
PacBioFileFormats Documentation

Release 11.0.0

Pacific Biosciences

Feb 02, 2022

Contents

1	Brief primer and lexicon for PacBio SMRT sequencing	3
2	BAM format specification for PacBio	5
3	PacBio BAM index file (<code>bam.pbi</code>) format	17
4	FASTA file format	23
5	DataSet format specification	25
6	Run Design CSV specification for PacBio	53
7	Legacy formats	59
8	APIs available	75
9	Data Model XSD	77

As of the 3.0 release of SMRTanalysis, PacBio is embracing the industry standard BAM format for (both aligned and unaligned) basecall data files. We have also formulated a BAM companion file format (*bam.pbi*) enabling fast access to a richer set of per-read information as well as compatibility for software built around the legacy *cmp.h5* format.

Brief primer and lexicon for PacBio SMRT sequencing

PacBio SMRT sequencing operates within a silicon chip (a **SMRTcell**) fabricated to contain a large number of microscopic holes (**ZMWs**, or **zero-mode waveguides**), each assigned a **hole number**.

Within a ZMW, PacBio SMRT sequencing is performed on a circularized molecule called a **SMRTbell**. The SMRTbell, depicted below, consists of:

- the customer's double-stranded DNA **insert** (with sequence I , read following the arrow)
- (optional) double-stranded DNA **barcodes** (sequences B_L, B_R) used for multiplexing DNA samples. While the barcodes are optional, they must be present at both ends if present at all. Barcodes may or may not be *symmetric*, where symmetric means $B_L = B_R^{RC}$.
- SMRTbell **adapters** (sequences A_L, A_R), each consisting of a double stranded stem and a single-stranded hair-pin loop. Adapters may or may not be *symmetric*, where symmetric means $A_L = A_R$.

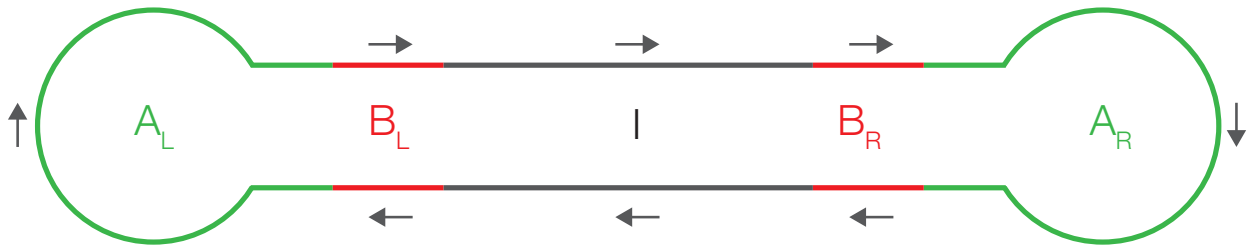


Fig. 1: A schematic drawing of a SMRTbell

SMRT sequencing interrogates the incorporated bases in the product strand of a replication reaction. Assuming the sequencing of the template above began at START, the following sequence of bases would be incorporated (where we are using the superscripts C, R, and RC to denote sequence complementation, reversal, and reverse-complementation):

$$A_L^C B_L^C I^C B_R^C A_R^C B_R^R I^R B_L^R A_L^C \dots$$

(note the identity $(x^{RC})^C = x^R$).

The **ZMW read** is the full output of the instrument/basecaller upon observing this series of incorporations, subject to errors due to optical and other limitations. **Adapter regions** and **barcode regions** are the spans of the ZMW read

corresponding to the adapter and barcode DNA. The **subreads** are the spans of the ZMW read corresponding to the DNA insert.

One complication arises when one considers the possibility that a ZMW might not contain a single sequencing reaction. Indeed it could contain zero—in which case the ensuing basecalls are a product of background noise—or it could contain more than one, in which case the basecall sequence represents two intercalated reads, effectively appearing as noise. To remove such noisy sequence, the **high quality (HQ) region finder** in PostPrimary algorithmically detects a maximal interval of the ZMW read where it appears that a single sequencing reaction is taking place. This region is designated the **HQ region**, and in the standard mode of operation, PostPrimary will only output the subreads detected within the HQ region.

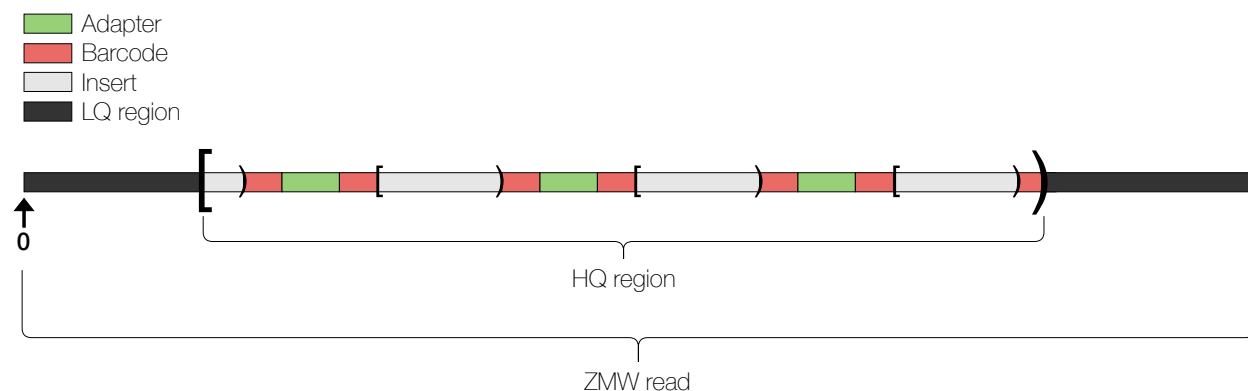


Fig. 2: A schematic of the regions designated within a ZMW read

Note: Our coordinate system begins at the first basecall in the ZMW read (deemed base 0)—i.e., it is *not* relative to the HQ region. Intervals in PacBio reads are given in end-exclusive (“half-open”) coordinates. This style of coordinate system should be familiar to Python or C++ STL programmers.

1.1 BAM everywhere

Unaligned BAM files representing the *subreads* will be produced natively by the PacBio instrument. The subreads BAM will be the starting point for secondary analysis. In addition, the artifacts resulting from removing adapter and LQ region sequences will be retained in a `scraps.bam` file.

