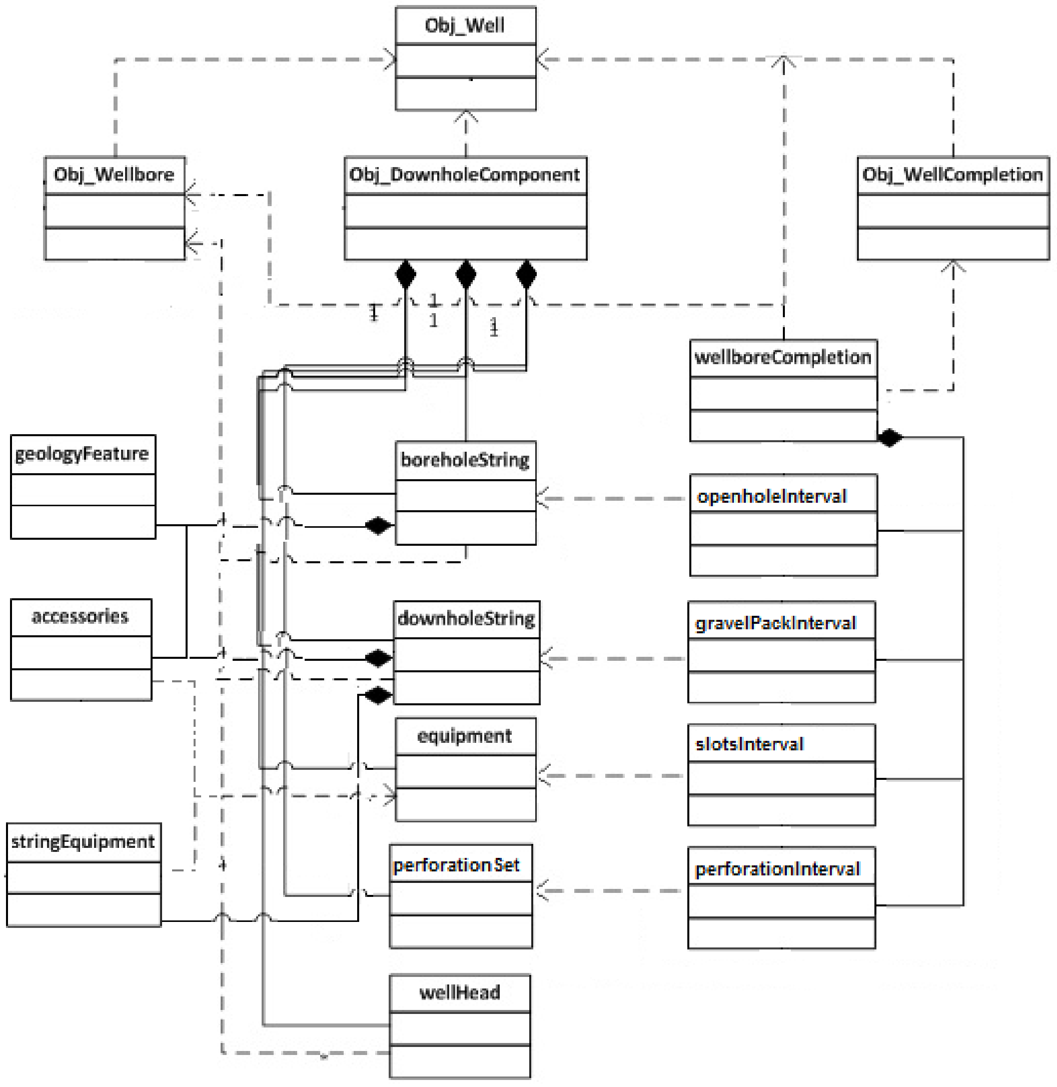


Figure 5. Well, wellbore, downholeComponent and wellCompletion relationship diagram. Text throughout this section explains each of these data-objects.

2.2.1 Downhole Component

The downholeComponent data-object contains all information about the physical completion, all of the well equipment data. The major elements of this information are shown in Figure 6.

Per the standard WITSML API, there is a plural data-object, downholeComponents, which can hold multiple downholeComponent data-objects.

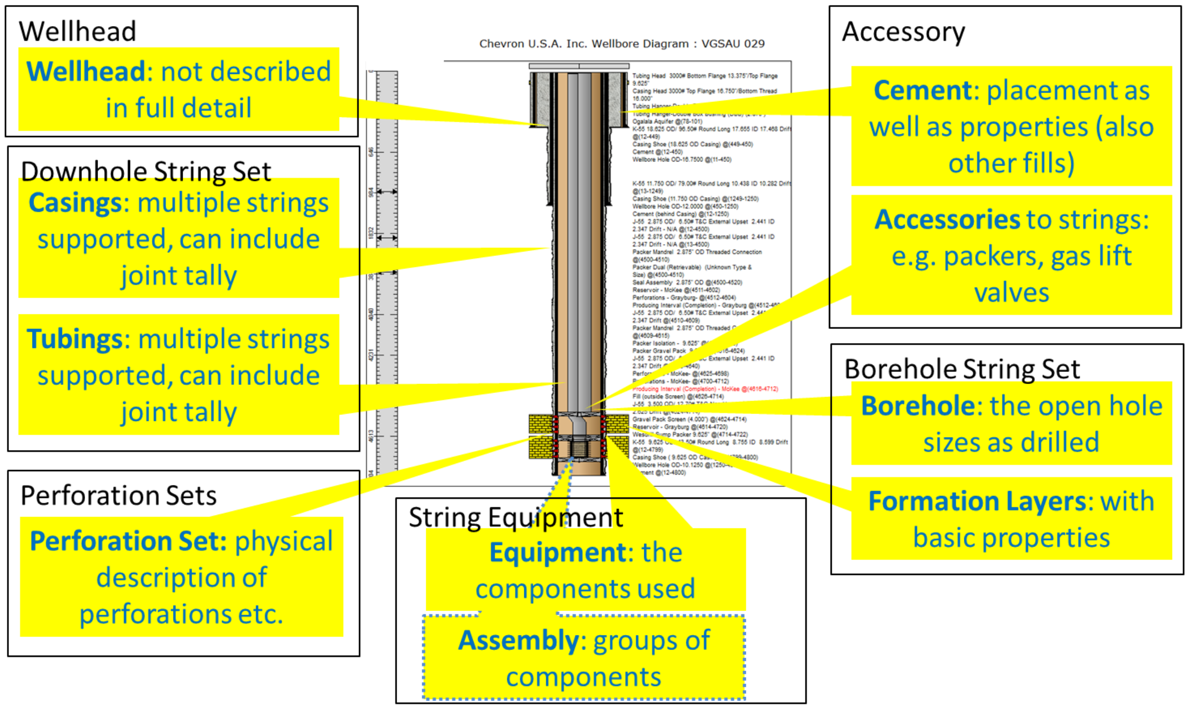

Wellhead
Wellhead: not described in full detail
Downhole String Set
Casings: multiple strings supported, can include joint tally
Tubings: multiple strings supported, can include joint tally
Perforation Sets
Perforation Set: physical description of perforations etc.
Chevron U.S.A. Inc. Wellbore Diagram : VGSAU 029
String Equipment
Equipment: the components used
Assembly: groups of components
Accessory
Cement: placement as well as properties (also other fills)
Accessories to strings: e.g. packers, gas lift valves
Borehole String Set
Borehole: the open hole sizes as drilled
Formation Layers: with basic properties

Figure 6. Scope of downHole component. These elements are described further in the text below.

Each of the constituent elements of the downholeComponents (e.g., each item of equipment in a string) has a date element called eventInfo. This eventInfo element represents either the beginning or ending event for the item, and it contains a timestamp and a pointer reference to the wellCMLedger entry corresponding to this event (thereby providing the link referred to above between event and equipment).

2.2.1.1 Wellhead

In this version of the data model, the wellhead is treated as a kind of *downholeString* (for details, see Section 2.2.1.3, page 17). It contains some data elements but is not intended to be comprehensive. A more detailed wellhead model could be developed in future releases.

2.2.1.2 Borehole String Set

This section of the schema is used to describe the borehole itself (the as-drilled hole through the earth) and some of the earth-related information, including the completed (flowing) intervals, the layers drilled through, etc.

Element	Description
Borehole String	Used to define the drilled hole that corresponds to the wellbore. A collection of contiguous and non-overlapping borehole sections is allowed. Each section has depth range, diameter, and kind.
Geology Feature	Aquifer or reservoir features are supported, and a range of parameters can be reported for each.
Accessory	An accessory is equipment or other data-object associated with but not part of the drilled open hole being described. For example: cement, junk in hole.

2.2.1.3 Downhole String Set

This section of the schema is used to describe all strings in the completion, including casing, tubing, and rod strings.

A completion may have multiple sets of strings which may be nested each inside another, or run in parallel as in dual string completions; all strings are contained in a parent wellbore. Each string is composed of equipment, and may also contain accessories and/or assemblies.

Element	Description
Downhole String	A collection of strings is allowed. Strings: • Are considered to be contained in parent wellbores. • Each has a type (casing, tubing, or rod). • Contain a set of string equipment and can contain string accessories and assemblies (described below in this table). • Have install and removal dates.
String Equipment	Each string is created by mechanically coupling individual pieces of equipment, for example, tubing, gravel pack screens, etc. The stringEquipment element is used to define the set of equipment that makes up a particular string. Each item of equipment in the String Equipment Set includes this information: • Type defines the type of equipment from an enumerated list. • Depth defines the MD where the item of equipment is located in the string. • connectionNext refers to the piece of equipment in the string to which the current equipment is connected. • Run history identifies if the equipment has been previously used. To avoid duplication of data, common equipment properties (such as diameter, model, materials, and drift) are stored in the Equipment Set. Each item of String Equipment contains a reference (equipmentRefUID) to the corresponding kind of component in the Equipment Set. Hence, stringEquipment covers “usage properties” and equipmentSet covers “common properties”. For more information, see Section 2.2.1.4, page 18.
Accessory	A string can contain one or more accessories, which refers to equipment or objects that are related to or part of the string but do NOT mechanically couple the string together, for example, a gas lift mandrel. “Fills” in the spaces between the strings are also accessories. For example: cement, or gravel or “accidental” objects, such as sand fill or dropped fish. Accessory components use the same data-object as String Equipment. They use data elements in the data-object to define where the Accessory is relative to the String and String Equipment with which they are associated: • A Boolean “outside string” flag to indicate if the Accessory is outside its parent string. • Two reference elements, “inside component” and “outside component”, point to the component either inside the Accessory or outside it, to which it is attached. To describe a cemented casing string: • String type = casing o String equipment = casing pipe, casing shoe, etc. o ID = 1 then • Accessory = cement • outsideString = true • outsideComponent = [using *refContainer*, ID = 1]

Element	Description
Assembly	The concept of an Assembly is used to refer to a collection of equipment as one "thing". It is expected that an Assembly will be used where the level of detail required varies depending on users' needs and/or the software applications' capability to process and store those details. Its use is implementation-specific. For example, one piece of software (or user) may only care about that there is an assembly representing "an ESP". In contrast, another piece of software or user may want to know about the detailed components of the ESP. In this case, the data that could be passed so the ESP components are all contained in the Assembly called "ESP" and the first user would not need further information, while the second can look at the Assembly element and get a list and structure of the components inside. Within an Assembly element, the sub-components available (in cs_stringEquipment) are the same as available in the String Equipment element itself. Assembly can be used recursively.

2.2.1.4 Equipment Set

This section of the schema contains a collection of kinds of equipment that comprise the string. Each kind of equipment in the set has a type (what it is) and attributes common across all instances of that type of equipment. The String Equipment then references these common attributes (as described above in Section 2.2.1.3, page 17).

Element	Description
Type	An enum list (equipmentType) which is listed in the cs_equipmentCatalog.xsd schema (For a printed list of equipment types available when this document was published, see Appendix A. Equipment Types and Properties, page 60.).
Equipment Property Group	Common Properties about the equipment, for example, the manufacturer, diameters, materials, etc. These properties apply to all Equipment Types and can all be seen in the grp_equipmentProperty.xsd schema.
Property	Extended (or "custom") properties for these Equipment Types. These properties are those which are unique to the particular Equipment Type. These are listed in a loader file EquipmentDict.xml, which is found in the *ancillary* folder of the schema set. They can also be seen in a spreadsheet EquipmentType.xlsx, in the same folder. For more details on equipment properties and the loader file, see Section 4.5, page 57. It is possible to extend the lists of properties by editing the loader file.

The following simplified example shows how the String Equipment and Equipment Set can be used.

Suppose we have the following equipment at the specified depths:

1. Tubing 3.5 in. 0–2,000 m (meters)
2. Packer 2000–2005 m
3. Tubing 3.5 in. 2005–3,000 m

Conceptually this is represented in the data-object as follows:

- Downhole Component
  - Downhole String name = "Tubing"
    - String Equipment Set
      1. Tubing 3.5" 0-2,000 meters, install date = abc, etc…equipmentRefUID= A
      2. Packer 2000-2005 meters, install date = abc, etc…equipmentRefUID= B
      3. Tubing 3.5" 2005-3,000 meters, install date = abc, etc…equipmentRefUID= A

- Equipment Set
  - ID= A, Type = tubing, OD = 3.5”, ID = 3.0”, manufacturer = abc, etc.
  - ID= B, Type = packer, OD = 4.9”, seal kind = xyz, etc.

The efficiency of this approach is self-evident.

2.2.1.5 Perforation Sets

This section contains information about perforations in the downhole components. Each set has a reference to its related downhole string, and borehole string, and then gives the perforation details. See also Section 2.2.5 for how this fits into the flow path description.

2.2.2 Well Construction and Maintenance Ledger (Event Ledger)

The "event ledger" concept is this: each time an activity associated with a well happens (an "Event" as defined in Section 2.1.4.2, page 13) it is recorded in the ledger, which is named the wellCMLedgers (Well Construction and Maintenance Ledgers).

Recording this information tracks the history of all events and makes it possible to derive the complete history of the completion, including the use cases described in Chapter 2. A single Event entry fills an instance of the data-object wellCMLedger and can contain this information:

Information	Description
Identities	Well and wellbore name, event name, etc
Event details	More detailed information about the operation of an event, e.g. depths, type, times, duration, etc.
Completion equipment reference	References to the Downhole String (see Section 2.2.1.3, page 17), and the String Equipment component being worked on within this string.
Operational and management information	Business associate, work order number, cost, rig ID, flags to indicate if the operation was planned or unplanned etc.
Event Extension	Contains a reference to a WITSML data-object containing the job data. This is done using a choice of extension types available in the cs_eventExtension element. This is an enumerated list of existing event data-objects defined in WITSML, for example, for cementing and fracture/stimulation treatments. The wellCMLledger entry has references to both the equipment (e.g., for a cement job, the cement component, which would be an Accessory) and if the Event Extension is used, to the job (the cementJob data-object). This enables the audit of well history to be as comprehensive as possible. There is not yet a comprehensive family of such “event/job data-objects”. For example, there is no “gravel pack job” data-object (which would include pumping schedules of gravel and completion fluids, etc.). Such data-objects could be developed over time and with them the ability to reference more and more well history in full detail.
Participant	A reference to any WITSML data-object associated with the Event. It is more generic than the Event Extension which is “xxxJob” specific. Participant is a reference using the WITSML Member data-object element. This enables e.g., a WITSML log to be referenced as relevant to the Event concerned.

2.2.3 Well Completion

This is a top-level object which works with other objects to report flow paths from reservoir to surface. See Section 2.2.5.

2.2.4 Wellbore Completion

This is a top-level object which works with other objects to report flow paths from reservoir to surface. See Section 2.2.5.

2.2.5 Flow Paths from Reservoir to Wellhead

Two top level objects are used in the flow path logic:

- Well Completion
- Wellbore Completion

These work together and with other sub-objects in a rather involved way to be able to report flow from defined places along the wellbore to defined surface streams. The following table summarizes how this is done. See also Figure 7, page 21.

Element	**Parent**	**Description**
Well Completion	Well	Occurs as a top level data-object in the schema. Represents a “flow” or “stream” in the well. The data-object also contains business information about this flow, such as field ID, ownership rights, etc. Referenced by the Wellbore Completion. Multiple Wellbore Completions can be associated with a Well Completion. In this way, a commingled set of connections with the reservoir can be associated with a flow within the well. Generally, a Well Completion will be associated with a single “Wellhead stream”, i.e., flow from a port in the wellhead. However, this is not always the case. For example, in cases where liquid and gas are separated downhole, the liquid may flow up the tubing and the gas up the casing annulus. In this case, there would be two “wellhead streams” from two ports (tubing and casing annulus), but both would be associated with the same Well Completion since they both come from the same set of commingled Wellbore Completions. It is expected that the wellhead streams will be reported in the PRODML Product Volume data-object, where a link to the Well Completion can in future be added.
Wellbore Completion	Wellbore	Occurs as a top level data-object in the schema. Each individual wellbore completion data-object represents a completion (i.e., open to flow) interval along a wellbore. Meaning “this section of wellbore is open to flow”. Has data for depth, API numbering, type of fluid, etc. Contains a reference to a Well Completion data-object (which is the flow stream from the wellhead), see above. Contains a Contact Interval Set, which represents the physical connection(s) to the reservoir.
Contact Interval Set	Wellbore Completion	Occurs in the Wellbore Completion data-object. Contains one or more “xxxInterval” objects, each representing the details of a single physical connection between well and reservoir, e.g., the perforation details, depth, reservoir connected. Meaning: this is the physical nature of a connection from reservoir to wellbore. The kinds of interval allowed are: • Gravel Pack (refs downhole string) • Open Hole (refs borehole string) • Perforated (refs perforation set) • Slotted (refs string equipment) Each interval has an ID, details about the interval itself, including history, and a reference to the data object giving full details (the kind of equipment is shown in parentheses in the list).

Element	Parent	Description
Perforation Set	Downhole Components	This section contains information about perforations in the downhole components. Each set has a reference to its related downhole string, and borehole string, and then gives the perforation details. It will be referred to from the Perforated type of Contact Interval.
String Equipment	Downhole Components	This section contains the individual items of equipment deployed in the downhole components. A given item of equipment can have a reference to a Perforation Set, meaning "this item of equipment contains the perforations of *thi*s Perforation Set".

Note that flow is *not* modeled through the components making up the completion (for example, through sliding sleeves, specific tubing strings, etc.). Such a full "flow path" may be added as a later development of the data standard.