The circular consensus tool/workflow (CCS) will take as input an unaligned subreads BAM file and produce an output BAM file containing unaligned *consensus* reads.

Alignment (mapping) programs take these unaligned BAM files as input and will produce *aligned* BAM files, faithfully retaining all tags and headers.

BAM format specification for PacBio

The BAM format is a binary, compressed, record-oriented container format for raw or aligned sequence reads. The associated SAM format is a text representation of the same data. The [specifications for BAM/SAM](#) are maintained by the SAM/BAM Format Specification Working Group.

PacBio-produced BAM files are fully compatible with the BAM specification. In this document we describe the way we make use of the extensibility mechanisms of the BAM specification to encode PacBio-specific information, as well as conventions we adhere to.

An example file adhering to this specification will be maintained in the *pbcore* Python library.

2.1 Version

The PacBio BAM specification version described here is 5.0.0. PacBio BAM files adhering to this spec contain the tag `pb:5.0.0` in the `@HD` header.

2.2 Coordinate conventions

The BAM format uses a 0-based coordinate system to refer to positions and intervals on the reference.

PacBio also uses a 0-based coordinate system to refer to positions and intervals within sequence reads. Positions in PacBio reads are reckoned from the first ZMW read base (as base 0), *not* the first base in the HQ region.

Perhaps confusingly, the text SAM format uses 1-based coordinate system.

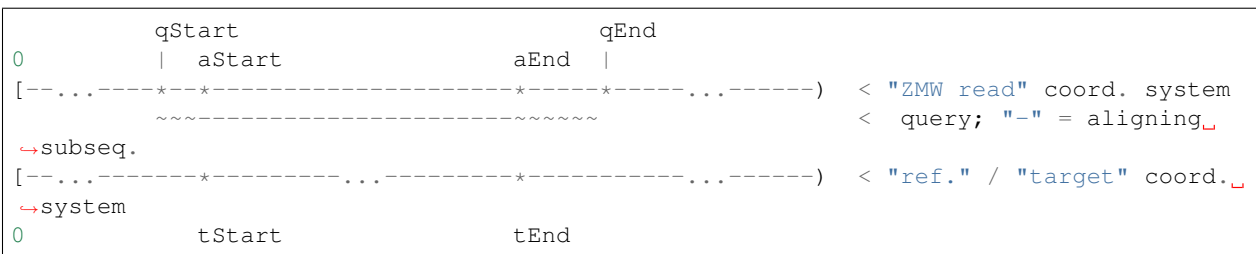
Note that following the SAM/BAM specification, 0-based coordinate intervals are defined as half-open (end exclusive) while 1-based intervals are closed.

2.3 Query versus *aligned query* terminology

A sequence read presented to an aligner is termed a *query*; typically this query will be a subsequence of an entire PacBio ZMW read—most commonly, it will be a *subread*, which is basecalls from a single pass of the insert DNA molecule.

Upon alignment, generally only a subsequence of the query will align to the reference genome, and that subsequence is referred to as the *aligned query*. Under *soft-clipping*, the entirety of the query is stored in the aligned BAM, but the CIGAR field indicates that some bases at either end are excluded from the alignment.

Abstractly, we denote the extent of the *query* in ZMW read as $[qStart, qEnd)$ and the extent of the aligned subinterval as $[aStart, aEnd)$. The following graphic illustrates these intervals:



In our BAM files, the $qStart$, $qEnd$ are contained in the qs and qe tags, (and reflected in the $QNAME$); the bounds of the *aligned query* in the ZMW read can be determined by adjusting qs and qe by the number of soft-clipped bases at the ends of the alignment (as found in the CIGAR).

Note: In the legacy cmp.h5 file format, soft-clipping was not possible, and the bounds of the original query were not stored. Only $aStart$, $aEnd$ were stored, although in that file format they were referred to as $rStart$, $rEnd$.

2.4 QNAME convention

By convention the $QNAME$ (“query template name”) for unrolled reads and subreads is in the following format:

```
{movieName}/{holeNumber}/{qStart}_{qEnd}
```

where $[qStart, qEnd)$ is the 0-based coordinate interval representing the span of the *query* in the ZMW read, as above.

For CCS reads, the $QNAME$ convention is:

```
{movieName}/{holeNumber}/ccs
```

In “by-strand mode” the $QNAME$ for CCS reads may include a suffix *fwd* or *rev* to convey strand information:

```
{movieName}/{holeNumber}/ccs/fwd
{movieName}/{holeNumber}/ccs/rev
```

2.5 CIGAR conventions

The “M” CIGAR op (BAM_CMATCH) is *forbidden* in PacBio BAM files. PacBio BAM files use the more explicit ops “X” (BAM_CDIFF) and “=” (BAM_CEQUAL). PacBio software will abort if BAM_CMATCH is found in a CIGAR field.

2.6 BAM filename conventions

Since we will be using BAM format for different kinds of data, we will use a `suffix.bam` filename convention:

Data type	Filename template
ZMW reads from movie	<i>movieName.zmws.bam</i>
Analysis-ready subreads ¹ from movie	<i>movieName.subreads.bam</i>
Excised adapters, barcodes, and rejected sub-reads	<i>movieName.scraps.bam</i>
CCS reads computed from movie	<i>movieName.ccs.bam</i>
Aligned subreads in a job	<i>jobID.aligned_subreads.bam</i>
Aligned CCS in a job	<i>jobID.aligned_ccs.bam</i>

¹ Data in a `subreads.bam` file should be `analysis ready`, meaning that all of the data present is expected to be useful for down-stream analyses. Any subreads for which we have strong evidence will not be useful (e.g. double-adapter inserts, single-molecule artifacts) should be excluded from this file and placed in `scraps.bam` as a `Filtered` with an SC tag of F.

2.7 BAM sorting conventions

Aligned PacBio BAM files shall be sorted by position in the standard fashion as done by `samtools sort`. The `BAM @HD : : SO` tag shall be set to `coordinate`.

Unaligned PacBio BAM files shall be sorted by `QNAME`, so that all subreads from a ZMW hole are stored contiguously in a file, with groups by ZMW hole number in numerical order, and within a ZMW, numerically by `qStart`. In case subreads and CCS reads are combined in a BAM, the CCS reads will sort after the subreads (`ccs` follows `{qStart}_{qEnd}`). Note that this sorting is not strictly alphabetical, so we shall set the `BAM @HD : : SO` tag to `unknown`.

2.8 Use of headers for file-level information

Beyond the usual information encoded in headers that is called for SAM/BAM spec, we encode special information as follows.

@RG (read group) header entries:

ID tag (identifier): contains an 8-character string interpretable as the hexadecimal representation of an integer. Optionally, a read group identifier may contain barcode labels to distinguish demultiplexed samples. Read groups should have distinct ID values.

Note: Standard read group identifiers for PacBio data are calculated as follows:

```
RGID_STRING := md5(movieName + "/" + readType)[:8]
```

where *movieName* is the moviename (@RG::PU) and *readType* is the read type (found in @RG::DS). Note that *movieName* is lowercase while *readType* is uppercase. *md5* is understood to be the (lowercase) hex md5 digest of the input string.

Optionally for *readType* CCS, strandness can be encoded in the ID. This is to ensure that multiple types of reads, double- and single- stranded, can be stored in the same BAM file, without hole number collisions in the PacBio BAM index file. The RGID_STRING is then defined as:

```
RGID_STRING := md5(movieName + "/" + readType + "/" + strand)[:8]
```

where strand must be lowercase fwd or rev; it may not be empty.

The RGID_INT is defined as:

```
RGID_INT := int32.Parse(RGID_STRING)
```

RGID_STRING is used in the @RG header and in the RG tag of BAM records, while RGID_INT is used in the PacBio BAM index file.

Note that RGID_INT may be negative.

Example: CCS reads for a movie named “movie32” would have

- RGID_STRING = “f5b4ffb6”
- RGID_INT = -172687434

Optional barcode labels must be appended to the RGID_STRING as follows:

```
{RGID_STRING}/{bcForward}--{bcReverse}
```

where the bcForward and bcReverse labels correspond to the 0-based positions in the FASTA file of barcodes. These are the same values used to populate a barcoded record’s bc tag.

PL tag (“platform”): contains "PACBIO".

PM tag (“platform model”): contains "ASTRO", "RS", or "SEQUEL", reflecting the PacBio instrument series.

PU tag (“platform unit”): contains the PacBio movie name.

LB tag (“Well Sample Name”): contains the user-supplied name of the library.

SM tag (“Bio Sample Name”): contains the user-supplied name of the biological sample.

BC tag (“barcodes”): contains the barcode sequences associated with this read group. This tag is not required in all PacBio BAM files, but must be provided when the read group ID includes barcode labels.

The value must be represented in the format recommended by the SAM/BAM spec. Barcode *sequences* will be concatenated by a single dash. If both barcodes are the same, only one needs to be provided.

```
{seq} {seq1}-{seq2}
```

Note that this differs from the format used to label barcode indices on a read group’s ID.

DS tag (“description”): contains some semantic information about the reads in the group, encoded as a semicolon-delimited list of “Key=Value” strings, as follows:

Mandatory items:

Key	Value spec	Value example
READTYPE	One of ZMW, HQREGION, SUB-READ, CCS, SCRAP, or UNKNOWN	SUBREAD
BINDINGKIT	Binding kit part number	100236500
SEQUENCINGKIT	Sequencing kit part number	001558034
BASECALLERVERSION	Basecaller version number	2.1
FRAMERATEHZ	Frame rate in Hz	100
CONTROL	TRUE if reads are classified as spike-in controls, otherwise CONTROL key is absent	TRUE
STRAND	Stores strandness of single-stranded reads as FORWARD or REVERSE. Key is absent if reads are double-stranded. Only applies to READTYPE CCS.	FORWARD

Note: The READTYPE values encountered in secondary analysis will be limited to SUBREAD and CCS. The remaining READTYPE values will only be encountered in intermediate steps before secondary analysis.

Base feature manifest—absent item means feature absent from reads:

Key	Value spec	Value example
DeletionQV	Name of tag used for DeletionQV	dq
DeletionTag	Name of tag used for DeletionTag	dt
InsertionQV	Name of tag used for InsertionQV	iq
MergeQV	Name of tag used for MergeQV	mq
SubstitutionQV	Name of tag used for SubstitutionQV	sq
SubstitutionTag	Name of tag used for SubstitutionTag	st
Ipd:Frames	Name of tag used for IPD, in raw frame count.	ip
Ipd:CodecV1	Name of tag used for IPD, compressed according to Codec V1.	ip
PulseWidth:Frames	Name of tag used for PulseWidth, in raw frame count.	pw
PulseWidth:CodecV1	Name of tag used for PulseWidth, compressed according to Codec V1.	pw

Optional items:

Note: These items are optional if there are no “bc” tags in the reads belonging to this read-group, otherwise they are mandatory.

Key	Value spec	Value example
Bar-codeFile	Name of the Fasta file containing the sequences of the barcodes used	pacbio_384_barcodes.fasta
Bar-code-Hash	The MD5 hash of the contents of the barcoding sequence file, as generated by the <i>md5sum</i> commandline tool	0a294bb959fc6c766967fc8beeb4d88d
Bar-code-Count	The number of barcode sequences in the Barcode File	384
Bar-code-Mode	Experimental design of the barcodes Must be Symmetric/Asymmetric/Tailed or None	Symmetric
BarcodeQuality	The type of value encoded by the bq tag Must be Score/Probability/None	Probability

2.9 Use of read tags for per-read information

Tag	Type	Description
qs	i	0-based start of query in the ZMW read (absent in CCS)
qe	i	0-based end of query in the ZMW read (absent in CCS)
ws	i	Start of first base of the query ('qs') in approximate raw frame count since start of movie. For a CCS read, the start of the first base of the first incorporated subread.
we	i	Start of last base of the query ('qe - 1') in approximate raw frame count since start of movie. For a CCS read, the start of the last base of the last incorporated subread.
zm	i	ZMW hole number
np	i	NumPasses (1 for subreads, variable for CCS—encodes number of <i>complete</i> passes of the insert)
ec	f	Effective coverage for CCS reads, the average subread coverage across all windows (only present in CCS reads)
rq	f	Float in [0, 1] encoding expected accuracy
sn	B,f	4 floats for the average signal-to-noise ratio of A, C, G, and T (in that order) over the HQRegion

2.10 Use of read tags for per-read-base information

The following read tags encode features measured/calculated per-basecall. Unlike *SEQ* and *QUAL*, aligners will not orient these tags. They will be maintained in *native* orientation (in the same order and sense as collected from the instrument) even if the read record has been aligned to the reverse strand.