For code examples and more details of the perforations and how these link to "completions" representing flows, see Section 4.1, page 40.

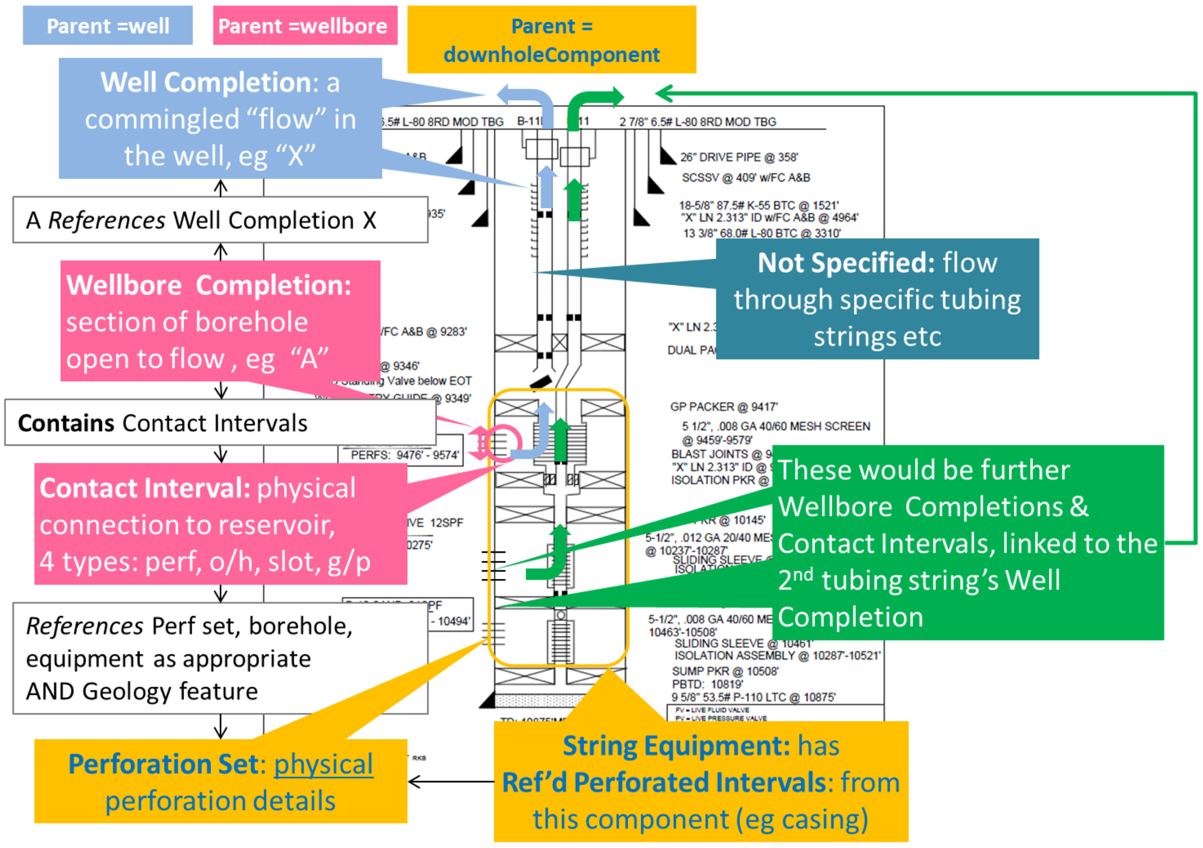

Parent =well
Parent =wellbore
Parent = downholeComponent
Well Completion: a commingled "flow" in the well, eg "X"
A References Well Completion X
Wellbore Completion: section of borehole open to flow , eg "A"
Contains Contact Intervals
Contact Interval: physical connection to reservoir, 4 types: perf, o/h, slot, g/p
References Perf set, borehole, equipment as appropriate AND Geology feature
Perforation Set: physical perforation details
Not Specified: flow through specific tubing strings etc
These would be further Wellbore Completions & Contact Intervals, linked to the 2nd tubing string's Well Completion
String Equipment: has Ref'd Perforated Intervals: from this component (eg casing)
2 7/8" 6.5# L-80 8RD MOD TBG
26" DRIVE PIPE @ 358'
SCSSV @ 409' w/FC A&B
18-5/8" 87.5# K-55 BTC @ 1521'
13 3/8" 68.0# L-80 BTC @ 3310'
GP PACKER @ 9417'
5 1/2", .008 GA 40/60 MESH SCREEN @ 9459'-9579'
PERFS: 9476' - 9574'
SLIDING SLEEVE @ 10461'
ISOLATION ASSEMBLY @ 10287'-10521'
SUMP PKR @ 10508'
PBTD: 10819'
9 5/8" 53.5# P-110 LTC @ 10875'

Figure 7. Scope of flow representation, showing elements and linkages used.

2.3 Role of "Well" and "Wellbore" Concepts as Applied in Completion

Most WITSML data-objects are children of the wellbore data-object. This organization is logical given that drilling operations (the subject of WITSML) are happening in only one wellbore within a well, at any given time. However, in a completed well with multiple wellbores, installed equipment spans all wellbores. For this reason, in the Completion data data-object, the parent data-object is the well, although many of the child data-objects of completion fit under the wellbore concept.

Figure 8 shows the main associations with well (blue boxes) and with wellbore (pink boxes).

- The wellhead is associated with well.
- The whole set of completion equipment, downhole components, is also associated with well.
- Individual strings and the equipment in them, and boreholes are associated with wellbore.
- Concerning flow, wellbore completions and contact intervals are associated with wellbore.
- However the well completion data-object, representing flow out of the wellhead, is associated with well because it may be a commingled flow from multiple wellbores.
- The Well Construction and Maintenance Ledger plural data-object (wellCMLedgers) can associate with multiple wells. Each entry in it (wellCMLedger) is associated with well and, if appropriate, also with specific *wellbore*.

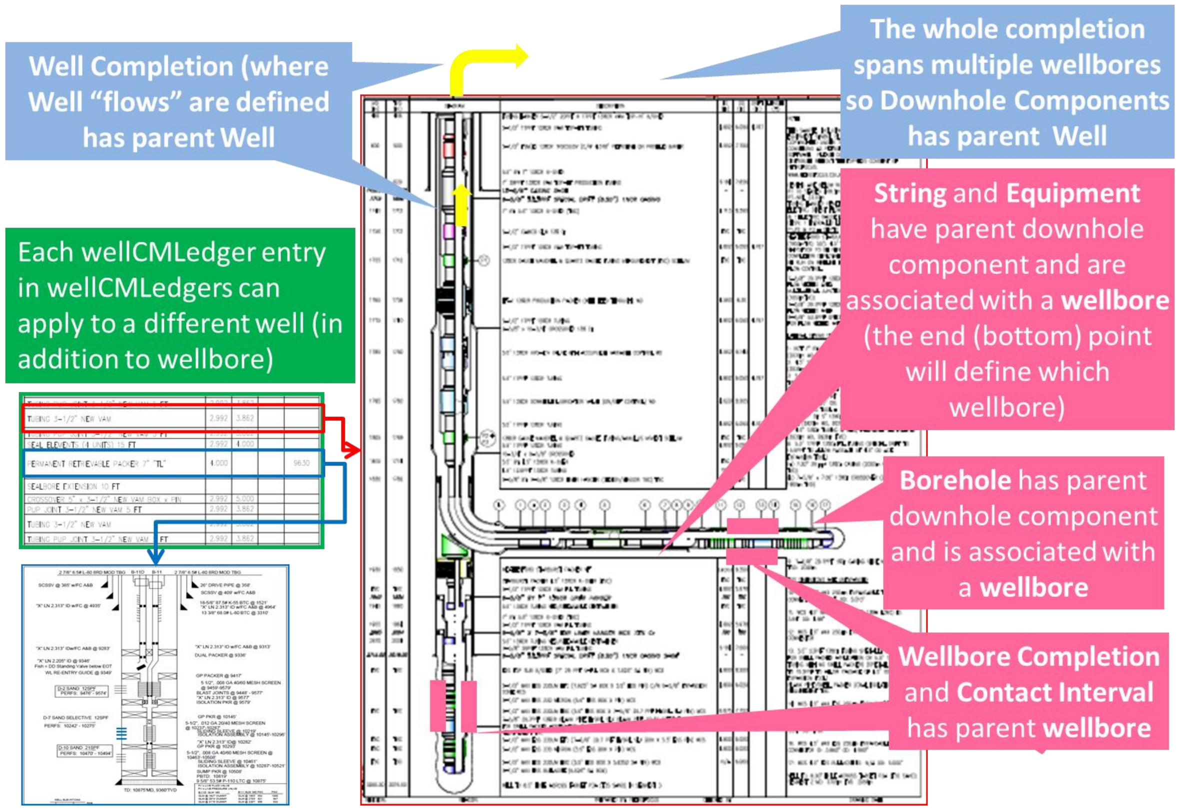

Well Completion (where Well "flows" are defined has parent Well
The whole completion spans multiple wellbores so Downhole Components has parent Well
Each wellCMLedger entry in wellCMLedgers can apply to a different well (in addition to wellbore)
String and Equipment have parent downhole component and are associated with a wellbore (the end (bottom) point will define which wellbore)
Borehole has parent downhole component and is associated with a wellbore
Wellbore Completion and Contact Interval has parent wellbore
TUBING 3-1/2" NEW VAM
SEAL ELEMENTS (4 UNITS) 15 FT
PERMANENT RETRIEVABLE PACKER 7" "TL"
SEALBORE EXTENSION 10 FT
CROSSOVER 4" x 3-1/2" NEW VAM BOX x PIN
PUP JOINT 3-1/2" NEW VAM 5 FT
TUBING 3-1/2" NEW VAM
TUBING PUP JOINT 3-1/2" NEW VAM 8 FT

Figure 8. Completion data-object in the context of well (blue boxes) and wellbore (pink boxes).

3 Code Examples: Use Cases

This chapter provides XML code examples for implementing the Completion data-object for each of the use cases described in Chapter 2, page 10.

The **example data used** in this chapter is available in the xml_examples folder of the Completions data-object package that you download from the Energistics website.

Note that, to illustrate specific points, the example code snippets may be abbreviations of or variations of the full example cases provided.

A second example of how to use the Completion data-object in the context of other Energistics data-objects is provided on the Energistics website, along with instructions on how to correctly set up the folder structure to combine the domains into referenceable objects. This addendum is accompanied by a "history" file detailing the events that transpired from post-drilling to the installation of various pieces of equipment that led to the wellbore's completion.

3.1 Use Case 1: Snapshot

The Snapshot use case refers to exchanging data that represents the well completion at a given instant in time. For the conceptual model of the Snapshot use case, see Section 2.1.1, page 10.

Figure 9 is an example snapshot of a vertical well that was drilled and completed with 2 naturally flowing completed intervals, in November 1990.

The Snapshot primarily uses the downholeComponent data-object to transfer the description of the hardware at an instant in time. The example XML in this section shows the use of downholeComponent, which, for ease of following the example, have been ordered slightly differently than they are in the schema.

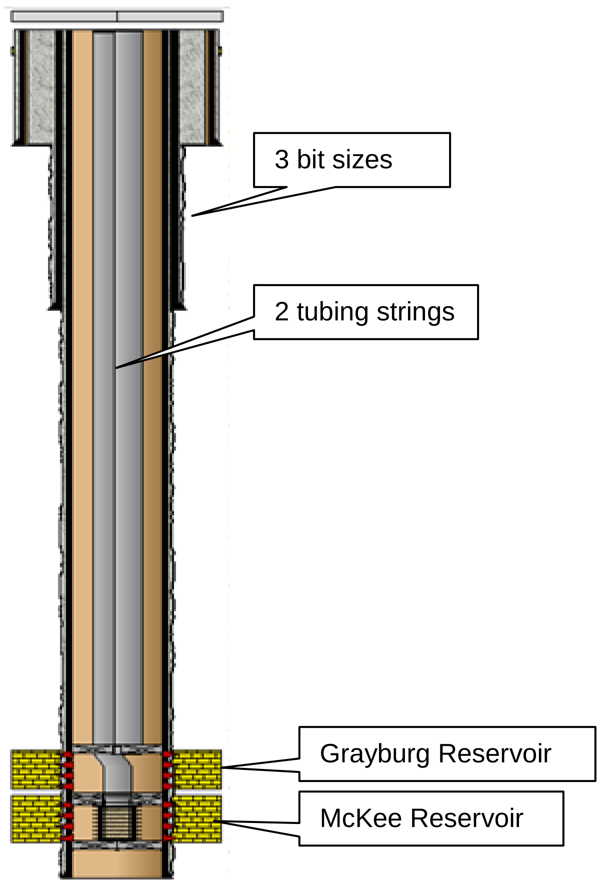


Figure 9. Example wellbore for Use Case 1 Snapshot.

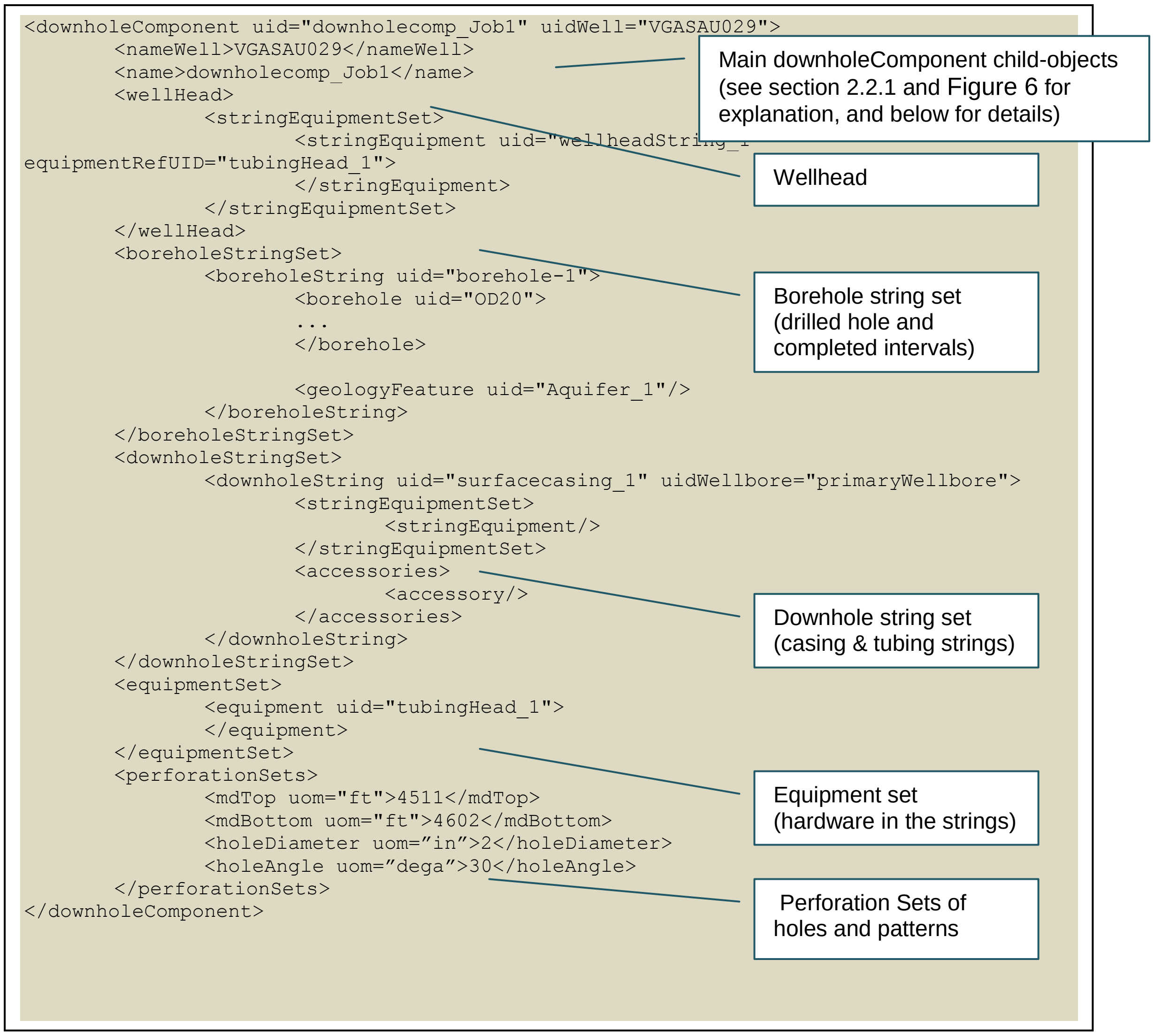
3.1.1 downholeComponent as Container Data-object

A snapshot data transfer has a downholeComponent data-object that contains all of the other data-objects being transferred. .

```
<downholeComponent uid="downholecomp_Job1" uidWell="VGASAU029">
             <nameWell>VGASAU029</nameWell>
             <name>downholecomp_Job1</name>
             …
             < commonData>
                    <dTimCreation>2012-11-16T20:40
                    <dTimLastChange>2012-11-
16T20:40:50.4513907Z</dTimLastChange>
             </commonData>
</downholeComponent>
```

Meta data for the downholeComponent container includes:

- UID
- well name and UID
- downholeComponent name
- Data common to all data-objects
- And if, used custom data.



```
<downholeComponent uid="downholecomp_Job1" uidWell="VGASAU029">
        <nameWell>VGASAU029</nameWell>
        <name>downholecomp_Job1</name>
        <wellHead>
                <stringEquipmentSet>
                        <stringEquipment uid="wellheadString_1"
equipmentRefUID="tubingHead_1">
                        </stringEquipment>
                </stringEquipmentSet>
        </wellHead>
        <boreholeStringSet>
                <boreholeString uid="borehole-1">
                        <borehole uid="OD20">
                        ...
                        </borehole>

                        <geologyFeature uid="Aquifer_1"/>
                </boreholeString>
        </boreholeStringSet>
        <downholeStringSet>
                <downholeString uid="surfacecasing_1" uidWellbore="primaryWellbore">
                        <stringEquipmentSet>
                                <stringEquipment/>
                        </stringEquipmentSet>
                        <accessories>
                                <accessory/>
                        </accessories>
                </downholeString>
        </downholeStringSet>
        <equipmentSet>
                <equipment uid="tubingHead_1">
                </equipment>
        </equipmentSet>
        <perforationSets>
                <mdTop uom="ft">4511</mdTop>
                <mdBottom uom="ft">4602</mdBottom>
                <holeDiameter uom=”in”>2</holeDiameter>
                <holeAngle uom=”dega”>30</holeAngle>
        </perforationSets>
</downholeComponent>
```