Tag	Type	Description
dq	Z	DeletionQV
dt	Z	DeletionTag
iq	Z	InsertionQV
mq	Z	MergeQV
sq	Z	SubstitutionQV
st	Z	SubstitutionTag
ip	B,C or B,S	IPD (raw frames or codec V1)
pw	B,C or B,S	PulseWidth (raw frames or codec V1)

Notes:

- QV metrics are ASCII+33 encoded as strings
- *DeletionTag* and *SubstitutionTag* represent alternate basecalls, or “N” when there is no alternate basecall available. In other words, they are strings over the alphabet “ACGTN”.
- The IPD (interpulse duration) value associated with a base is the number of frames *preceding* its incorporation, while the PW (pulse width) is the number of frames during its incorporation.
- Encoding of kinetics features (*ip*, *pw*) is described below.

2.11 Use of read tags for HiFi per-read-base kinetic information

The following read tags contain averaged kinetic information (IPD/PulseWidth) from subreads when applying CCS to generate HiFi reads. These are computed and stored independently for both orientations of the insert. Forward is defined & stored with respect to the orientation represented in *SEQ* and is considered to be the native orientation. Reverse tags are stored in the opposite direction, e.g. from the last base to the first. As with other PacBio-specific tags, aligners will not re-orient these fields.

Tag	Type	Description
fi	B,C	Forward IPD (codec V1)
ri	B,C	Reverse IPD (codec V1)
fp	B,C	Forward PulseWidth (codec V1)
rp	B,C	Reverse PulseWidth (codec V1)
fn	i	Forward number of complete passes (zero or more)
rn	i	Reverse number of complete passes (zero or more)

The following clipping example illustrates the coordinate system for these tags, shown as stored in the BAM file:

```

-----
Original
-----

  SEQ:  A  A  C  C  G  T  T  A  G  C
fi/fp: f0, f1, f2, f3, f4, f5, f6, f7, f8, f9
ri/rp: r9, r8, r7, r6, r5, r4, r3, r2, r1, r0

-----
Clipped to [1, 4)
-----

  SEQ:  A  C  C
fi/fp: f1, f2, f3
ri/rp: r3, r2, r1

```

Notes:

- When CCS filtering is disabled, no averaging occurs with ZMWs that don’t have enough passes to generate HiFi reads. Instead, the *pw/ip* values are passed as is from a representative subread.
- Minor cases exist where a certain orientation may get filtered out entirely from a ZMW, preventing valid values from being passed for that record. In these cases, empty lists will be passed for the respective record/orientation and number of passes will be set to zero.

- Flanking zeroes in kinetics arrays should be ignored for the respective strand. For instance, when SEQ is AAACGCGTTT and fp:B:C,0,0,0,3,4,5,6,0,0,0, then any downstream application should only use CGCG in its analysis, and ignore the AAA and TTT stretches.

2.12 Use of read tags for per-read-base base modifications

The following read tags encode base modification information. Base modifications are encoded according to the [SAM tags specifications](#) and any conflict is unintentional.

Tag	Type	Description
Mm	Z	Base modifications / methylation
Ml	B,C	Base modification probabilities

Notes:

- For informational purposes only: The continuous probability range of 0.0 to 1.0 is remapped to the discrete integers 0 to 255 inclusively in the Ml tag. The probability range corresponding to an integer N is $N/256$ to $(N + 1)/256$.

2.13 How to annotate scrap reads

Reads that belong to a read group with READTYPE=SCRAP have to be annotated in a hierarchical fashion:

- 1) Classification with tag *sz* occurs on a per ZMW level, distinguishing between spike-in controls, sentinels of the basecaller, malformed ZMWs, and user-defined templates.
- 2) A region-wise annotation with tag *sc* to label adapters, barcodes, low-quality regions, and filtered subreads.

Tag	Type	Description
sz	A	ZMW classification annotation, one of N:=Normal, C:=Control, M:=Malformed, or S:=Sentinel ¹
sc	A	Scrap region-type annotation, one of A:=Adapter, B:=Barcode, L:=LQRegion, or F:=Filtered ²

¹ reads in the subreads/hqregions/zmws.bam file are implicitly marked as Normal, as they stem from user-defined templates.

² *sc* tags 'A', 'B', and 'L' denote specific classes of non-subread data, whereas the 'F' tag is reserved for subreads that are undesirable for downstream analysis, e.g., being artifactual or too short.

2.14 QUAL

The QUAL field in BAM alignments is intended to reflect the reliability of a basecall, using the Phred-encoding convention, as described in the [SAM spec](#).

Both CCS and raw read BAM files respect this convention; historically, and for the present moment, the encoded probability reflects the confidence of a basecall against alternatives including substitution, deletion, and insertion.

We expect that more details will follow here in a later spec revision.

2.15 Subread local context

Some algorithms can make use of knowledge that a subread was flanked on both sides by adapter or barcode hits, or that the subread was in one orientation or the other (as can be deduced when asymmetric adapters or barcodes are used).

To facilitate such algorithms, we furnish the `cx` bitmask tag for subread records. The `cx` value is calculated by binary OR-ing together values from this flags enum:

```
enum LocalContextFlags
{
    ADAPTER_BEFORE      = 1,
    ADAPTER_AFTER       = 2,
    BARCODE_BEFORE      = 4,
    BARCODE_AFTER       = 8,
    FORWARD_PASS        = 16,
    REVERSE_PASS        = 32,
    ADAPTER_BEFORE_BAD  = 64,
    ADAPTER_AFTER_BAD   = 128
};
```

Orientation of a subread (designated by one of the mutually exclusive `FORWARD_PASS` or `REVERSE_PASS` bits) can be reckoned only if either the adapters or barcode design is asymmetric, otherwise these flags must be left unset. The convention for what is considered a “forward” or “reverse” pass is determined by a per-ZMW convention, defining one element of the asymmetric barcode/adaptor pair as the “front” and the other as the “back”. It is up to tools producing the BAM to determine whether to use adapters or barcodes to reckon the orientation, but if pass directions cannot be confidently and consistently assessed for the subreads from a ZMW, neither orientation flag should be set. Tools consuming the BAM should be aware that orientation information may be unavailable for subreads in a ZMW, but if is available for any subread in the ZMW, it will be available for all subreads in the ZMW.

The `ADAPTER_*` and `BARCODE_*` flags reflect whether the subread is flanked by adapters or barcodes at the ends.

The `ADAPTER_BEFORE_BAD` and `ADAPTER_AFTER_BAD` flags indicate that one or both adapters flanking this subread do not align to the adapter reference sequence(s). The adapter on this flank could be missing from the pbell molecule, or obscured by a local decrease in accuracy. Likewise, some nearby barcode or insert bases may be missing or obscured. `ADAPTER_*_BAD` flags can not be set unless the corresponding `ADAPTER_*` flag is set.

This tag is mandatory for subread records, but will be absent from non-subread records (scraps, ZMW read, CCS read, etc.)

Tag	Type	Description
<code>cx</code>	<code>i</code>	Subread local context Flags

2.16 Missing adapter annotation in CCS reads

The `ma` and `ac` tags indicate whether the molecule that produces a CCS read is missing a SMRTbell adapter on its left/start or right/end. The tags are produced by CCS version 6.3.0 and newer based on the `ADAPTER_BEFORE_BAD` and `ADAPTER_AFTER_BAD` information in the subread `cx` tag.

Tag	Type	Description
ac	B,i	Array containing four counts, in order: - detected adapters on left/start - missing adapters on left/start - detected adapters on right/end - missing adapter on right/end
ma	i	Bitmask storing if an adapter is missing on either side of the molecule. A value of 0 indicates neither end has a confirmed missing adapter. - 0x1 if adapter is missing on left/start - 0x2 if adapter is missing on right/end

2.17 Barcode analysis

In multiplexed workflows, we record per-subread tags representing the barcode call and a score representing the confidence of that call. The actual data used to inform the barcode calls—the barcode sequences and associated pulse features—will be retained in the associated `scraps.bam` file.

Tag	Type	Description
bc	B,S	Barcode Calls (per-ZMW)
bq	i	Barcode Quality (per-ZMW)

- Both the `bc` and `bq` tags are calculated per-ZMW, so every subread belonging to a given ZMW should share identical `bc` and `bq` values. The tags are also inter-dependent, so if a subread has the `bc` tag, it must also have a `bq` tag and vice-versa. If the tags are present for any subread in a ZMW, they must be present for all of them. In the absence of barcodes, both the `bc` and `bq` tags will be absent
- The `bc` tag contains the *barcode call*, a `uint16[2]` representing the inferred forward and reverse barcode sequences (as determined by their ordering in the Barcode FASTA), or more succinctly, it contains the integer pair B_F, B_R . Integer codes represent 0-based position in the FASTA file of barcodes.
- The integer (`int`) `bq` tag contains the barcode call confidence. If the `BarcodeQuality` element of the header is set to `Score`, then the tag represents the mean normalized sum of the calculated Smith-Waterman scores that support the call in the `bc` tag across all subreads. For each barcode, the sum of the Smith-Waterman score is normalized by the length of the barcode times the match score, then multiplied by 100 and rounded; this provides an integer value between 0 - 100. On the other hand, if the value of the header-tag is `Probability` instead, then the tag value is a the Phred-scaled posterior probability that the barcode call in `bc` is correct. In both cases, the value will never exceed the `int8` range, but for backward-compatibility reasons we keep the BAM `bq` as `int`. This contract allows the PBI to store `bq` as a much smaller `int8`.

The following (optional) tags describe clipped barcode sequences:

Tag	Type	Description
bl	Z	Barcode sequence clipped from leading end
bt	Z	Barcode sequence clipped from trailing end
ql	Z	Qualities of barcode bases clipped from leading end, stored as a FASTQ string
qt	Z	Qualities of barcode bases clipped from trailing end, stored as a FASTQ string
bx	B,i	Pair of clipped barcode sequence lengths

Barcode information will follow the same convention in CCS output (`ccs.bam` files).

2.17.1 Examples (subreads)

Scenario	bc	bq	cx
No barcodes, end-to-end, unknown orientation	<i>absent</i>	<i>absent</i>	1 2 = 3
Asymmetric barcodes, end-to-end, forward pass	{ 1, 37 }	35	1 2 4 8 16 = 31
Symmetric barcodes, end-to end	{ 8, 8 }	33	1 2 4 8 = 15
Barcoded, HQ region terminates before second barcode; unknown orientation	{ 8, 8 }	33	1 4 = 5

2.18 Alignment: the contract for a mapper

An aligner is expected to accept BAM input and produce aligned BAM output, where each aligned BAM record in the output preserves intact all tags present in the original record. The aligner should not attempt to orient or complement any of the tags.

(Note that this contrasts with the handling of *SEQ* and *QUAL*, which are mandated by the BAM/SAM specification to be (respectively) reverse-complemented, and reversed, for reverse strand alignments.)

2.19 Alignment: soft-clipping

In the standard production configuration, PacBio’s aligners will be used to align either subreads or CCS reads. In either case, we will use *soft clipping* to preserve the unaligned bases at either end of the query in the aligned BAM file.

2.20 Encoding of kinetics pulse features

Interpulse duration (IPD) and pulsewidth are measured in frames; natively they are recorded as a `uint16` per pulse/base event. They may be encoded in BAM read tags in one of two fashions:

- losslessly as an array of `uint16`; necessary for PacBio-internal applications but entails greater disk space usage.
- lossy 8-bit compression stored as a `uint8` array, following the codec specified below (“codec V1”). Provides a substantial disk-space savings without affecting important production use cases (base modification detection).

In the default production instrument configuration, the lossy encoding will be used. The instrument can be switched into a mode (PacBio-internal mode) where it will emit the full lossless kinetic features.

The lossy encoding for IPD and pulsewidth values into the available 256 codepoints is as follows (**codec v1**):

Frames	Encoding
0 .. 63	0, 1, .. 63
64, 66, .. 190	64, 65, .. 127
192, 196 .. 444	128, 129 .. 191
448, 456, .. 952	192, 193 .. 255

In other words, we use the first 64 codepoints to encode frame counts at single frame resolution, the next 64 to encode the frame counts at two-frame resolution, and so on. Durations exceeding 952 frames are capped at 952. Durations not enumerated in “Frames” above are rounded to the nearest enumerated duration then encoded. For example, a duration of 194 frames would round to 196 and then be encoded as codepoint 129.

This encoding has the following features, considered essential for internal analysis use cases:

- *Exact* frame-level resolution for small durations (up to 64 frames)
- Maximal representable duration is 9.52 seconds (at 100fps), which is reasonably far into the tail of the distributions of these metrics. Analyses of “pausing” phenomena may still need to account for this censoring.