Main downholeComponent child-objects (see section 2.2.1 and Figure 6 for explanation, and below for details)
Wellhead
Borehole string set (drilled hole and completed intervals)
Downhole string set (casing & tubing strings)
Equipment set (hardware in the strings)
Perforation Sets of holes and patterns

3.1.2 Implementation of Equipment within the downholeComponent Child Objects

Both the wellHead and downholeString child objects, within the downholeComponent, use the same data-object, which is downholeString. This data-object contains stringEquipment items. Each item of stringEquipment has a reference to an item of equipment in the equipmentSet data-object. See Sections 2.2.1.3 (page 17) and 2.2.1.4 (page 18). This construct is common to both wellHead and downholeString and is illustrated in the following example.

```
<downholeComponent uid="downholecomp_Job1" uidWell="VGASAU029">
        <downholeStringSet>
                <downholeString uid="surfacecasing_1" uidWellbore="primaryWellbore">
                        <stringEquipmentSet>
                                <stringEquipment uid="tubingstring_21"
equipmentRefUID="tubingOD2.875_1">
                                        <equipmentName>Tubing_21</equipmentName>
                                        <equipmentType>Tubing</equipmentType>
                                        <name>tubing_OD_2.875</name>
                                        <count>143</count>
                                        <length uom="ft">4486</length>
                                        <mdTop uom="ft">12</mdTop>
                                        <mdBottom uom="ft">4500</mdBottom>
                                        <tvdTop uom="ft">12</tvdTop>
                                        <tvdBottom uom="ft">4500</tvdBottom>
                        </stringEquipmentSet>
                </downholeString>
        </downholeStringSet>
        <equipmentSet>
                <equipment uid="tubingOD2.875_1">
                        <equipmentName> tubing</equipmentName>
                        <equipmentType>Tubing</equipmentType>
                        <material>Steel</material>
                        <grade>J-55</grade>
                        <unitWeight uom="lbm">6.5</unitWeight>
                        <OD uom="in">2.875</OD>
                        <ID uom="in">2.441</ID>
                        <drift uom="dega">2.347</drift>
                </equipment>
        </equipmentSet>
</downholeComponent>
```

The downhole string is an installed string, containing string equipment, e.g. a length of tubing

The equipment set contains types of equipment, e.g. a type of tubing (like a catalog)

Each type of equipment has a uid, and each use of that type in the string's equipment set has a ref pointing to that type

3.1.3 Use of equipmentEventHistory

Every item of equipment in the downholeComponent data-object can contain an equipmentEventHistory. The event history can comprise a beginEvent and an endEvent. Each event has an eventDate, and an eventRefID, which is a pointer to the wellCMLedger entry having this value as its UID.

```
<stringEquipmentSet>
        <!-- surfacecasing's casing and casingliner -->
        <stringEquipment uid="surfacecasingString_1"
equipmentRefUID="surfacecasing_casingliner_1">
                <equipmentType>Casing/Casing Liner</equipmentType>
                <name>K-55 18.625 OD/ 96.50# Round Long 17.655 ID 17.468 Drift</name>
                <equipmentEventHistory>
                        <beginEvent>
                                <eventRefID>E07</eventRefID>
                                <eventDate>1990-11-11T11:00:00Z</eventDate>
                        </beginEvent>
                </equipmentEventHistory>
        </stringEquipment>
</stringEquipmentSet>

<wellCMLedger uid="E07" uidWell="VGASAU029" uidWellbore="primaryWellbore">
        <nameWellbore>primaryWellbore</nameWellbore>
        <parentEventID>E00</parentEventID>
        <dTimStart>1990-11-11T11:00:00Z</dTimStart>
        <dTimEnd>1990-11-11T17:00:00Z</dTimEnd>
        <duration uom="h">6</duration>
        <mdTop uom="ft">12</mdTop>
        <mdBottom uom="ft">200</mdBottom>
</wellCMLedger>
```

Each item of equipment has an event history element with begin/end events, with dates and ref to the well CM ledger entry corresponding.

The well CM ledger entry has the UID referenced, identifying this specific event (and giving more details)

For simplicity, in the rest of the following examples, the equipmentEventHistory element is not shown. Whether or not this information is required is implementation-specific.

3.1.4 boreholeStringSet

The boreholeStringSet is used to describe the borehole itself (the as-drilled hole through the earth) and some of the earth-related information, including the completed intervals, the layers drilled through, etc.

A boreholeStringSet contains a collection of boreholeString data-objects. Each boreholeString corresponds to a single borehole (e.g. the parent wellbore or a sidetrack borehole).

Each boreholeString contains a collection of borehole data-objects. Each borehole data-object:

- represents a section of drilled borehole.
- corresponds to a section of borehole drilled with single bit size that was used for that section of the boreholeString.

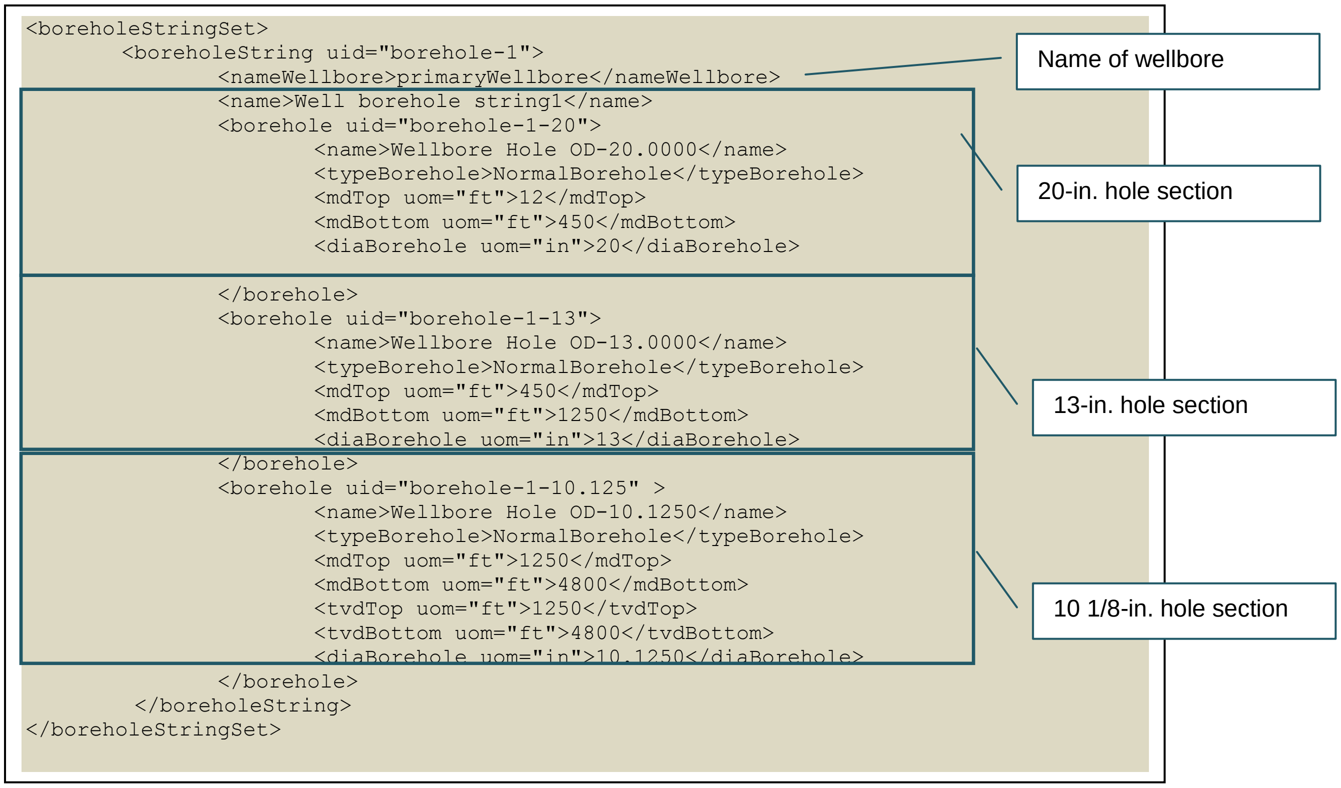
In addition to borehole data-objects, the boreholeString data-object also contains these other elements:

- geologyFeature. This is a representation of layers of the rock which have been penetrated by this wellbore.
- The boreholeString data-object can also contain accessories (for more information, see Section 2.2.1.3). Accessories of the boreholeString data-object (which represents the drilled open hole) are likely to be fills such as cement or fish.

3.1.4.1 boreholeString

The boreholeString (part of the boreholeStringSet) is a collection of multiple borehole sections. Each section has depth range, diameter, and kind

This example has a wellbore that consists of 3 borehole sections with diameters of 20 in., 13 in., and 10.125 in.



```
<boreholeStringSet>
        <boreholeString uid="borehole-1">
                <nameWellbore>primaryWellbore</nameWellbore>
                <name>Well borehole string1</name>
                <borehole uid="borehole-1-20">
                        <name>Wellbore Hole OD-20.0000</name>
                        <typeBorehole>NormalBorehole</typeBorehole>
                        <mdTop uom="ft">12</mdTop>
                        <mdBottom uom="ft">450</mdBottom>
                        <diaBorehole uom="in">20</diaBorehole>

                </borehole>
                <borehole uid="borehole-1-13">
                        <name>Wellbore Hole OD-13.0000</name>
                        <typeBorehole>NormalBorehole</typeBorehole>
                        <mdTop uom="ft">450</mdTop>
                        <mdBottom uom="ft">1250</mdBottom>
                        <diaBorehole uom="in">13</diaBorehole>
                </borehole>
                <borehole uid="borehole-1-10.125" >
                        <name>Wellbore Hole OD-10.1250</name>
                        <typeBorehole>NormalBorehole</typeBorehole>
                        <mdTop uom="ft">1250</mdTop>
                        <mdBottom uom="ft">4800</mdBottom>
                        <tvdTop uom="ft">1250</tvdTop>
                        <tvdBottom uom="ft">4800</tvdBottom>
                        <diaBorehole uom="in">10.1250</diaBorehole>
                </borehole>
         </boreholeString>
</boreholeStringSet>
```

3.1.4.2 geologyFeature

This data-object is a representation of layers of the rock that have been penetrated by this wellbore.

This example shows one of aquifer kind and two layers of reservoir kind.

```
<boreholeStringSet>
        <boreholeString uid="borehole-1">
                <geologyFeature uid="Aquifer_1">
                        <name>Ogalala</name>
                        <geologyType>Aquifer</geologyType>
                        <mdTop uom="ft">78</mdTop>
                        <mdBottom uom="ft">101</mdBottom>
                </geologyFeature>
                <geologyFeature uid="Reservoir_1">
                        <name>Grayburg_Reservoir</name>
                        <geologyType>Reservoir</geologyType>
                        <mdTop uom="ft">4511</mdTop>
                        <mdBottom uom="ft">4602</mdBottom>
                </geologyFeature>
                <geologyFeature uid="Reservoir_2">
                        <name>McKee_Reservoir</name>
                        <geologyType>Reservoir</geologyType>
                        <mdTop uom="ft">4614</mdTop>
                        <mdBottom uom="ft">4720</mdBottom>
                </geologyFeature>
        </boreholeString>
</boreholeStringSet>
```

3.1.4.3 Accessory

As noted above, accessory data-objects within the boreholeString data-object are likely to be fills of one kind or another. This example shows how the fish would be represented within the borehole.

```
<accessories>
        <accessory uid="FishInWellbore_1">
                <equipmentType>Fish</equipmentType>
                <name>Fish in wellbore</name>
                <mdTop uom="ft">4800</mdTop>
                <mdBottom uom="ft">4805</mdBottom>
                <tvdTop uom="ft">4800</tvdTop>
                <tvdBottom uom="ft">4805</tvdBottom>
                <outsideString>false</outsideString>
        </accessory>
</accessories>
```

outsideString is false because the fish is *inside* this string.

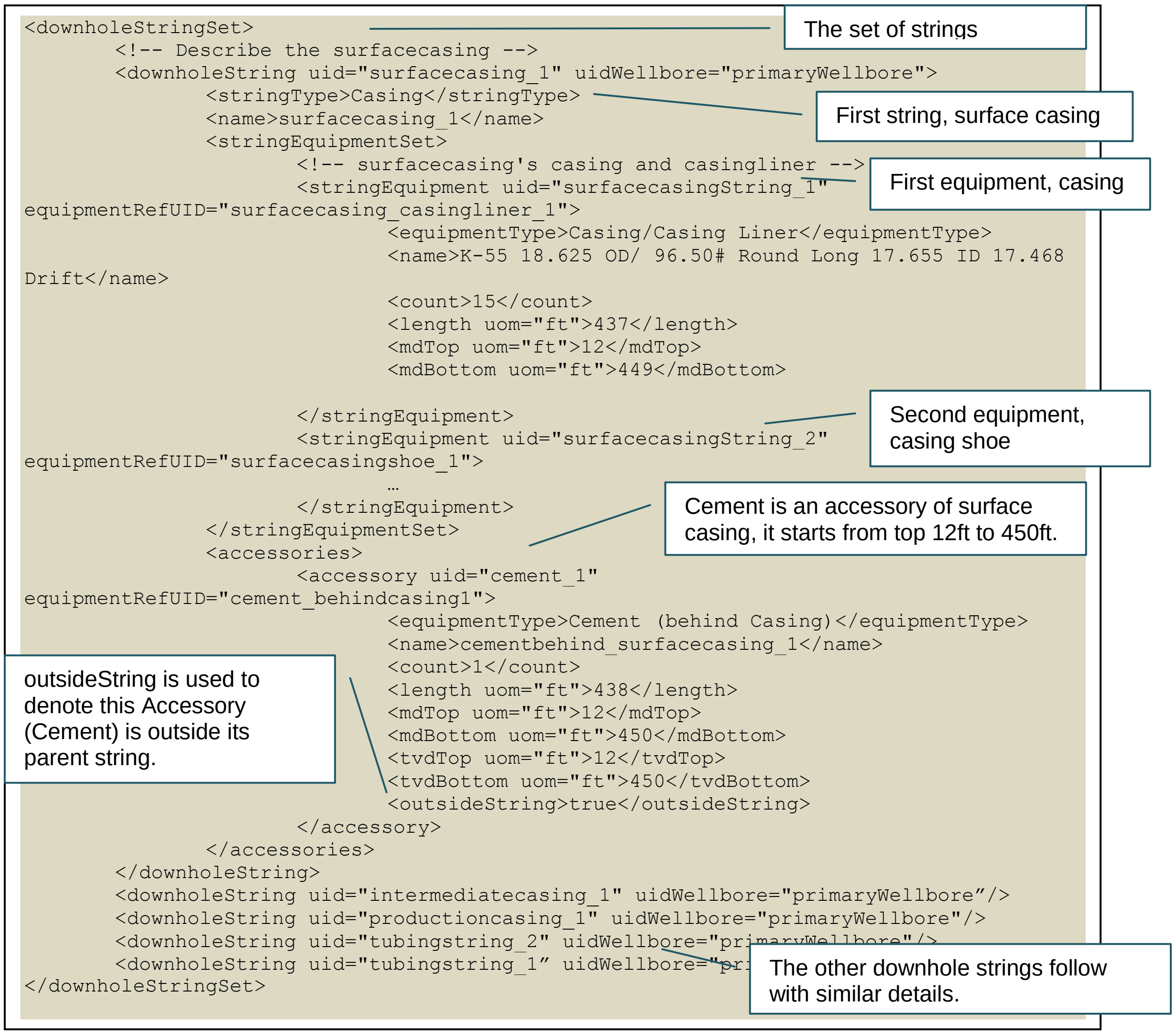
3.1.5 downholeStringSet

The downholeStringSet is used to describe all strings in the completion, including casing, tubing, and rod strings. Each string is composed of equipment, may also be defined as an assembly, and may also contain accessories.

Note that wellHead is a kind of downholeString. The wellHead data-object logically appears earlier in the schema, before the boreholeStringSet, but to make this description more generic, it is described below here, along with the other kinds of downholeStrings.

This following example describes the casing (surface casing, intermediate casing, production casing) and two tubing strings, with just the surface casing shown in detail.

Note that the completion contains packers (as shown in Figure 9). Because these items of equipment can have a number of configurations, they are dealt with separately, see Section 4.3, page 47.



```
<downholeStringSet>
        <!-- Describe the surfacecasing -->
        <downholeString uid="surfacecasing_1" uidWellbore="primaryWellbore">
                <stringType>Casing</stringType>
                <name>surfacecasing_1</name>
                <stringEquipmentSet>
                        <!-- surfacecasing's casing and casingliner -->
                        <stringEquipment uid="surfacecasingString_1"
equipmentRefUID="surfacecasing_casingliner_1">
                                <equipmentType>Casing/Casing Liner</equipmentType>
                                <name>K-55 18.625 OD/ 96.50# Round Long 17.655 ID 17.468
Drift</name>
                                <count>15</count>
                                <length uom="ft">437</length>
                                <mdTop uom="ft">12</mdTop>
                                <mdBottom uom="ft">449</mdBottom>

                        </stringEquipment>
                        <stringEquipment uid="surfacecasingString_2"
equipmentRefUID="surfacecasingshoe_1">
                                …
                        </stringEquipment>
                </stringEquipmentSet>
                <accessories>
                        <accessory uid="cement_1"
equipmentRefUID="cement_behindcasing1">
                                <equipmentType>Cement (behind Casing)</equipmentType>
                                <name>cementbehind_surfacecasing_1</name>
                                <count>1</count>
                                <length uom="ft">438</length>
                                <mdTop uom="ft">12</mdTop>
                                <mdBottom uom="ft">450</mdBottom>
                                <tvdTop uom="ft">12</tvdTop>
                                <tvdBottom uom="ft">450</tvdBottom>
                                <outsideString>true</outsideString>
                        </accessory>
                </accessories>
        </downholeString>
        <downholeString uid="intermediatecasing_1" uidWellbore="primaryWellbore”/>
        <downholeString uid="productioncasing_1" uidWellbore="primaryWellbore"/>
        <downholeString uid="tubingstring_2" uidWellbore="primaryWellbore"/>
        <downholeString uid="tubingstring_1” uidWellbore="pri
</downholeStringSet>
```

The set of strings

First string, surface casing

First equipment, casing

Second equipment, casing shoe

Cement is an accessory of surface casing, it starts from top 12ft to 450ft.