A reference implementation of this encoding/decoding scheme can be found in *pbcore*.

2.21 Unresolved issues

- Need to move from strings to proper array types for QVs
- ‘/’ preferable to ‘.’ in “IPD:CodecV1”
- Desire for spec for shorter movienames, especially if these are ending up in QNAMEs.

PacBio BAM index file (`bam.pbi`) format

PacBio's previous alignment file format (`cmp.h5`) contained a data table called the *alignment index* that recorded auxiliary identifying information and precomputed summary statistics per aligned read. This table served several purposes:

1. it enabled fast random access to aligned reads satisfying fairly complex predicates, for example, reads from a specific list of ZMWs which had unambiguous mapping (`MapQV==254`), or a read with a given readname.
2. it allowed summary reports (readlength, mapped identity/accuracy, etc.) to be constructed by quick operations over the alignment index instead of loading all of the sequence reads for each analysis.

In order to provide backwards-compatibility with the APIs enabled for accessing the `cmp.h5`, we have devised a new BAM companion file, the *PacBio BAM index*, which supports the two use cases above.

3.1 Version

This is version `4.0.0` of the `bam.pbi` specification.

3.2 Changelog

[4.0.0] - 2020-10-08

- Added `nInsOps`, `nDelOps` fields to mapped data section.

[3.0.2] - 2018-09-13

- `qStart`, `qLen` for CCS records now stored as `(0,qLen)` instead of `(-1,-1)`

[3.0.1] - 2015-09-23

- Local context flags moved into the always-present “basic” data section.

[3.0.0] - 2015-07-12

- Initial publishing of specification.

3.3 Format

The format mimics the HDF5 column-based approach without requiring the additional library dependency. Most sections are laid out as a series of 1-dimensional arrays, a la HDF5 datasets. Calculating an average or maximum mapQV, for example, would simply involve a block read of the array and the relevant computation.

It enables the 2 major use cases listed above

1. Random-access queries, including:
 - by reference or genomic region
 - by read group
 - by query name
 - by ZMW
 - by barcode index
 - etc.
2. Obtain information without processing entire BAM file
 - Calculate summary statistics
 - Reverse-lookup - get information for a record, given its index

Like the associated BAM & BAI formats, PBI is compressed in the BGZF format. See [SAM/BAM spec](#) for details.

All multi-byte numbers in PBI are stored little-endian.

All optional columns in PBI are either present for all rows or for none of them, and always present in the order described in this document. This is important for correctly accessing specific values by column index.

File sections follow each other immediately in the file and are described below.

- *PBI Header*
- *Basic Data*
- *Mapped Data* (optional)
- *Coordinate-Sorted Data* (optional)
- *Barcode Data* (optional)

3.3.1 PBI Header

Field	Size	Definition	Value
magic	char[4]	PBI magic string	PBI\1
version	uint32_t	PBI format version (xx.yy.zz)	0x00xyyyzz
pbi_flags	uint16_t	bitflag describing file contents ¹	
n_reads	uint32_t	number of reads in the BAM file	
reserved	char[18]	reserved space for future expansion	fill(0x00)

¹ pbi_flags:

Flag	Value	Description
Basic	0x0000	PbiHeader & BasicData only
Mapped	0x0001	MappedData section present
Coordinate Sorted	0x0002	CoordinateSortedData section present
Barcode	0x0004	BarcodeData section present

(0x0008 - 0x8000) are available to mark future data modifiers, add'l sections, etc.

3.3.2 Basic Data

This section contains data that apply to all PacBio BAM reads.

Each index field is laid out as a column (array), consisting of one value per BAM record. Thus for N records, this section will contain N rgId values, followed by N qStart values, then N qEnd values, and so on.

Field	Size	Definition
rgId	int32_t	Integral value of @RG::ID ¹
qStart	int32_t	Start of query in polymerase read (0 for CCS reads)
qEnd	int32_t	End of query in polymerase read (qLen for CCS reads)
holeNumber	int32_t	The holenummer of the ZMW producing the read
readQual	float	Expected accuracy ('rq' tag) [0-1]
ctxt_flag	uint8_t	Local context of subread ('cx' tag) ²
fileOffset	int64_t	Virtual offset of record (bgzf_tell)

¹ Read group identifiers for PacBio data are calculated as follows:

```
RGID_STRING := md5(movieName + "/" + readType)) [:8]
RGID_INT    := int32.Parse(RGID_STRING)

RGID_STRING is used in the @RG header and in the `RG` tag of BAM
records.  RGID_INT is used here in the PBI index.

Note that RGID_INT may be negative.
```

² Local context flags are only valid for Subread / Insert records. For all other record-types, or if the CX tag is not present in the record, this value should be 0

3.3.3 Mapped Data

This section contains data that apply to all mapped PacBio BAM reads.

Each index field is laid out as a column (array), consisting of one value per BAM record. Thus for N records, this section will contain N tId values, followed by N tStart values, then N tEnd values, and so on.

Field	Size	Definition
tId	int32_t	BAM tid indication aligned reference
tStart	uint32_t	(0-based) Start of alignment in reference
tEnd	uint32_t	End of alignment in reference (endpos)
aStart	uint32_t	Start of aligned query in polymerase read
aEnd	uint32_t	End of aligned query in polymerase read
revStrand	uint8_t	1 if reverse strand alignment, else 0
nM	uint32_t	Number of base matches in alignment
nMM	uint32_t	Number of base mismatches in alignment
mapQV	uint8_t	The mapping quality [valid ranges 0-254]
nInsOps	uint32_t	Number of insertion operations (not bases)
nDelOps	uint32_t	Number of deletion operations (not bases)

Note: Inserted and deleted base counts are not included in the index. These values are readily computed as:

```
nInsertedBases = aEnd - aStart - nM - nMM
nDeletedBases = tEnd - tStart - nM - nMM
```

Alignment length can be computed using:

```
aEnd - aStart + tEnd - tStart - nM - nMM
```

3.3.4 Coordinate-Sorted Data

In a coordinate-sorted BAM file, the records that are mapped to each reference form contiguous blocks. The data in this section provide a mapping between each tId and its start/end rows ².

The lookup table is prefixed with the number of reference entries.

Field	Size	Definition
n_tids	uint32_t	Number of reference sequences

The lookup table is laid out as a column (array) of tuples, one per reference.