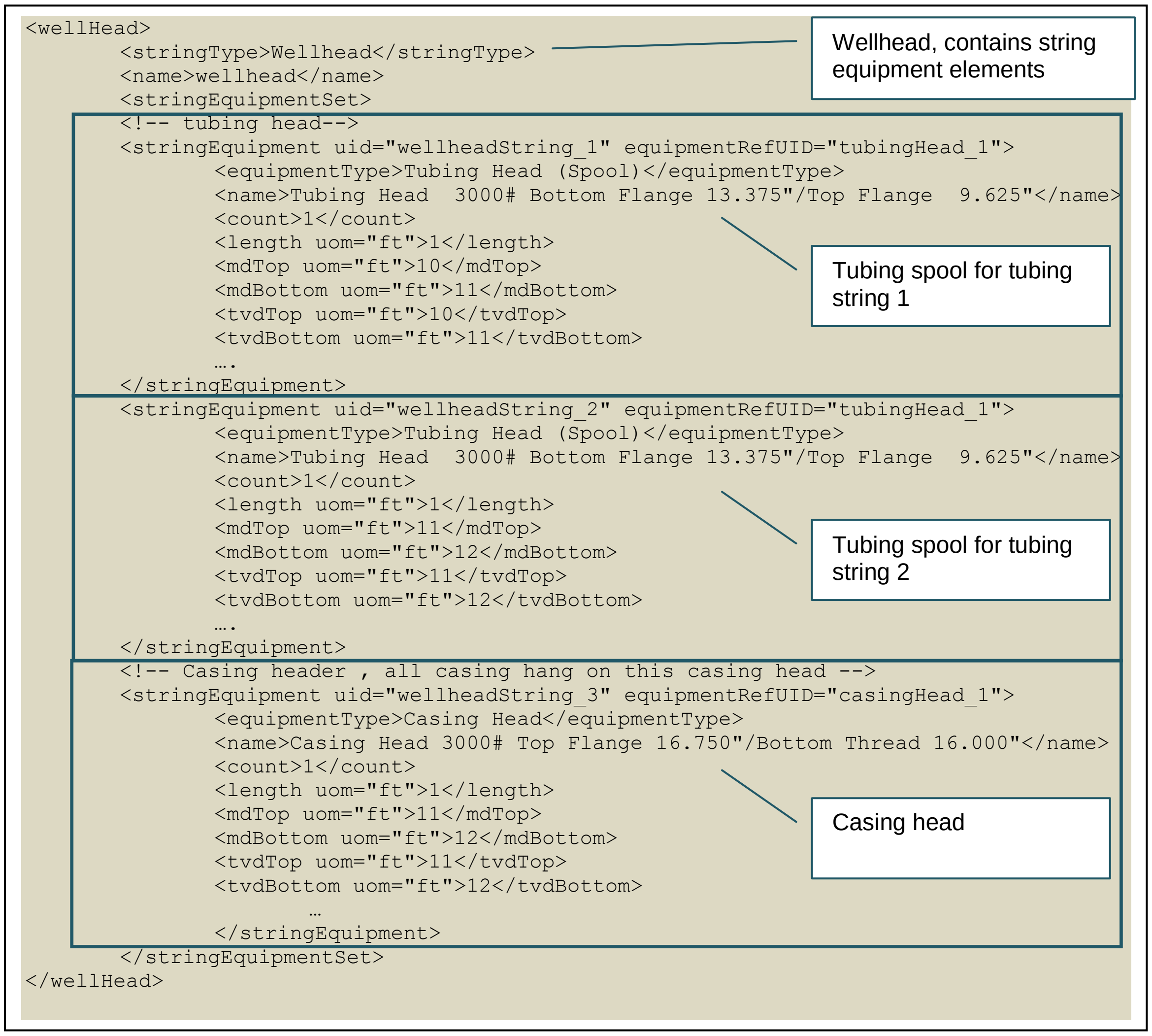
outsideString is used to denote this Accessory (Cement) is outside its parent string.

The other downhole strings follow with similar details.

3.1.6 wellHead

In this version of the data model, the wellhead is treated as a kind of downholeString. It contains some data elements but is not intended to be comprehensive.

The wellhead for the Snapshot use case example contains two tubing head spools and a casing head.



```
<wellHead>
        <stringType>Wellhead</stringType>
        <name>wellhead</name>
        <stringEquipmentSet>
        <!-- tubing head-->
        <stringEquipment uid="wellheadString_1" equipmentRefUID="tubingHead_1">
                <equipmentType>Tubing Head (Spool)</equipmentType>
                <name>Tubing Head  3000# Bottom Flange 13.375"/Top Flange  9.625"</name>
                <count>1</count>
                <length uom="ft">1</length>
                <mdTop uom="ft">10</mdTop>
                <mdBottom uom="ft">11</mdBottom>
                <tvdTop uom="ft">10</tvdTop>
                <tvdBottom uom="ft">11</tvdBottom>
                ….
        </stringEquipment>
        <stringEquipment uid="wellheadString_2" equipmentRefUID="tubingHead_1">
                <equipmentType>Tubing Head (Spool)</equipmentType>
                <name>Tubing Head  3000# Bottom Flange 13.375"/Top Flange  9.625"</name>
                <count>1</count>
                <length uom="ft">1</length>
                <mdTop uom="ft">11</mdTop>
                <mdBottom uom="ft">12</mdBottom>
                <tvdTop uom="ft">11</tvdTop>
                <tvdBottom uom="ft">12</tvdBottom>
                ….
        </stringEquipment>
        <!-- Casing header , all casing hang on this casing head -->
        <stringEquipment uid="wellheadString_3" equipmentRefUID="casingHead_1">
                <equipmentType>Casing Head</equipmentType>
                <name>Casing Head 3000# Top Flange 16.750"/Bottom Thread 16.000"</name>
                <count>1</count>
                <length uom="ft">1</length>
                <mdTop uom="ft">11</mdTop>
                <mdBottom uom="ft">12</mdBottom>
                <tvdTop uom="ft">11</tvdTop>
                <tvdBottom uom="ft">12</tvdBottom>
                        …
                </stringEquipment>
        </stringEquipmentSet>
</wellHead>
```

Wellhead, contains string equipment elements
Tubing spool for tubing string 1
Tubing spool for tubing string 2
Casing head

3.1.7 equipmentSet

The equipmentSet contains a collection of equipment that comprises all the equipment in the downholeEquipment instance. Each piece of equipment in the set has a type and other attributes common across all instances of that piece of equipment. The stringEquipment items (and any accessory items) then reference these common pieces of equipment, which have their common attributes. Example: several sections of tubing string may all have the same type of tubing, so can all reference the same entry in the equipmentSet element.

This concept is outlined in Section 2.2.1.4, and illustrated in Section 3.1.2.

The example below shows some of the equipment referenced in the preceding code snippets.

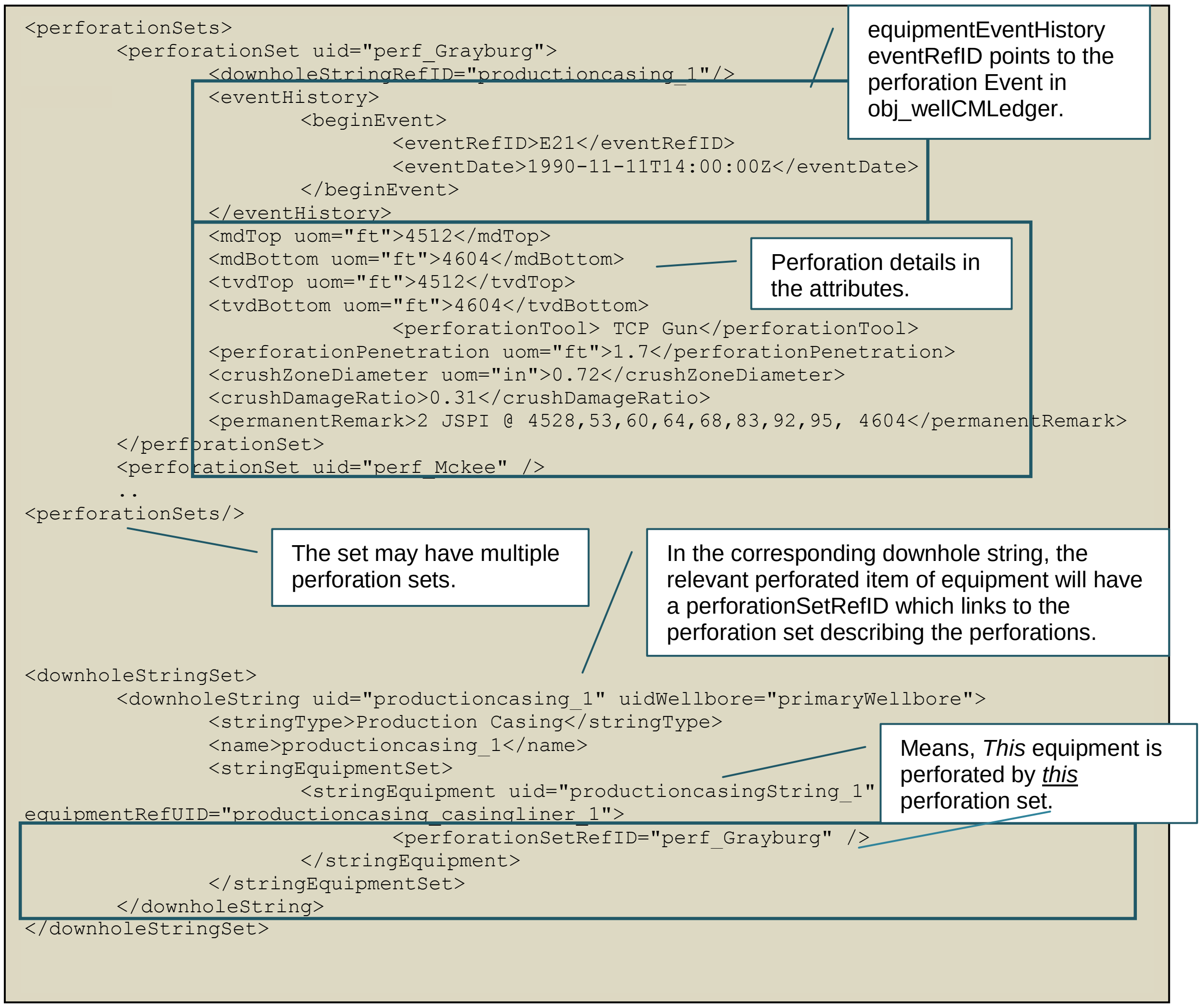
```
<equipmentSet>
        <equipment uid="tubingHead_1">
                <equipmentName>tubing Head</equipmentName>
                <equipmentType>Tubing Head (Spool)</equipmentType>
                <model>API 3000#</model>
                <material>Steel</material>
        </equipment>
        <equipment uid="casingHead_1">
                <equipmentName>casing Head</equipmentName>
                <equipmentType>Casing Head</equipmentType>
                <model>API 3000#</model>
                <material>Steel</material>
        </equipment>
        <equipment uid="tubingOD2.875_1">
                <equipmentName> tubing</equipmentName>
                <equipmentType>Tubing</equipmentType>
                <material>Steel</material>
                <grade>J-55</grade>
                <unitWeight uom="lbm">6.5</unitWeight>
                <OD uom="in">2.875</OD>
                <ID uom="in">2.441</ID>
                <drift uom="dega">2.347</drift>
        </equipment>
        <equipment uid="cement_behindcasing1">
                <equipmentName>Cement</equipmentName>
                <equipmentType>Cement (behind Casing)</equipmentType>
        </equipment>
        <equipment uid="fishinWellbore_1">
                <equipmentType>Fish</equipmentType>
                <name>Fish in wellbore</name>
        </equipment>
</equipmentSet>
```

The UID of the equipment is what is referenced from each instance of this type of equipment in the string equipments elements.

The attributes of each type of equipment are listed in the loader file, which can be used as a way to extend both the types and the attributes of equipment. See Sections 4.5.

3.1.8 Perforation Set

The perforationSet contains a collection of contactInterval data-objects in the downholeComponent parent. A contactInterval corresponds to the *physical* attributes of the connection between the wellbore in the reservoir (see section 2.2.3 for the conceptual details). The contactInterval contains the type, position and other attributes of perforations. It can also represent an openhole connection to the reservoir. The Contact Interval also has a reference to the Wellbore Completion (i.e. flow stream in the wellbore) to which it contributes. See Figure 7 and Section Well Completion for more details on how flow streams are integrated.



```
<perforationSets>
        <perforationSet uid="perf_Grayburg">
                <downholeStringRefID="productioncasing_1"/>
                <eventHistory>
                        <beginEvent>
                                <eventRefID>E21</eventRefID>
                                <eventDate>1990-11-11T14:00:00Z</eventDate>
                        </beginEvent>
                </eventHistory>
                <mdTop uom="ft">4512</mdTop>
                <mdBottom uom="ft">4604</mdBottom>
                <tvdTop uom="ft">4512</tvdTop>
                <tvdBottom uom="ft">4604</tvdBottom>
                                <perforationTool> TCP Gun</perforationTool>
                <perforationPenetration uom="ft">1.7</perforationPenetration>
                <crushZoneDiameter uom="in">0.72</crushZoneDiameter>
                <crushDamageRatio>0.31</crushDamageRatio>
                <permanentRemark>2 JSPI @ 4528,53,60,64,68,83,92,95, 4604</permanentRemark>
        </perforationSet>
        <perforationSet uid="perf_Mckee" />
        ..
<perforationSets/>


<downholeStringSet>
        <downholeString uid="productioncasing_1" uidWellbore="primaryWellbore">
                <stringType>Production Casing</stringType>
                <name>productioncasing_1</name>
                <stringEquipmentSet>
                        <stringEquipment uid="productioncasingString_1"
equipmentRefUID="productioncasing_casingliner_1">
                                <perforationSetRefID="perf_Grayburg" />
                        </stringEquipment>
                </stringEquipmentSet>
        </downholeString>
</downholeStringSet>
```

equipmentEventHistory eventRefID points to the perforation Event in obj_wellCMLedger.

Perforation details in the attributes.

The set may have multiple perforation sets.

In the corresponding downhole string, the relevant perforated item of equipment will have a perforationSetRefID which links to the perforation set describing the perforations.

Means, *This* equipment is perforated by *this* perforation set.

3.2 Use Case 2: Change Log

The Change Log use case concerns the exchange of the completion data which has changed over a specified time period. Components that did not change are not exchanged. For more background information, see Section 2.1.2, page 11.

The following table shows the Events from the sample well history. In this table in the "Note" column, gray is used to denote Events for which there is no WITSML referenced object, e.g. for a safety meeting, Event 06, there is no WITSML data-object so none can be referenced for further details. A pink color indicates that a number of items of equipment in the completion could be affected by this event. For full details, we may picture child events under these events, which would set out the operation step-by-step referencing single items of equipment installed or changed. A green color indicates that there can be a referenced single example object in the completion, e.g., the cement in Events E09 and E13, and the bridge plug in E14 and E16.

Event	Time Start	Time End	Event Type	Note
E00	1990-11-11 00:00:00Z	1990-11-22 00:00:00Z		Parent Event
E01	1990-11-11 00:08:00Z	1990-11-11 00:09:00Z	Drilling Plan	no completion object referenced
E02	1990-11-11 00:09:00Z	1990-11-11 00:10:00Z	permit	no completion object referenced
E03	1990-11-11 00:10:00Z	1990-11-11 00:14:00Z	site work	no completion object referenced
E04	1990-11-11 00:14:00Z	1990-11-11 00:18:00Z	environmental survey	no completion object referenced
E05	1990-11-11 00:18:00Z	1990-11-11 00:10:00Z	rig up	no completion object referenced
E06	1990-11-11 00:10:00Z	1990-11-11 00:11:00Z	safety meeting	no completion object referenced
E07	1990-11-11 00:11:00Z	1990-11-11 00:17:00Z	drill spud	could reference borehole being drilled, wellhead, surface casing…
E08	1990-11-12 00:08:00Z	1990-11-12 00:14:00Z	drill - day 1	could span drill plus run casing…many events if detail required
E09	1990-11-12 00:14:00Z	1990-11-12 00:18:00Z	cement	references cement element of completion
E10	1990-11-13 00:08:00Z	1990-11-13 00:18:00Z	drill - day 2	should reference borehole being drilled
E11	1990-11-14 00:08:00Z	1990-11-14 00:18:00Z	drill - day 3	should reference borehole being drilled
E12	1990-11-15 00:08:00Z	1990-11-15 00:14:00Z	drill - day 4	could span drill plus run casing…many events if detail required
E13	1990-11-15 00:14:00Z	1990-11-15 00:18:00Z	cement	references cement element of completion
E14	1990-11-15 00:08:00Z	1990-11-16 00:10:00Z	set bridge plug	references bridge plug element of completion (begin event)
E15	1990-11-16 00:10:00Z	1990-11-16 00:16:00Z	pressure test	no completion object referenced
E16	1990-11-16 00:16:00Z	1990-11-16 00:18:00Z	mill bridge plug	references bridge plug element of completion (end event)

Figure 10. Partial list of events to illustrate Change Log use case.

The events colored yellow are the ones used to illustrate the example Change Log use case here.

The usage illustrated in this section and as shown in Figure 2 (page 11), shows exchange of two types of data for each Event (or change):

- The "Event", using the wellCMLedger, which is data about the "job" that created the change.

- The changed completion component, using the downholeComponent, which is data about the hardware that changed.

These two items of data can contain cross references to each other as shown below. However, depending on the usage requirement, it may be adequate to exchange just one of these types of data or the other, rather than needing both.

Note that it is possible for these Events to be "contained" in a “parent job”, as outlined in Section 2.1.4. For example, a daily report may contain several Events of that day, or a service company report of a job that it performed can contain several Events as stages in the job. These types of examples are illustrated in Section 4.4 and not shown in these use case examples.

3.2.1 Basic Event Using Well CM Ledger

The first Event in the example is E06, which is a safety meeting. This is, as shown in the table in Figure 10, an example where there is no WITSML data-object that can be referenced from the ledger. That is, there is neither a component in the well completion (which would be in the downholeComponent data-object), nor a WITSML “Job” data-object corresponding to a safety meeting. Therefore the entry just lists the Event details with no references to external data-objects.

```
<wellCMLedger uid="E06" uidWell="VGASAU029" uidWellbore="primaryWellbore">
       <nameWellbore>primaryWellbore</nameWellbore>
       <parentEventID>E00</parentEventID>
       <dTimStart>1990-11-11T10:00:00Z</dTimStart>
       <dTimEnd>1990-11-11T11:00:00Z</dTimEnd>
       <duration uom="h">1</duration>
       <type>Safety Meeting</type>
       <isPlan>true</isPlan>
</wellCMLedger>
```

Shows the parent event.

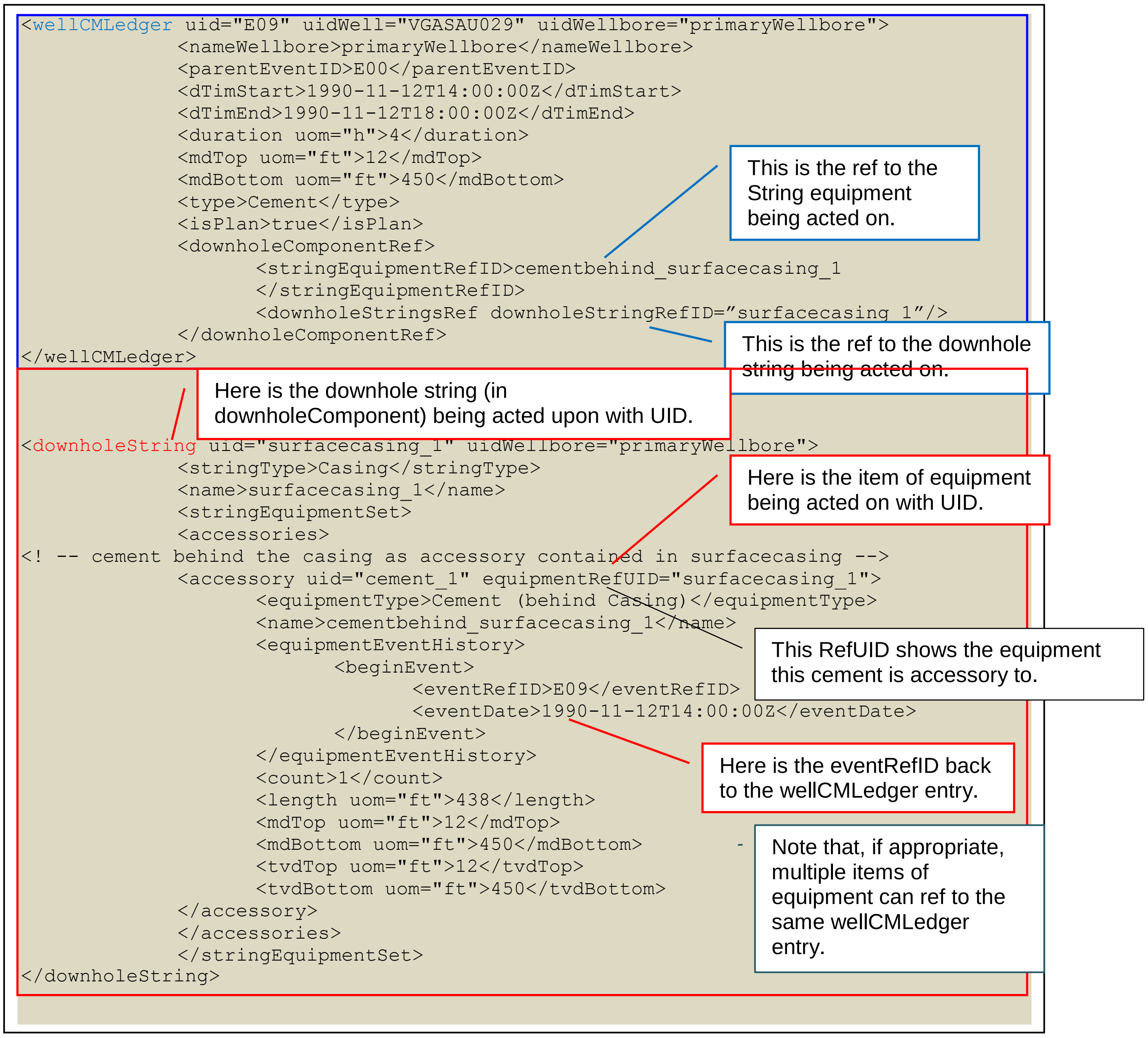
Note that the *type* attribute (the type of Event) allows you to pick from an enumerated list, OR to create your own value. The xsd includes an extensionAny schema type.

3.2.2 Event Referencing a Completion Component Using Ref to downholeComponent

The second event is E07 which concerns spudding the well. In this example, the event shown is for running the surface casing string (there could be events for other stages, e.g., installing the wellhead, drilling the surface hole, etc, before this example event).

Because the event acts on an item of the physical completion, there can be a link from the event in the wellCMLedger with reference to the downhole string and to the item of string equipment within that string. Then within that item of equipment, there is a reference back to the wellCMLedger instance that installed (or if appropriate) removed this item of equipment. Both links are shown in this code example, for cementing the surface casing, (the borders of the callouts on the code example are color coded blue and red, for wellCMLedger and downhole string, respectively).

Note that the reference to equipment is in the element downholeComponentRef, which can contain any number of references to String Equipment, Borehole String, Downhole String, or Perforation Set. A



```
<wellCMLedger uid="E09" uidWell="VGASAU029" uidWellbore="primaryWellbore">
             <nameWellbore>primaryWellbore</nameWellbore>
             <parentEventID>E00</parentEventID>
             <dTimStart>1990-11-12T14:00:00Z</dTimStart>
             <dTimEnd>1990-11-12T18:00:00Z</dTimEnd>
             <duration uom="h">4</duration>
             <mdTop uom="ft">12</mdTop>
             <mdBottom uom="ft">450</mdBottom>
             <type>Cement</type>
             <isPlan>true</isPlan>
             <downholeComponentRef>
                    <stringEquipmentRefID>cementbehind_surfacecasing_1
                    </stringEquipmentRefID>
                    <downholeStringsRef downholeStringRefID=”surfacecasing_1”/>
             </downholeComponentRef>
</wellCMLedger>

<downholeString uid="surfacecasing_1" uidWellbore="primaryWellbore">
             <stringType>Casing</stringType>
             <name>surfacecasing_1</name>
             <stringEquipmentSet>
             <accessories>
<! -- cement behind the casing as accessory contained in surfacecasing -->
             <accessory uid="cement_1" equipmentRefUID="surfacecasing_1">
                    <equipmentType>Cement (behind Casing)</equipmentType>
                    <name>cementbehind_surfacecasing_1</name>
                    <equipmentEventHistory>
                           <beginEvent>
                                  <eventRefID>E09</eventRefID>
                                  <eventDate>1990-11-12T14:00:00Z</eventDate>
                           </beginEvent>
                    </equipmentEventHistory>
                    <count>1</count>
                    <length uom="ft">438</length>
                    <mdTop uom="ft">12</mdTop>
                    <mdBottom uom="ft">450</mdBottom>
                    <tvdTop uom="ft">12</tvdTop>
                    <tvdBottom uom="ft">450</tvdBottom>
             </accessory>
             </accessories>
             </stringEquipmentSet>
</downholeString>
```

This is the ref to the String equipment being acted on.

This is the ref to the downhole string being acted on.

Here is the downhole string (in downholeComponent) being acted upon with UID.

Here is the item of equipment being acted on with UID.

This RefUID shows the equipment this cement is accessory to.

Here is the eventRefID back to the wellCMLedger entry.

Note that, if appropriate, multiple items of equipment can ref to the same wellCMLedger entry.

3.2.3 Adding Further "Job" Details Using Event Extension Element

In certain types of Events the element of eventExtension can be used to supply more details. These details may be in the form of a reference to a WITSML object external to the Completion object, or in the form of additional data elements packaged within the Completion object.

Example: the WITSML cement object can supply more details for an Event which is a "cement job". The link to the WITSML cement object is provided by the cementExtension form of eventExtension.

Note that the reference to the completion equipment (as illustrated in Section3.2.2) remains—this is the "equipment" reference, in contrast to eventExtension which is the "job" reference or details, which resulted in the equipment being as it is.

The eventExtension types are listed in the table below. The column "WITSML data object?" indicates whether an independent WITSML data object is referenced. The column "Completion data element?" indicates which completion data element the event extension points to. Where neither kind of reference exists, it may be assumed that the event extension contains adequate data elements itself to describe the operation.

Extension type	Type of operation referred to	WITSML data object?	Completion data element?
Cement	Cement job	cementJob	
CleanFill	Cleaning fill		
DirectionalSurvey	Directional survey	trajectory	
LostCirculation	Volume of lost circulation		
BHP	Bottom hole pressure survey	BHP	
AcidizeFrac	Stimulation by acidizing or fracturing	StimJob	
Job	Generic job (with reasons)		
PressureTest	Pressure test		
WaitingOn	Lost time while waiting on some other event (with categories)		
Perforation	Perforation		PerforationSet
FluidReport	Fluid report	fluidReport	
Downhole	Generic downhole		DownholeComponent

Event E09 is an example (also used in 3.2.2 above), comprising a "cement job". Note that the types of "xxxJob" WITSML data-objects which may be referenced are limited to those contained in the eventExtension element. However, further types could be added in future.

```
<wellCMLedger uid="E09" uidWell="VGASAU029" uidWellbore="primaryWellbore">
       <nameWellbore>primaryWellbore</nameWellbore>
       <parentEventID>E00</parentEventID>
       <dTimStart>1990-11-12T14:00:00Z</dTimStart>
       <dTimEnd>1990-11-12T18:00:00Z</dTimEnd>
       <duration uom="h">4</duration>
       <mdTop uom="ft">12</mdTop>
       <mdBottom uom="ft">450</mdBottom>
       <type>Cement</type>
       <isPlan>true</isPlan>
       <downholeComponentRef>
              <stringEquipmentRefID>cementbehind_surfacecasing_1</stringEquipmentRefID>
              <downholeStringsRef downholeStringRefID=”surfacecasing_1”/>
       </downholeComponentRef>
       <eventExtension>
              <cementExtension>
                     <cementJobRefID>cement_job1</cementJobRefID>
              </cementExtension>
       </eventExtension>
       <participant uid="participant_1">
              <objectReference object="trajectory">trajectory_1</objectReference>
              <sequence1 description=”primary”>1</sequence1>
              <referenceDepth uom=”ft”>400</referenceDepth>
       </participant >
</wellCMLedger>
```

Event details

The eventExtenstion gives a link to a WITSML data-object with further details of this event, in this example a WITSML cement job object.

3.2.4 Adding Further “Job” Details Using Participant Element

The Participant element of the wellCMLedger can be used to reference WITSML Log objects relevant to the Job. In the example above, this element provides a reference to a trajectory object and a set of elements to define the range reference times over which data is relevant to the job. Log objects may also be referenced, with indexing information available.

3.2.5 Adding Further “Job” Details Using Extension Name Values

The element extNameValues can be added under a Participant element. The similar element extensionNameValue can be added under any of the eventExtension job types, or under the wellCMLedger element itself. Obviously, use of these elements will render this part of the data non-interoperable with other systems unless these are customized to the same extensions.

3.3 Use Case 3: Cumulative History

The Cumulative History use case refers to the transfer of data corresponding to the whole history of the completion. The outline of this is described in Section 2.1.3, page 12. In general, the usage patterns described for the Change Log use case (see Section 3.2) apply here. For example, Events and changed Completion components can both be exchanged or just one or the other.

The collection of wellCMLedger data-objects, in time sequence, is expected to be the whole list of events that have been recorded for this well. The collection of downholeComponent data-objects is expected to build up to the current snapshot at the time of the last of the events exchanged. However, unlike a simple snapshot, the collection of downhole components may include items that were previously in the well but that have since been removed. Eliminating all the items that have endEvents in their equipmentEventHistory elements will result in the collection that has been installed but not yet removed, i.e., the current snapshot.

The schema has been designed to transfer drilling data in the sense of the sequence of events and the progression of borehole depths, and the installations of casing, cementing, and completion as described in this document. The “details” of drilling as transferred in the set of WITSML data-objects, are not repeated, but can be referenced using Event Extension (see Section 3.2.3). It is therefore possible that the life of the well including drilling, initial completion, and subsequent well services and workovers through to abandonment, can all be tracked—or any single phase can be.

Figure 11 shows an example section of event ledgers related to this use case. The code examples in Section 3.2 are based on these event ledgers, and the same principles as illustrated there apply here. Note that this list of events is a simplification in that some of the events listed are shown as having impact upon more than one component of the completion. We may think of these as parent events containing more detail we would expect to capture in real life.

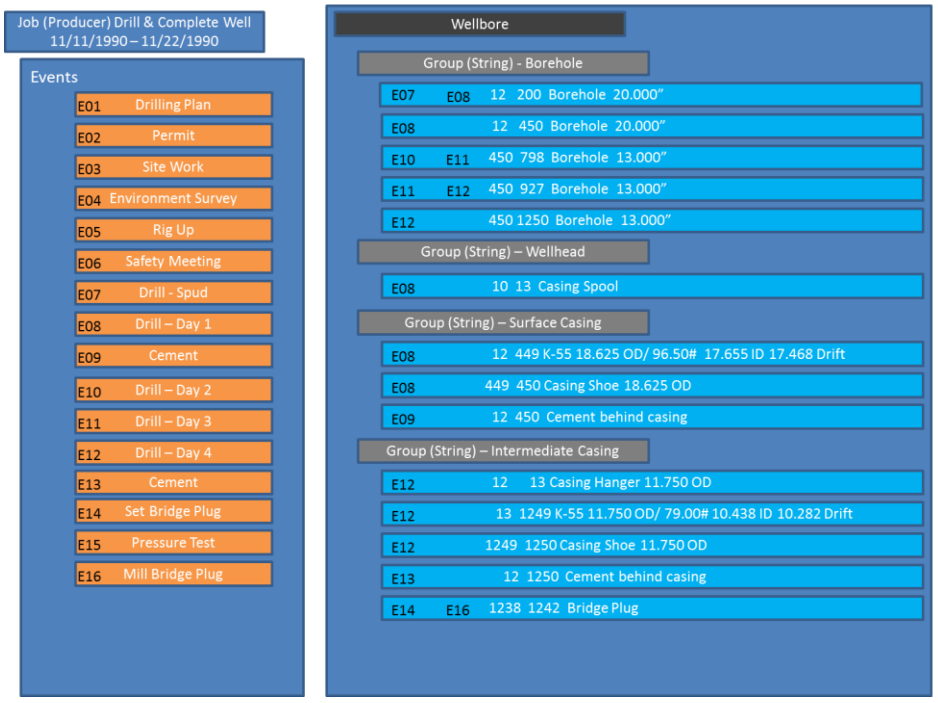

Job (Producer) Drill & Complete Well
11/11/1990 – 11/22/1990
Events
E01 Drilling Plan
E02 Permit
E03 Site Work
E04 Environment Survey
E05 Rig Up
E06 Safety Meeting
E07 Drill - Spud
E08 Drill – Day 1
E09 Cement
E10 Drill – Day 2
E11 Drill – Day 3
E12 Drill – Day 4
E13 Cement
E14 Set Bridge Plug
E15 Pressure Test
E16 Mill Bridge Plug
Wellbore
Group (String) - Borehole
E07 E08 12 200 Borehole 20.000”
E08 12 450 Borehole 20.000”
E10 E11 450 798 Borehole 13.000”
E11 E12 450 927 Borehole 13.000”
E12 450 1250 Borehole 13.000”
Group (String) – Wellhead
E08 10 13 Casing Spool
Group (String) – Surface Casing
E08 12 449 K-55 18.625 OD/ 96.50# 17.655 ID 17.468 Drift
E08 449 450 Casing Shoe 18.625 OD
E09 12 450 Cement behind casing
Group (String) – Intermediate Casing
E12 12 13 Casing Hanger 11.750 OD
E12 13 1249 K-55 11.750 OD/ 79.00# 10.438 ID 10.282 Drift
E12 1249 1250 Casing Shoe 11.750 OD
E13 12 1250 Cement behind casing
E14 E16 1238 1242 Bridge Plug

Figure 11. Portion of event ledger for the Cumulative History use case showing example Events on left and the Downhole Components these events affect, by string, on the right. (Detail from Figure 3. Example well cumulative history, page 12).

The cumulative history would be expected to be transferred as a wellCMLedgers plural object, which would contain all the events (each a wellCMLedger), together with a downholeComponent object containing all the components of the completion that have ever been impacted by the events listed—both those components subsequently removed and those still installed. We can think of this construct as enabling the reconstruction of a snapshot at any point in the history, by filtering the list of downhole components down to those which have been impacted by the events leading up to that point and excluding the components impacted by subsequent events. We can also think of reconstructing a change log by selecting only the events between two points in time, and filtering the downhole components to be those impacted by the events that we have selected.

```
</witsml:wellCMLedgers>
       <witsml:wellCMLedger uid="E14" uidWell="VGASAU029" uidWellbore="primaryWellbore"/>
       ..
       <witsml:wellCMLedger uid="E15" uidWell="VGASAU029" uidWellbore="primaryWellbore"/>
       ..
       <witsml:wellCMLedger uid="E16" uidWell="VGASAU029" uidWellbore="primaryWellbore"/>
       ..
       <witsml:wellCMLedger uid="E26" uidWell="VGASAU029" uidWellbore="primaryWellbore"/>
</witsml:wellCMLedgers>

<downholeComponent uid="downholecomp_Job1" uidWell="VGASAU029">
       <nameWell>VGASAU029</nameWell>
       <name>downholecomp_Job1</name>
       <wellHead>
              <stringEquipmentSet>
                     <stringEquipment uid="wellheadString_1"
equipmentRefUID="tubingHead_1">
                            <witsml:equipmentEventHistory>
                                   <beginEvent>
                                          <eventRefID>E20</eventRefID>
                                          <eventDate>1990-11-11T11:00:00Z</eventDate>
                                   </beginEvent>
                            </witsml:equipmentEventHistory>
                     </stringEquipment>
              </stringEquipmentSet>
       </wellHead>
       <boreholeStringSet>
              <boreholeString uid="borehole-1">
                     ..
                     <borehole/>
                            ..
                     <geologyFeature uid="Aquifer_1"/>
              </boreholeString>
       </boreholeStringSet>
       <downholeStringSet>
              ..
              <downholeString uid="tubing_1" uidWellbore="primaryWellbore">
                     <stringEquipmentSet>
                            ..
                            <witsml:equipmentEventHistory>
                                   <beginEvent>
                                          <eventRefID>E22</eventRefID>
                                          <eventDate>1990-11-13T11:00:00Z</eventDate>
                                   </beginEvent>
                            </witsml:equipmentEventHistory>
                            <witsml:equipmentEventHistory>
                                   <endEvent>
                                          <eventRefID>E26</eventRefID>
                                          <eventDate>1993-11-11T11:00:00Z</eventDate>
                                   </endEvent>
                            </witsml:equipmentEventHistory>
                     </stringEquipmentSet>
              </downholeString>
       </downholeStringSet>
</downholeComponent>
```

wellCMLedger entries over all history

downholeComponent entries over history, expected to have equipmentEventHistories

Components still current at end of history (no endEvent)

4 Code Examples: for Commonly Used Equipment, Scenarios, and Events

This section contains example of XML code for types of equipment commonly used in well completions.