Field	Size	Definition
tId	uint32_t	reference sequence ID ¹
beginRow	uint32_t	index of first record mapped on tId ²
endRow	uint32_t	index of last record mapped on tId ²

¹ This dataset should be sorted in *ascending order of the uint32 cast of tId* (thus a tId of -1 will follow all other tId values)

² Data fields beginRow and endRow. If $tId[i] == t$, then $[beginRow, endRow)$ represents range of reads (by 0-based ordinal position in the BAM file) mapped to the reference contig with *tId* of *t*. If no BAM records are aligned to *t*, then we should have $beginRow, endRow = -1$.

3.3.5 Barcode Data

This section contains data that apply to all barcoded PacBio BAM reads.

Each index field is laid out as a column (array), consisting of one value per BAM record. Thus for N records, this section will contain N bc_forward values, followed by N bc_reverse values, then N bc_qual values.

Field	Size	Definition
bc_forward	int16_t	B_F from 'bc' tag (index to barcode FASTA), or -1 if not present
bc_reverse	int16_t	B_R from 'bc' tag (index to barcode FASTA), or -1 if not present
bc_qual	int8_t	barcode call confidence ('bq' tag), or -1 if not present

Note:

If the Barcode flag is set in the header, these values must be present in all rows, otherwise it should be present for none of them.

If one Barcode field is set to -1 / non-existent, then all barcode-related fields should be set as such.

FASTA file format

4.1 Best practices

The ubiquitous **FASTA** format is flexible, to a fault. The following best practices will guarantee success in using FASTA files with PacBio software (for example as genome references).

1. Sequences in FASTA files should be wrapped at a uniform line length, to enable indexing. (A common convention is to wrap lines at 60 characters.) Windows and UNIX line endings are both acceptable.
2. FASTA files should contain no empty lines (or lines containing only whitespace¹). FASTA files should contain no unnecessary whitespace (for example, no trailing whitespace on sequence lines).
3. Sequence contigs in a FASTA file are preceded by identifying header lines. The *header*, which is the text between the ‘>’ character and the newline, is comprised of:
 - an *identifier*, which is the first whitespace-delimited token of the header
 - an (optional) *comment*, which consists of the suffix of the header following the first whitespace. Some APIs may call this string the *description* or the *metadata*, but the usage of the comment is completely application-defined.

Sequence contigs will be identified in the PacBio system and in downstream analysis files using the *identifier*, so the *identifiers* contained in a FASTA file must be unique.

4. Sequences should only contain characters from the following string (IUPAC nucleotide characters, minus ‘-’ and ‘.’):

```
"gatcuryswkmbdhvnGATCURYSWKMBDHVN"
```

5. Headers should not contain any instances of the ‘>’ character, and *identifiers* should not begin with an asterisk (*) or contain any of the following characters:

```
',' ' ':' '"" (double quote)
```

¹ We define a whitespace character as one for which the standard C *isspace* function returns *true* (nonzero).

6. PacBio software that imports reference genomes is allowed to outright reject FASTA files that do not meet our requirements; it will never attempt to rewrite/translate a file to fit the requirements.

4.2 Examples

The following FASTA file would be accepted by the PacBio reference uploader. Downstream files (reports, variant call files) would report the chromosomes as ‘chr1’, ‘chr2’:

```
>chr1 Jackalope chromosome 1;length=7
GATTACA
>chr2 Jackalope chromosome 2;length=7
TTACAGA
```

The following file would be *rejected* by the PacBio reference uploader, for violating the identifier uniqueness requirement (in 3, above):

```
>Jackalope chromosome 1;length=7
GATTACA
>Jackalope chromosome 2;length=7
TTACAGA
```

4.3 Implications for bioinformatics tool writers

Since a FASTA presented to a bioinformatics tool may contain characters beyond “GATCN”, tools should use a FASTA reader API capable of normalizing sequence characters. For example, [SeqAn](#) uses a *Dna5String* abstraction to project input sequence characters into “GATCN”; FASTA readers in pbcore will soon offer normalization.

DataSet format specification

A PacBio DataSet is an XML file representing a set of a particular sequence data type such as subreads, references or aligned subreads. The actual data elements contained in a DataSet are stored in files referred to by the XML, usually in FASTA or BAM files. The DataSet can optionally filter these files or store metadata about their contents.

5.1 Version

This document defines the 4.0.1 SMRT Link DataSet abstraction and its XML file representation. The 4.0.1 format is produced by the 11.0.0 release of PacBio Primary Analysis and SMRT Link software.

5.2 Motivating Use Cases

The concept of a homogenous set of elements of a particular data type is used throughout Pacific Biosciences software. These “sets of data” have historically been represented in different ways, sometimes explicitly by creating large files that contain all the data in the set (e.g. a FASTA file of subreads), and sometimes implicitly by using pointers to the data, such as with a FOFN (file of file names).

In many cases these solutions are serviceable, but they have disadvantages. The reliance on explicit file creation imposes a heavy burden that will be exacerbated by future instrument throughput increases. The FOFN partially breaks the tight coupling between explicit files and sets of data, but it fails to allow facile subsetting of files. With previous methods, consuming tools were forced to reimplement filtering or subsetting logic in their own idiosyncratic ways.

The DataSet XML abstraction (or DataSet for short) addresses these problems by providing a standard way to perform the following activities:

- Refer to a **set of subreads** in multiple BAM files
- Refer to a **subset of subreads** by id from one or more BAM files
- Refer to a **subset of alignments** for input to algorithms such as Arrow (e.g. on a particular chromosome, or a particular chromosome region)

- Perform any analysis that can be performed on an entire file of a particular data type (subreads, alignments) on a **subset of that data type** without creating a new file.

5.3 Format Definition

5.3.1 XML Representation

The canonical representation of a DataSet is an XML file that contains a single DataSet element with four major sections, one mandatory and three optional:

1. A mandatory `<pbbase:ExternalResources>` section with references to external data sources in BAM or FASTA format and their associated indices or metadata files. The records in these files are the elements of the set of data represented by the DataSet.
2. An optional `<pbdbs:Filters>` section that filters or subsets the elements of the set in the above files, for example by length.
3. An optional `<pbdbs:DataSetMetadata>` section that contains metadata about the DataSet, usually at minimum the number of records and their total length, but possibly much more. For example, subread and CCS read DataSets have metadata regarding instrument collection or biological samples. The Metadata section should be considered to refer to the DataSet elements prior to applying Filters.
4. An optional `<pbdbs:DataSets>` section that can be used to annotate subsets of the top-level DataSet. For example, a DataSet that resulted from merging multiple DataSets can store a record of the original DataSets in this section, along with their original metadata.
5. An optional `<pbbase:SupplementalResources>` section with references to external data or metadata sources in any format, separate from the main data files in ExternalResources, but represented using the same `<pbbase:ExternalResource>` model. This may include files such as reports or unbarcoded reads BAM.