4.1 Flow (Well/Wellbore Completions and Contact Intervals)

As explained in 2.2.5 (page 20), Well Completion is a well-level flow and Wellbore Completion is flow over a section of wellbore. Wellbore completion is regarded as the basic “reporting” level of flow. However, a wellbore completion may be composed of multiple Contact Intervals, each of which represents a physical connection from wellbore to reservoir, and may be of these types: Perforated, Open Hole, Slotted, or Gravel Pack.

Thus there are several distinct types of data related with flow:

- The physical aspects of the perforation into the borehole and out into the formation
- The identity of the tubing or casing that is perforated
- The service or related information concerning how the perforating was implemented

Or,

- The Holes (or slots) in the equipment such as casing or tubing… as manufactured

Or,

- The Open Hole Completion details

Or,

- The Gravel Pack Completion details

Plus:

- The associated linkage to a Wellbore Completion and Well Completion
- The status and history of the Completion and Contact Interval (open, closed, etc.) showing when it is capable of flow into or out of the borehole.

4.1.1 Physical Aspects of Perforation into the Formation

The borehole/formation information has physical characteristics associated with the service of perforating, such as the resultant hole size in the formation, pattern, spacing, crush area, depth of the holes into the formation, etc. This data is contained within a perforationSet (For more details, see Section 3.1.8, page 32).

```
<downholeComponent>
<perforationSets>
       <perforationSet uid="perf_Mckee"
              <boreholeStringRefID> "borehole-1" <boreholeStringRefID/>
              <downholeStringRefID> "productioncasing_1" <downholeStringRefID/>
              <eventHistory>
                     <beginEvent>
                            <eventRefID>E22</eventRefID>
                            <eventDate>1990-11-19T14:00:00Z<
                     </beginEvent>
              </eventHistory>
              <mdTop uom="ft">4625</mdTop>
              <mdBottom uom="ft">4698</mdBottom>
              <tvdTop uom="ft">4616</tvdTop>
              <tvdBottom uom="ft">4698</tvdBottom>
              <perforationTool>TCP Gun</perforationTool>
              <perforationPenetration uom="ft">1.7</perforationPenetration>
              <crushZoneDiameter uom="in">0.69</crushZoneDiameter>
              <crushDamageRatio>0.28</crushDamageRatio>
              <perforationDate>1990-11-12T08:00:00Z</performationDate>
              <permanentRemark>2 JSPI @ 4616, 26, 33, 37, 64, 69, 74, 87, 94,
4704 &amp; 4712'.</permanentRemark>
       </perforationSet>
</perforationSets>
</downholeComponent>
```

Show the borehole and downhole string which this perforation set affects.

4.1.2 Identity of the Borehole and Equipment Perforated

As shown in the snippet above, the perforationSet contains references to the borehole and to the equipment which it perforates.

4.1.3 Service Related Information for Perforating

The service in which the holes are created ("perforating") is captured in the wellCMLedger. There is an Event Extension type for perforating, in which "perforation job" information can be stored (see example below).

```
<wellCMLedger uid="E21" uidWell="VGASAU029" uidWellbore="primaryWellbore">
        <nameWellbore>primaryWellbore</nameWellbore>
        <parentEventID>E00</parentEventID>
        <dTimStart>1990-11-18T12:00:00Z</dTimStart>
        <dTimEnd>1990-11-18T18:00:00Z</dTimEnd>
        <duration uom="h">6</duration>
        <mdTop uom="ft">4512</mdTop>
        <mdBottom uom="ft">4604</mdBottom>
        <type>perforate</type>
        <isPlan>true</isPlan>
        <eventExtension>
                <perforationExtension>
                <perforationSetRefID>perf_Garyburg</perforationIntervalRefID>
                <perforating>
                        <stageNumber>1</stageNumber>
                        <reservoirPressure uom="psi">210</reservoirPressure>
                        <conveyanceMethod>Wireline</conveyanceMethod>
                        <shotsPlanned>2</shotsPlanned>
                        <shotsDensity>2</shotsDensity>
                        <perforationCompany>weatherford</perforationCompany>
                        <gunLeftInHole>true</gunLeftInHole>
                </perforating>
                </perforationExtension>
        </eventExtension>
</wellCMLedger>
```

4.1.4 Holes or Slots

Holes or Slots are contained in the equipment information. Holes or Slots may have been drilled or cut at the manufacturer. In a casing or liner, these will likely provide a connection to the reservoir. In that case there will be a contactInterval for the length of wellbore open to flow. Holes or slots in Tubing are similar, except that there may be no direct association with the reservoir, and thus only the hole or slot data is pertinent.

```
<downholeComponent uid="downholecomp_1" uidWell="VGASAU029">
<equipmentSet>
<equipment uid="GravelPackScreen_1">
        <equipmentName>GravelPackScreen_1</equipmentName>
        <equipmentType>Gravel Pack Screen</equipmentType>
        <slotAsManufactured>
                <slotHeight uom="in">12</slotHeight>
                <slotWidth uom="in">2.5</slotWidth>
                <slotCenterDistance uom="in">20</slotCenterDistance>
                <slotCount>5</slotCount>
        </slotAsManufactured>
        <holeAsManufactured>
                <holeDiameter uom="in">5</holeDiameter>
                <holeAngle uom="dega">45</holeAngle>
                <holePattern>regular</holePattern>
                <holeDensity uom="in">2</holeDensity>
                <holeCount>10</holeCount>
        </holeAsManufactured>
</equipment>
</equipmentSet>
</downholeComponent>
```

4.1.5 Open Hole Completion

Openhole completions do not have any physical objects associated with them, because by definition they are merely a section of as-drilled borehole.

4.1.6 Gravel Pack Completion

Gravel pack completions can comprise a complex set of equipment. The details of this are dealt with in full in Section 4.2, page 45.

4.1.7 Linkage from Well and Wellbore Completion to Physical Contact Interval

The Well Completion data-object associates flow from one or more Wellbore Completions (which may have flowed from different wellbores) with a wellhead stream at the surface. These streams can flow in either direction (producing or injecting). For more on the conceptual linkages, see Section 2.2.3, page 19.

The well completion is a top level data-object that can be referenced without including all of the detail of a well underneath. This may be a major advantage in some situations where a completion needs to be referenced but there is no need for any downhole component information.

```
wellCompletion uidWell="VGASAU029" uid="completion_1">
       <nameWell>VGASAU029</nameWell>
       <name>well_completion_1</name>
       <fieldID>1</fieldID>
       <fieldCode>North Area</fieldCode>
       <fieldType>Hillside slope</fieldType>
       <effectiveDate>1990-11-11T0:00:00Z</effectiveData>
       <expiredDate></expiredDate>
</wellCompletion>
```

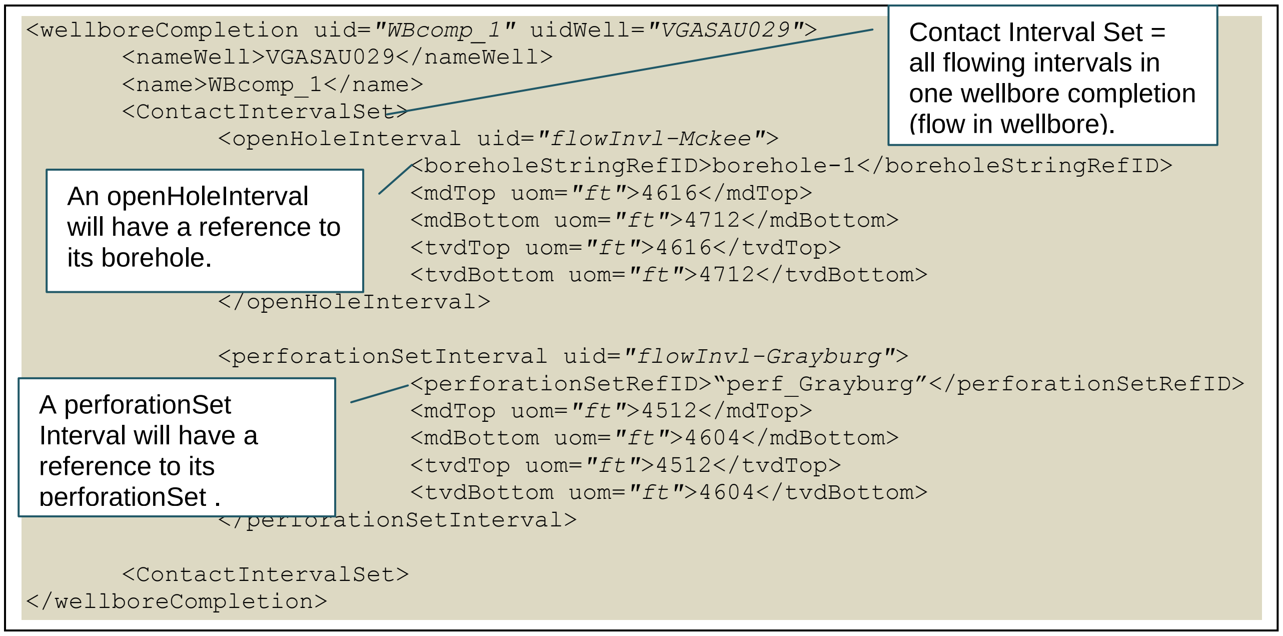
Another aspect of the well completion—which will likely be addressed in a later release by the PRODML SIG—is the relationship between wellhead streams and a well completion. There may be multiple streams flowing between a well completion and the wellhead, such as from a tubing string and casing head. In the physical model in this schema, the aim was to connect the downhole strings to the wellhead, laying the foundation to define wellhead streams above the wellhead which can then be tied to a well completion.

The Wellbore Completion data-object associates flow from one or more Contact Intervals with a flow in the wellbore.

The Contact Interval object represents one physical connection between the wellbore and the reservoir. Each contact interval may be one of four kinds. This construct means that a single Wellbore Completion (one flow within a wellbore) can originate from a set of Contact Intervals, each one of which can be of a different kind.

The four kinds of contact interval are list below. They all have common data concerning the Contact Interval (depths, status, geological feature connected, etc.), but differ in the reference to another part of the Completion data-object, which serves as a pointer to their physical description.

- Perforated (references a perforationSet)
- Slotted Liner (references an item of stringEquipment)
- Open Hole (references a boreholeString)
- Gravel Packed (references a downholeString)



```
<wellboreCompletion uid="WBcomp_1" uidWell="VGASAU029">
       <nameWell>VGASAU029</nameWell>
       <name>WBcomp_1</name>
       <ContactIntervalSet>
              <openHoleInterval uid="flowInvl-Mckee">
                            <boreholeStringRefID>borehole-1</boreholeStringRefID>
                            <mdTop uom="ft">4616</mdTop>
                            <mdBottom uom="ft">4712</mdBottom>
                            <tvdTop uom="ft">4616</tvdTop>
                            <tvdBottom uom="ft">4712</tvdBottom>
              </openHoleInterval>

              <perforationSetInterval uid="flowInvl-Grayburg">
                            <perforationSetRefID>"perf_Grayburg"</perforationSetRefID>
                            <mdTop uom="ft">4512</mdTop>
                            <mdBottom uom="ft">4604</mdBottom>
                            <tvdTop uom="ft">4512</tvdTop>
                            <tvdBottom uom="ft">4604</tvdBottom>
              </perforationSetInterval>

       <ContactIntervalSet>
</wellboreCompletion>
```

Contact Interval Set = all flowing intervals in one wellbore completion (flow in wellbore).

An openHoleInterval will have a reference to its borehole.

A perforationSet Interval will have a reference to its perforationSet .

4.1.8 Recording Status of Flow (Well/Wellbore Completions and Contact Intervals)

Status of flow or contact interval may be recorded at several places in the hierarchy:

1. Well Completion has:
    a. Current Status attribute (active, inactive, permanent abandonment, planned, suspended, temporarily abandoned, testing)
    b. Status Date (the date at which the current status came into effect)
    c. Any number of Status History elements, each of which has a status with the same enum values as Current Status, start and end dates and depths. This enables the history of changes to a well completion to be recorded, including the fact that the flow may have been partially abandoned, e.g. by selectively squeezing some of a set of perforations).
2. Wellbore Completion has: the same elements as described above for Well Completion.
3. Each of the Intervals in the Contact Interval Set of the Wellbore Completion has a number of Interval Status History elements, such as, start and end dates and depths. Slotted, openhole and gravel pack intervals contain a Physical Status attribute (open, closed, proposed), while perforation intervals contain a Perforation Status attribute (open, squeezed, proposed). This represents the fact that these Intervals are physical connections rather than representations of the ability to flow. They can therefore only be physically open or closed. In contrast, Well Completion and Wellbore Completion are regarded as representing flow which may be controlled by some external regulation, and this is why they have the different enum values possible.

4.2 Gravel Pack Assembly

There is a design pattern built into the completion data-object schema that allows the definition of an assembly and its associated component parts. It is up to the user to employ just the assembly, only the components, or both.

A gravel pack assembly is usually made up of several components, such as packers, tubing screens, gravel, etc. These components can point to the parent gravel pack assembly and have a “usage” attribute of “sand control”. This structure also allows an assembly (such as gravel pack) to change its configuration over time, such as perforating tubing, removing packers, injecting gravel, etc. This capability allows the completion data-object schema to accurately and historically portray the physical well components, if such detail is desired. If not, the schema allows only the assembly to be tracked.

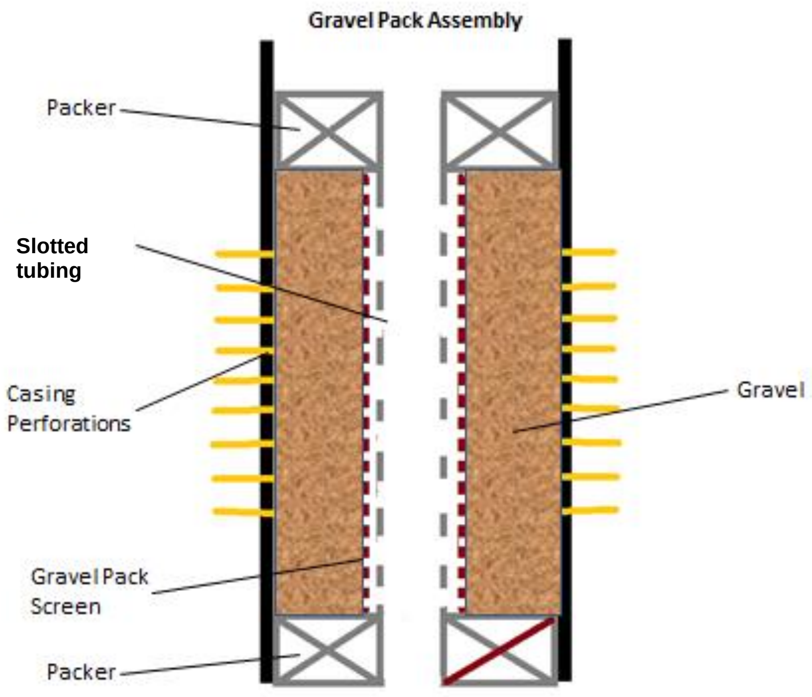


Figure 12. Main components of a gravel pack assembly.

```
<downholeString uid="gravelpackString" uidWellbore="primaryWellbore">
                    <stringType>Tubing</stringType>
      <name>GravelPack</name>
      <stringEquipmentSet>
             <stringEquipment uid="gpString_1" equipmentRefUID="topGPPcker_1">
                    <equipmentType>Packer</equipmentType>
                    <name>GPS-II Frac Pack Packer</name>
                    <length uom="ft">8</length>
                    <mdTop uom="ft">4616</mdTop>
                    <mdBottom uom="ft">4624</mdBottom>
                    <tvdTop uom="ft">4616</tvdTop>
                    <tvdBottom uom="ft">4624</tvdBottom>
                    <connectionNext uid="gpCon_1"
stringEquipmentRefUID="gpString_2">

                           <connectionForm>Mandrel</connectionForm>
                    </connectionNext>
             </stringEquipment>
             <stringEquipment uid="gpString_2" equipmentRefUID="tubingOD3.5_1">
                    <equipmentType>Tubing</equipmentType>
                    <name>Tubing OD3.5</name>
                    <length uom="ft">90</length>
                    <mdTop uom="ft">4624</mdTop>
                    <mdBottom uom="ft">4714</mdBottom>
                    <tvdTop uom="ft">4624</tvdTop>
                    <tvdBottom uom="ft">4714</tvdBottom>
                    <insideComponent/>
                    <outsideComponent/>
                    <connectionNext uid="gpCon_4"
stringEquipmentRefUID="gpString_4">

                           <connectionForm>Mandrel</connectionForm>
                    </connectionNext>
                    <perforationHole/>
             </stringEquipment>
             <stringEquipment uid="gpString_4"
equipmentRefUID="bottomGPPacker_1">
                    <equipmentType>Packer</equipmentType>
                    <name>Wesdrill Sump Packer</name>
                    <length uom="ft">8</length>
                    <mdTop uom="ft">4714</mdTop>
                    <mdBottom uom="ft">4722</mdBottom>
                    <tvdTop uom="ft">4714</tvdTop>
                    <tvdBottom uom="ft">4722</tvdBottom>
             </stringEquipment>
      </stringEquipmentSet>

EXAMPLE CONTINUES ON NEXT PAGE>
```

This list shows the components making up the gravel pack string, with detail of how connected.

CONTINUED FROM PREVIOUS PAGE

```
       <accessories>
<!--if it is not a part then the fillin can be an accessory-->
<accessory uid="fillinWellbore_1" equipmentRefUID="gpString_3">
                     <equipmentType>Fill</equipmentType>
                     <name>Gravel behind screen</name>
                     <length uom="ft">90</length>
                     <mdTop uom="ft">4624</mdTop>
                     <mdBottom uom="ft">4714</mdBottom>
                     <tvdTop uom="ft">4624</tvdTop>
                     <tvdBottom uom="ft">4714</tvdBottom>
                     <outsideString>true</outsideString>
              </accessory>
              <accessory uid="gpScreen_1"
equipmentRefUID="GravelPackerScreen_1">
                     <equipmentType>Gravel Pack Screen</equipmentType>
                     <name>Screen Mesh 8 gauge micro pack</name>
                     <length uom="ft">90</length>
                     <mdTop uom="ft">4624</mdTop>
                     <mdBottom uom="ft">4714</mdBottom>
                     <tvdTop uom="ft">4624</tvdTop>
                     <tvdBottom uom="ft">4714</tvdBottom>
                     <insideComponent>gpString_2</insideComponent>
                     <outsideComponent>gpString_1></outsideComponent>
              </accessory>
       </accessories>
       <usageComment>sand control</usageComment>
</downholeString>
```

The gravel is an accessory to a component in the string itself, ie gpString_2 (the tubing, see above).

4.2.1 Flow through Gravel Packed Interval

As explained in Section 4.1.7, page 43, a Wellbore Completion may have one of its Contact Intervals be a Gravel Pack Interval. In this case the interval will have a reference to the downhole string of which the gravel pack assembly is part.

4.3 Packers

A packer embeds itself in the interior wall of casing, forming a seal and has one or more tubing strings running through the center forming a tight seal. The complexity for representing this in the schema comes from determining with which group of wellbore components it belongs and that it can become a component belonging to more than one group.

Packers generally have one or more tubing strings passing through them. In some cases, it has a slip/seal fit, allowing the tubing to move up and down while retaining a seal. This configuration involves a seal assembly on the tubing string, which provides the smooth surface needed to pressure seal the connection to the packer. This seal assembly screws into the tubing string, so essentially no part of the packer actually connects to the string other than the seal. So it forms a seal with that tubing string but does not really belong to that tubing string.

In other cases, the packer does connect with a threaded connection into the tubing string. These threaded connections are on a section of tubing that is affixed through the center of the packer. You could consider this a permanent connection built into the packer, but it may be a mandrel that can be changed out of the packer to allow other sizes of tubing to be used. In either case, there is a joint of tubing that runs through the center of the packer and connects in line with the tubing string.

In the latter case (where there is a threaded connection) the packer can be conveyed to its set position via a tubing string or work string. While it is connected in line, it is easy to visualize it as part of that string. However, what string does it belong to if it is disconnected and left downhole? It may have a section of tubing dangling below with none above: what group or string does it belong to then? There are other strange combinations that make defining a schema that works with a packer difficult. Here we discuss the solutions which can be implemented using the completion schema.

The next sections step through a variety of scenarios concerning packers and describe how to populate the Completion data-object schema for each scenario.

4.3.1 Scenario I: Tubing-Set Packer

The most typical scenario involves a single tubing string where the packer is conveyed via the tubing string and set in production casing (Figure 13). The packer is listed in the stringEquipmentSet section of the downholeString data object for the tubing string. The in-line order and connection information is included. The connection information for the packer includes a “radial” connection to the production casing.

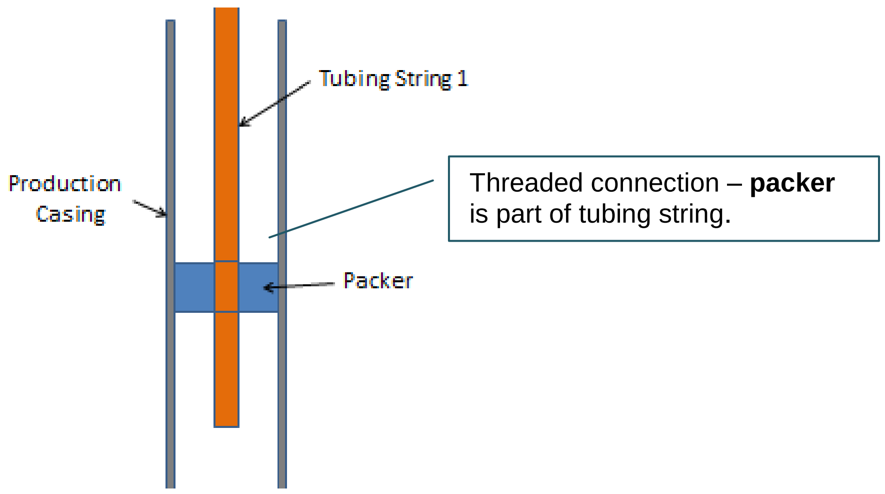


Figure 13. Scenario I: Most typical packer scenario: a single tubing string where the packer is conveyed via the tubing string and set in production casing.

```
<downholeString uid="productiontubing_1" uidWellbore="primaryWellbore">
        <stringType>Tubing</stringType>
        <name>productiontubing</name>
        <stringEquipmentSet>
                <stringEquipment uid="tubingstring_1"
equipmentRefUID="tubingtype_1">
                        <equipmentType>Tubing</equipmentType>
                        <name>tubing_for_production_casing1</name>
                        <orderOfObject>1</orderOfObject>
                        <connectionNext stringEquipmentRefUID="tubingstring_1">
                                <tubingConnectionType>Threaded</tubingConnectionType>
                        </connectionNext>
                </stringEquipment>
                <stringEquipment uid="packer_1" equipmentRefUID="packertype_1">
                        <equipmentType>Packer</equipmentType>
                        <name>packer_1</name>
                        <orderOfObject>2</orderOfObject>
                        <connectionNext stringEquipmentRefUID="productioncasing_1">
                                <otherConnectionType>DogsCompressionFit-
Sealed</otherConnectionType>
                        </connectionNext>
                </stringEquipment>
        </stringEquipmentSet>
</downholeString>
```

connectionNext used here to show the threaded connection tubing to packer.

Optional orderOfObject enables exact connection order to be defined.

connectionNext used here to show the radial connection from packer to outer casing.

4.3.2 Scenario II: Wireline-Set Packer

In Figure 14, the well diagram may look the same as Scenario I; however, the schema is considerably different. In this scenario, a packer has been set by wireline in the production casing and a seal assembly runs through the middle of the packer. Because it does not screw in line with the production casing string, the packer is an “accessory” of the production casing string (not part of with the “screwed together” tubing string). The seal assembly is part of the tubing string and is in the stringEquipmentSet section of the downholeString data object for the tubing string. In the connection information of the seal assembly, we reference the packer.

So, in Scenario I the packer is referenced in the stringEquipment section of Tubing String 1. In Scenario II it is included as an “accessory” of Tubing String 1.

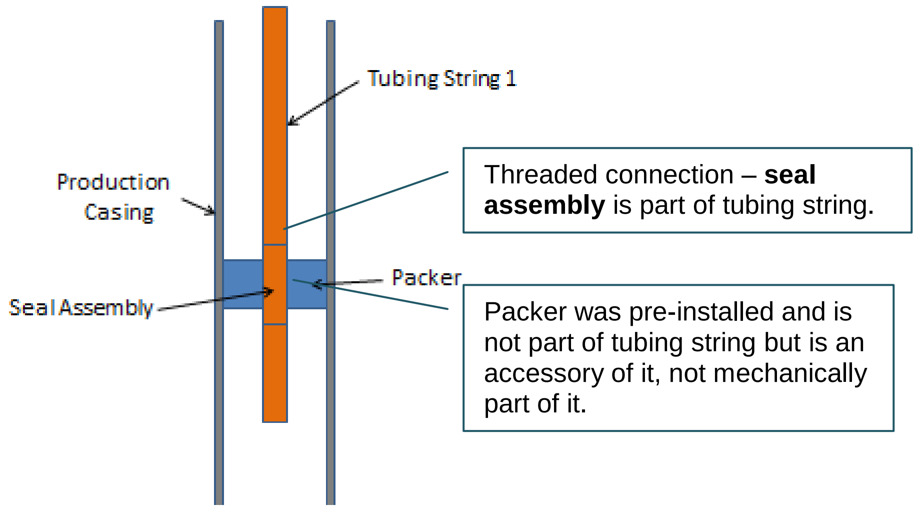

Tubing String 1
Production Casing
Threaded connection – **seal assembly** is part of tubing string.
Packer
Seal Assembly
Packer was pre-installed and is not part of tubing string but is an accessory of it, not mechanically part of it.

Figure 14. Scenario II: A packer has been set by wireline in the production casing and a seal assembly runs through the middle of the packer.

```
<downholeString uid="productiontubing_1" uidWellbore="primaryWellbore">
       <stringType>Tubing</stringType>
       <subStringType>Production tubing</subStringType>
       <name>productiontubing</name>
       <stringEquipmentSet>
              <stringEquipment uid="tubingstring_1"
equipmentRefUID="tubingtype_1">
                     <equipmentType>Tubing</equipmentType>
                     <name>tubing_for_production_casing1</name>
              </stringEquipment>
              <stringEquipment uid="sealAssembly1" equipmentRefUID=" seal
assemblytype_1">
                     <equipmentType>Seal Assembly</equipmentType>
                     <name>Seal Assembly</name>
                     <connectionNext stringEquipmentRefUID=" packer_1">
                            <tubingConnectionType>Radial</tubingConnectionType>
                     </connectionNext>
              <stringEquipment>
       </stringEquipmentSet>
       <accessories>
              <accessory uid="packer_1" equipmentRefUI
                     <equipmentType>Packer</equipmentType>
                     <name>packer_1</name>
              </accessory>
       </accessories>
</downholeString>
```

connectionNext used to show seal assembly connects radially to packer_1.

Packer is an accessory in this case.

4.3.3 Scenario III: Dual String Packer, Tubing Set on First String

This scenario (Figure 15) is a combination of Scenarios I & II. The packer and seal Assembly are listed in the Equipment Listing section. Like Scenario I, the packer is contained in Tubing String 1. The seal assembly is contained in Tubing String 2 as it is connected in line with the string. It has a radial connection to the packer, similarly to Scenario II.

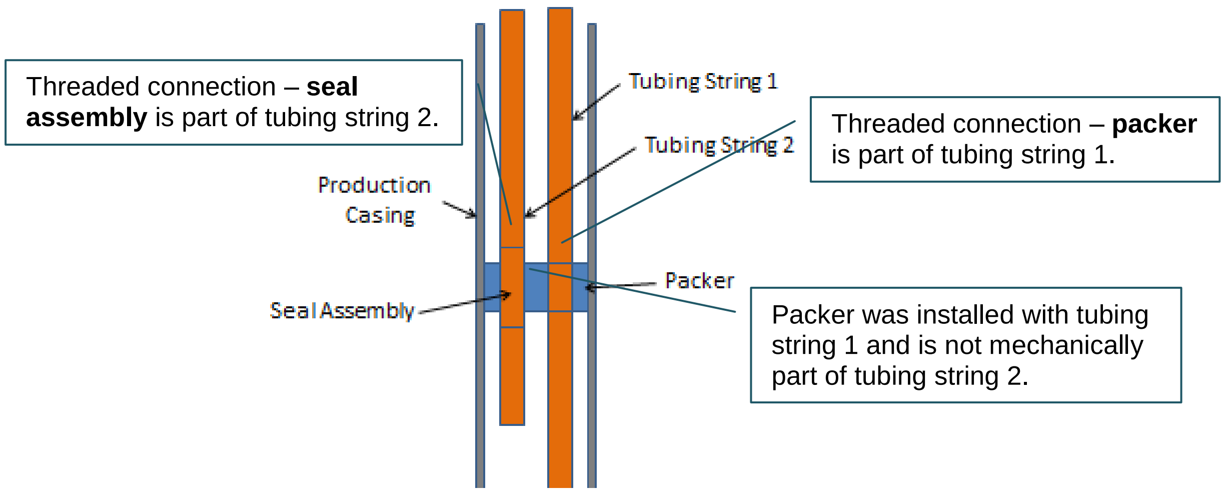

Threaded connection – **seal assembly** is part of tubing string 2.
Tubing String 1
Tubing String 2
Threaded connection – **packer** is part of tubing string 1.
Production Casing
Packer
Seal Assembly
Packer was installed with tubing string 1 and is not mechanically part of tubing string 2.

Figure 15. Scenario III: Dual string packer, tubing set on first string (a combination of Scenarios I & II).

```
<downholeString uid="productiontubing_1" uidWellbore="primaryWellbore">
       <stringType>Tubing</stringType>
       <subStringType>Production tubing</subStringType>
       <name>productiontubing</name>
       <stringEquipmentSet>
              <stringEquipment uid="tubingstring_1"
equipmentRefUID="tubingtype_1">
                     <equipmentType>Tubing</equipmentType>
                     <name>tubing_for_production_tubing1</name>
              </stringEquipment>
              <stringEquipment uid="packer_1" equipmentRefUID="packertype_1">
                     <equipmentType>Packer</equipmentType>
                     <name>packer_1</name>
              </stringEquipment>
</downholeString>
<downholeString uid="productiontubing_2" uidWellbore="primaryWellbore">
       <stringType>Tubing</stringType>
       <subStringType>Production tubing</subStringType>
       <name>productiontubing</name>
       <stringEquipmentSet>
              <stringEquipment uid="tubingstring_2"
       equipmentRefUID="tubingtype_1">
                     <equipmentType>Tubing</equipmentType>
                     <name>tubing_for_production_tubing2</name>
              </stringEquipment>
              <stringEquipment uid="sealAssembly1"
equipmentRefUID="sealassemblytype_1">
                     <equipmentType>Seal Assembly</equipmentType>
                     <name>Seal Assembly</name>
                     <connectionNext stringEquipmentRefUID="packer_1">
                            <tubingConnectionType>Radial</tubingConnectionType>
                     </connectionNext>
              <stringEquipment>
       </stringEquipmentSet>
</downholeString>
```

Tubing string 1 – Has:
Tubing...
..and packer

Note how both tubing strings can have same type of tubing so have same equipmentRefUID

Tubing string 2 – Has:
Tubing...
..and seal assembly

connectionNext used to show seal assembly on string 2 connects to packer_1 on string 1

4.3.4 Scenario IV: Packer Downhole with Tailpipe Below it

This scenario (Figure 16) could have started out as Scenario I, but then the tubing was detached from the packer leaving it set in the casing with tailpipe dangling down the wellbore. Nothing has changed from Scenario I below the packer. But now the first item in the Downhole String Tubing String is the Packer.

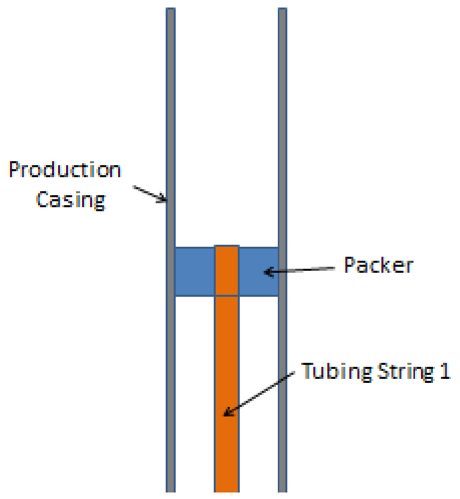


Figure 16. Scenario IV: Packer downhole with tailpipe below it.

```
<downholeString uid="productiontubing_1" uidWellbore="primaryWellbore">
       <stringType>Tubing</stringType>
       <subStringType>Production tubing</subStringType>
       <name>productiontubing</name>
       <stringEquipmentSet>
              <stringEquipment uid="packer_1" equipmentRefUID="packertype_1">
                     <equipmentType>Packer</equipmentType>
                     <name>packer_1</name>
                     <connectionNext stringEquipmentRefUID="productioncasing_1">
                            <tubingConnectionType>Radial</tubingConnectionType>
                     </connectionNext>
              </stringEquipment>
              <stringEquipment uid="tubingstring_1"
equipmentRefUID="tubingtype_1">
                     <equipmentType>Tubing</equipmentType>
                     <name>tubing_for_production_casing1</name>
              </stringEquipment>
       </stringEquipmentSet>
</downholeString>
```

4.3.5 Scenario V: Packer Downhole with Tailpipe and Separate Tubing with Packer Up Hole

Adding on to Scenario IV, another tubing string with packer is installed above the equipment (Figure 17), which allows production to flow through the bottom string, into the production casing annulus, and into the top tubing string. The original Scenario IV and the new Scenario V tubing strings are treated as separate physical strings. One definition of "string" is that when you pull on the top piece, everything attached that comes with it is part of that string.

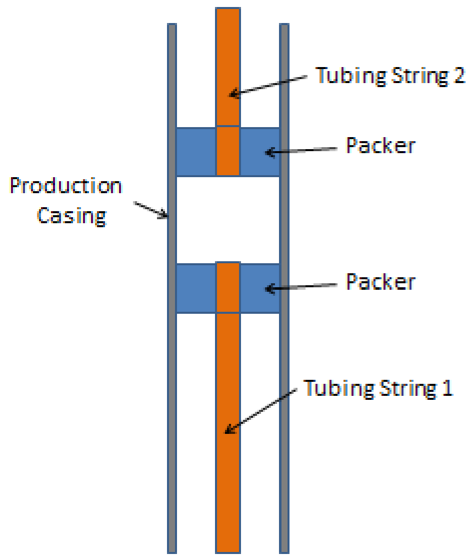


Figure 17. Scenario V (an extension to Scenario IV): Packer downhole with tailpipe and separate tubing with packer up hole.

```
<downholeString uid="productiontubing_1" uidWellbore="primaryWellbore">
       <stringType>Tubing</stringType>
       <subStringType>Production tubing</subStringType>
       <name>productiontubing_1</name>
       <stringEquipmentSet>
              <stringEquipment uid="packer_1" equipmentRefUID="packertype_1">
                     <equipmentType>Packer</equipmentType>
                     <name>packer_1</name>
                     <connectionNext stringEquipmentRefUID="productioncasing_1">
                            <tubingConnectionType>Radial</tubingConnectionType>
                     </connectionNext>
              </stringEquipment>
              <stringEquipment uid="tubingstring_1"
equipmentRefUID="tubingtype_1">
                     <equipmentType>Tubing</equipmentType>
                     <name>tubing_for_production_tubing1</name>
              </stringEquipment>
       </stringEquipmentSet>
</downholeString>
<downholeString uid="productiontubing_2" uidWellbore="primaryWellbore">
       <stringType>Tubing</stringType>
       <subStringType>Production tubing</subStringType>
       <name>productiontubing_2</name>
       <stringEquipmentSet>
              <stringEquipment uid="tubingstring_2"
equipmentRefUID="tubingtype_1">
                     <equipmentType>Tubing</equipmentType>
                     <name>tubing_for_production_tubing2</name>
              </stringEquipment>
              <stringEquipment uid="packer_2" equipmentRefUID="packertype_1">
                     <equipmentType>Packer</equipmentType>
                     <name>packer_2</name>
                     <connectionNext stringEquipmentRefUID="productioncasing_1">
                            <tubingConnectionType>Radial</tubingConnectionType>
                     </connectionNext>
              </stringEquipment>
       </stringEquipmentSet>
</downholeString>
```

Two downhole strings. Both shown with same equipment types.

4.3.6 Scenario VI: Isolated Packer in Casing

In this scenario (Figure 18), a packer is set by wireline in production casing. No tubing is attached above or below. The center of the packer is open allowing flow (not sealing anything). The packer listed in the stringEquipmentSet section of the downholeString data object for the casing string as an accessory of production casing and radially connected.

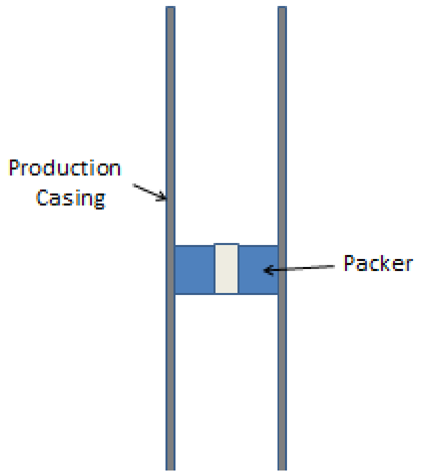


Figure 18. Scenario VI: Isolated packer in casing.

```
<downholeString uid="productioncasing_1" uidWellbore="primaryWellbore">
       <stringType>Casing</stringType>
       <subStringType>Production casing</subStringType>
       <name>productioncasing</name>
       <accessories>
              <accessory uid="packer_1" equipmentRefUID=" productioncasing_1">
                     <equipmentType>Packer</equipmentType>
                     <name>packer_1</name>
                     <insideComponent></insideComponent>
                     <connectionNext>
       <otherConnectionType>DogsCompressionFit-Sealed</otherConnectionType>
                     <stringEquipmentRefUID>productioncasing_1
                     </stringEquipmentRefUID>
                     </connectionNext>
              </accessory>
```

4.3.7 Scenario VII: Bridge Plug/Packer with Plug

This scenario (Figure 19) is exactly the same as Scenario VI, except there is a plug in the packer, which makes it fundamentally a bridge plug.

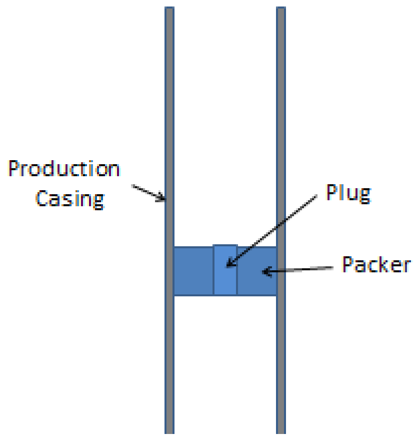


Figure 19. Scenario VII: Bridge plug/packer with plug.

```
<downholeString uid="productioncasing_1" uidWellbore="primaryWellbore">
       <stringType>Casing</stringType>
       <subStringType>Production casing</subStringType>
       <name>productioncasing</name>
       <accessories>
              <accessory uid="packer_1" equipmentRefUID=" productioncasing_1">
                     <equipmentType>Packer</equipmentType>
                     <name>packer_1</name>
              </accessory>
              <accessory uid="Packerplug_1" equipmentRefUID=" packer_1">
                     <equipmentType>Packer Plug</equipmentType>
                     <name>packerplug_1</name>
              </accessory>
       </accessories>
</downholeString>
```

Connections and inside component elements omitted.

4.4 Parent and Child "Jobs" and Events

This section shows the simple way in which events can be nested into "jobs" comprising multiple events (or stages). Examples could be a daily aggregation of activity, or the report of a service company doing a multi-stage job.

4.4.1 Parent Event

```
       <witsml:wellCMLedger uid="E00" uidWell="VGASAU029"
uidWellbore="primaryWellbore">
              <witsml:wellboreName>primaryWellbore</witsml:wellboreName>
              <witsml:dTimStart>1990-11-11T00:00:00Z</witsml:dTimStart>
              <witsml:dTimEnd>1990-11-22T00:00:00Z</witsml:dTimEnd>
              <witsml:duration uom="h">47</witsml:duration>
              <witsml:type class="Job">Drilling &amp; Complete Well </witsml:type>
              <witsml:isPlan>true</witsml:isPlan>
              <witsml:eventExtension>
                     <witsml:jobExtension>
                            <jobReason>drilling init</jobReason>
                     </witsml:jobExtension>
              </witsml:eventExtension>
       </witsml:wellCMLedger>
```

This event has no parent event. Could be called something such as, e.g., "Drill & Complete Well".

4.4.2 Children Events

Example, Event 08 is "drill – day 1"; parentEvent element points to the job "Drilling & Complete Well".

```
       <witsml:wellCMLedger uid="E08" uidWell="VGASAU029" uidWellbore="primaryWellbore">
              <witsml:wellboreName>primaryWellbore</witsml:wellboreName>
              <witsml:parentEventID>E00</witsml:parentEventID>
              <witsml:dTimStart>1990-11-12T08:00:00Z</witsml:dTimStart>
              <witsml:dTimEnd>1990-11-12T14:00:00Z</witsml:dTimE
              <witsml:duration uom="h">6</witsml:duration>
              <witsml:mdTop uom="ft">12</witsml:mdTop>
              <witsml:mdBottom uom="ft">450</witsml:mdBottom>
              <witsml:type>drill - day 1</witsml:type>
              <witsml:isPlan>true</witsml:isPlan>
              <downholeStringsRef downholeStringRefID="surfacecasing_1"/>
              <witsml:eventExtension>
                     <witsml:downholeExtension>
              <downholeComponentRefID>downholecomp_Job1</downholeComponentRefID>
                     </witsml:downholeExtension>
              </witsml:eventExtension>
       </witsml:wellCMLedger>
```

This event has a parent event. Drill & Complete Well ("E00" here) is the parent for a set of events including this one.

4.5 Loader File Extensibility

A loader file is used to define equipment "extended properties" for the following purposes:

- To define the "extended properties" of equipment types beyond the common properties that all equipment has. Example: OD is common to all types of equipment wheras diameter of sand may only apply to a gravel pack screen.
- To support extension of the equipment properties beyond those available in the standard Energistics release.

For the overview of the equipment types and properties, see Section 2.2.1.4, page 18. For a list of equipment types, also showing those with extended properties in the loader file, see Appendix A. Equipment Types and Properties, page 60.

4.5.1 Use of Properties in the Loader File

The loader file is an XML file, EquipmentDict.xml, controlled by a schema cs_xmlEquipmentProperties.xsd. Both these files are located in the *ancillary* folder of the schema set. The content of the loader file is an equipmentPropertySet comprising an equipmentProperties element for each Equipment Type whose properties are being extended. Note, the common properties for all equipment types are defined in the schema, in the grp_equipmentProperty.xsd schema file.

The equipmentProperties element comprises an equipment element (which must be one of the types in the equipmentType enum), and any number of property elements.

For example: the loader file contains the extended properties about the gravel pack screen, including the definitions for “diaSand” and “diaScreenGauge”.

```
<witsml:equipmentProperties>
              <witsml:equipment>Gravel Pack Screen</witsml:equipment>
              <witsml:property name="diaSand">
                     <witsml:type>lengthMeasure</witsml:type>
                     <witsml:isOptional>true</witsml:isOptional>
              </witsml:property>
              <witsml:property name="diaScreenGauge">
                     <witsml:type>lengthMeasure</witsml:type>
                     <witsml:isOptional>true</witsml:isOptional>
              </witsml:property>
…
</witsml:equipmentProperties>
```

These extended properties which are defined this way in the loader file can appear as property elements in the Equipment (in cs_equipment.xsd). See Figure 20.

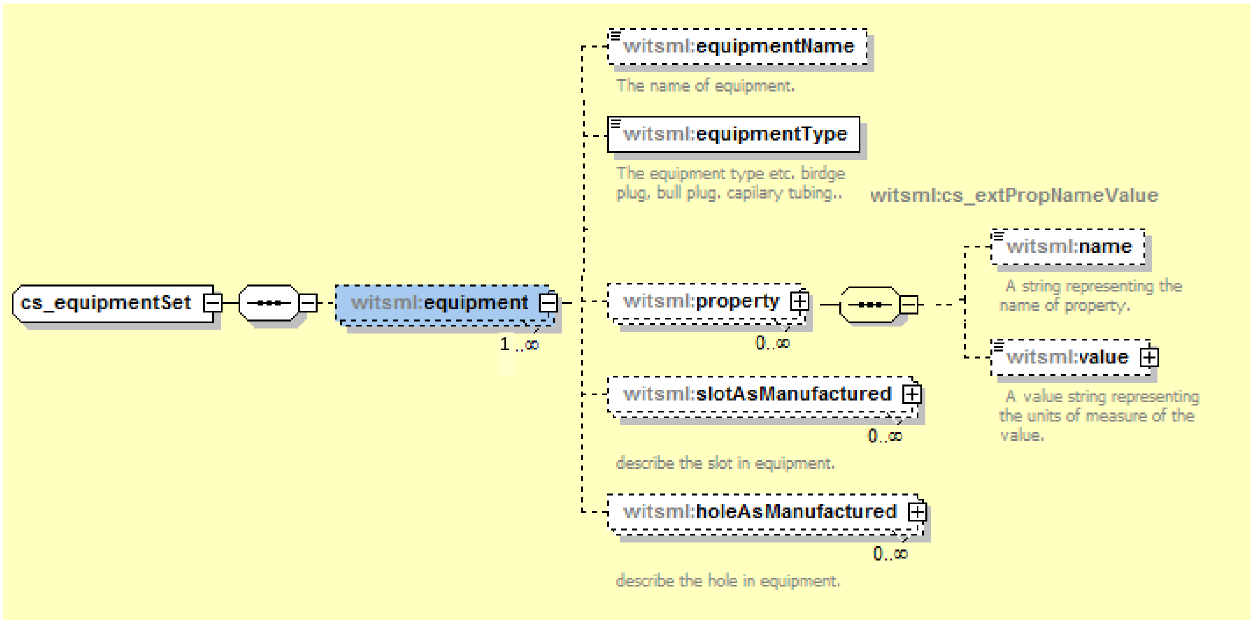

witsml:equipmentName
The name of equipment.
witsml:equipmentType
The equipment type etc. birdge plug, bull plug, capilary tubing..
witsml:cs_extPropNameValue
witsml:name
A string representing the name of property.
cs_equipmentSet
witsml:equipment
1..∞
witsml:property
0..∞
witsml:value
A value string representing the units of measure of the value.
witsml:slotAsManufactured
0..∞
describe the slot in equipment.
witsml:holeAsManufactured
0..∞
describe the hole in equipment.

Figure 20. Schema structure for accessing the loader file.

Taking our example of the gravel pack screen, in the downholeComponent xml, we can list the element “diaScreenGauge” in the property section under the equipment “GravelPackScreen_1”. The type is defined as “lengthMeasure”, so we can apply a value and unit measurement (“in”).

```
<witsml:equipment uid="GravelPackScreen_1">
             <witsml:equipmentName>GravelPackScreen_1</witsml:equipmentName>
             <witsml:equipmentType>Gravel Pack Screen</witsml:equipmentType>
...
             <witsml:property>
                    <witsml:name>diaScreenGauge</witsml:name>
                    <witsml:value uom="in">8</witsml:value>
             </witsml:property>
...
</witsml:equipment>
```

Note however that there is no validation mechanism for the XML document which would be created by inserting *property* elements to the *equipment* element. This is because the loader file is not incorporated into the schema set. An application requiring to validate the *property* elements would need to add logic to ensure the content of these elements does match the data specified in the loader file.

4.5.2 Extensibility

All Energistics standards (WITSML, PRODML and RESQML) use loader files to define valid values. This loader file has been developed to provide both consistency (for maximum interoperability) and flexibility for future needs.

The downholeComponent data-object includes various kinds of equipment and each might have different attributes according to different manufacturers, which will also change over time. To address this type of natural change, the loader file is used with the schemas to extend the equipment properties.

For flexibility and future needs, organizations may add new properties to equipment types which are defined in the schema. This is done by editing the loader file, *EquipmentDict.xml*, in accordance with its schema *cs_xmlEquipmentProperties.xsd*. Any software which uses the loader file to define expected extended values for equipment should then work with the modified loader file.

However, to maintain maximum interoperability across the standards, Energistics strongly recommends that organizations adding such new extensions submit these extensions back to Energistics for inclusion in the released loader file distributed to users.

Appendix A. Equipment Types and Properties

Note: These printed tables are provided for easy reference. For the most current information, always see the files referred to here. The first table lists all the equipment types defined (in the enum of equipmentTypes in the cs-equipmentCatalog.xsd schema file). It shows if these have extended properties defined in the loader file (EquipmentDict.xml in the ancillary folder). The second table lists the common equipment properties in the schema, i.e., those common to all equipment, and to which the loader file extended properties are added. These common properties are found in the grp_equipmentProperty.xsd schema file.

Equipment Type	**Extended Properties**
Bridge Plug	
Bull Plug	
Capillary Tubing	
Casing Crossover	yes
Casing Hanger	
Casing Head	yes
Casing Liner-Expandable	
Casing Shoe	yes
Casing Spool	yes
Casing/Casing Liner	yes
Cement (behind Casing)	
Cement Basket	
Cement Retainer	
Cement Squeeze	
Cement Stage Tool	
Chemical Injection Mandrel	yes
Chemical Injection Valve	yes
Corrosion Coupon Carrier	yes
Dip Tube	yes
Downhole Choke	
Downhole Sensor	yes
ESP Assembly	yes
ESP Bolt on Discharge	
ESP Bolt on Intake	yes
ESP Bolt on Motor Base	yes
ESP Bolt on Motor Head	yes
ESP Cable	yes
ESP Gas Handler	yes
ESP Gas Separator	yes
ESP Lower Pigtail	yes
ESP Motor	yes
ESP Motor Base Centralizer	
ESP Motor Flat Cable	yes
ESP Motor Shroud	
ESP Promotor	yes
ESP Pump	yes
ESP Pump Discharge Sensor Sub	yes
ESP Seal	yes
Expansion Joint	
External Cementing Port	
Fill	
Fish	
Float Collar	
Float Shoe/Guide Shoe	
Gas Anchor	
Gas Lift Mandrel	yes
Gas Lift Valve	yes
Gravel Pack Screen	yes
Hydraulic Pump	
Injection Mandrel	yes
Injection Valve	yes
Junk in Wellbore	
Landing Collar	
Liner Entry Guide	
Liner Hanger	yes
Mule Shoe	
Notched Collar	
On-Off Tool	
Overshot	yes
Packer	yes
Packer-Multiple Strings	
Packer Plug	
Packoff (Tubing)	
PCP-Flex shaft Intake	

Equipment Type	Extended Properties
PCP-Gear Reducer (Subsurface)	
PCP-No Turn Tool/Torque Anchor	yes
PCP-Rotor	yes
PCP-Stator	yes
PCP-Tag Bar	yes
Plug - Cement	
Plug - Mud	
Plunger Lift Ball	
Plunger Lift Bottom Hole Bumper Assembly	
Plunger Lift Bumper Spring	
Plunger Lift Collar Stop	
Plunger Lift Plunger	
Polished Rod	
Polished Rod Liner	yes
Ported Collar	
Profile Nipple	yes
Profile Nipple Plug	
Pump-Out Plug	
Seal Assembly	
Sucker Rod	
Sucker Rod Backoff Coupling	
Sucker Rod-Continuous	yes
Sucker Rod Pump-Insert	yes
Sucker Rod Pump-Jacket	
Sucker Rod Pump-Tubing Pump Barrel	yes
Sucker Rod Pump-Tubing Pump Plunger	yes
Sucker Rod-Ribbon	
Sucker Rod-Sinker Bar	
Sucker Rod Sub	
Sand Screen-Tubing	yes
Sand Separator	
Screen Liner/Insert	
Seal Bore Extension	
Seat Nipple/Shoe	
Shear Tool	yes
Sliding Sleeve	
Steam Cup Mandrel	yes
Steam Deflectors	
Strainer Nipple	

Equipment Type	Extended Properties
Subsurface Safety Valve	yes
TCP Gun	
Tubing	
Tubing (Coiled)	
Tubing Anchor/Catcher	
Tubing Crossover	
Tubing Drain	
Tubing Hanger	
Tubing Head (Spool)	yes
Tubing Purge Check Valve	
Tubing Sub	
Whipstock	
Wireline Re-Entry Guide (Bell Collar)	
Wellbore Notes	yes
Y-Tool	

List of Properties Common to all Equipment Types		
manufacturer		model
catalogID		catalogName
brandName		modelType
series		isSerialized
serialNumber		partNo
surfaceCondition		material
grade		unitWeight
coatingLinerapplied		outsideCoating
insideCoating		unitLength
majorOD		minorOD
OD		MaxOD
MinOD		majorID
minorID		ID
MaxID		MinID
drift		nominalSize
nameService		description
descriptionPermanent		remark