5.3.2 Types of DataSets

The DataSet `XSD` defines DataSet subclasses for the most common entities consumed and produced by SMRT Analysis pipelines:

- **SubreadSet** - This type of DataSet represents the basic sequence data generated by the Sequel instrument, with `ExternalResource`'s in BAM format. Note that the prefix "Sub" in this case is part of "Subread", and does not in any way denote a "subset".
- **ConsensusReadSet** - CCS sequence data in BAM format.
- **AlignmentSet** - Aligned reads in BAM format.
- **ConsensusAlignmentSet** - Aligned CCS data in BAM format.
- **ReferenceSet** - The FASTA reference and associated indices used by Resequencing and Base Mods. Replaces the reference repository entries.
- **GmapReferenceSet** - A FASTA ReferenceSet with a GMAP database.
- **ContigSet** - Any FASTA containing contigs, e.g. those produced by HGAP.
- **BarcodeSet** - The FASTA file and metadata used by barcode detection.
- **TranscriptSet** - Processed RNA transcripts in BAM format.
- **TranscriptAlignmentSet** - Processed and aligned RNA transcripts in BAM format.

5.3.3 SubreadSet example

Here is a simple example of a DataSet XML using a SubreadSet containing all four sections. It creates a set of subreads from two subread BAM files and associated indices and filters the subreads by quality using the `rq` field of the underlying BAM records:

```
<?xml version="1.0" encoding="utf-8"?>
<pbds:SubreadSet
  xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  xmlns:pbbmeta="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="subreadset_150304_231155"
  MetaType="PacBio.DataSet.SubreadSet"
  Name="DataSet_SubreadSet"
  Tags=""
  Version="4.0.1"
  CreatedAt="2015-01-27T09:00:01">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5193"
      TimeStampedName="subread_bam_150304_231155"
      MetaType="PacBio.SubreadFile.SubreadBamFile"
      ResourceId="m150404_101626_42267_s1_p0.1.subreads.bam">
      <pbbase:FileIndices>
        <pbbase:FileIndex
          UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5194"
          TimeStampedName="bam_index_150304_231155"
          MetaType="PacBio.Index.PacBioIndex"
          ResourceId="m150404_101626_42267_s1_p0.1.subreads.bam.pbi"/>
        </pbbase:FileIndices>
      </pbbase:ExternalResource>
      <pbbase:ExternalResource
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5197"
        TimeStampedName="subread_bam_150304_231155"
        MetaType="PacBio.SubreadFile.SubreadBamFile"
        ResourceId="m150404_101626_42267_s1_p0.2.subreads.bam">
        <pbbase:FileIndices>
          <pbbase:FileIndex
            UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5198"
            TimeStampedName="bam_index_150304_231155"
            MetaType="PacBio.Index.PacBioIndex"
            ResourceId="m150404_101626_42267_s1_p0.2.subreads.bam.pbi"/>
          </pbbase:FileIndices>
        </pbbase:ExternalResource>
      </pbbase:ExternalResources>
    <pbds:Filters>
      <pbds:Filter>
        <pbbase:Properties>
          <pbbase:Property Name="rq" Operator="gt" Value="0.80"/>
        </pbbase:Properties>
      </pbds:Filter>
    </pbds:Filters>
  </pbds:SubreadSet>
```

(continues on next page)

(continued from previous page)

```
<pbbase:TotalLength>5000</pbbase:TotalLength>
<pbbase:NumRecords>500</pbbase:NumRecords>
</pbds:DataSetMetadata>
</pbds:SubreadSet>
```

5.3.4 Operations on DataSets

DataSets support operations that would naively be expected of sets, such as subsetting and union (although notably not intersection) as well as some additional operations such as consolidation. The result of performing these operations is itself a new DataSet, with the operations included as a kind of “recipe” for producing the new DataSet from the original. Because of this, operations are presented here as part of the DataSet format.

Subsetting (Filtering)

The SubreadSet example given above is the result of applying a length subset or `Filter` operation on a DataSet of two BAM files (e.g. by applying the `dataset` command).

Each `Filter` is composed of `Property` tags representing logical predicates that elements of the DataSet must satisfy. `Property` tags are defined by three attributes: `Name`, `Operator` and `Value`, where the `Name` refers to a field or derived value from individual DataSet records, and the `Operator` is used to compare a particular record’s field with the `Value` attribute. Individual `Filter` tags can be combined to create more complicated filters; these `Filter` tags are logically “ORed” while individual `Property` tags within a `Filter` are “ANDed” together.

In summary, the `Filter` and `Property` tags provide a powerful means of subsetting DataSets without manipulating the underlying file representations.

Filter Property Names and Values

The following table shows `Property Name`’s that are supported by either the pbbam C++ API, the pbcore Python API (*italicized*) or both (**bold**). Some accepted alternative ways of representing a `Property` are given in the `Alternative Names` column. Those `Property`’s that allow list values in both pbbam and pbcore are indicated by brackets. Finally, the right column shows the allowed type of `Property Value` associated with that particular `Property Name`.

Property Name	Description	Alternative Names	Value
readstart	Alignment start	astart as	uint32_t
ae	Alignment end	aend	uint32_t
alignedlength	Alignment length		uint32_t
accuracy	Alignment identity	identity	float
qname	Query name	<i>qid</i>	string
qstart	Query start	qs	int
qend	Query end	qe	int
length	Query length	querylength	int
rname	Reference name		string
tstart	Reference start	ts pos	uint32_t
tend	Reference end	te	uint32_t
qname_file	Query names from a file		filename
rq	Predicted read quality		float
movie	Movie name (e.g. m150404_101626_42267_s1_p0)		string
zm	ZMW (e.g. m150404_101626_42267_s1_p0/100)	zmw	string[]
bc	Barcode name	barcode	string[]
bcq	Barcode quality	bq	uint8_t
bcf	Barcode forward		string[]
bcr	Barcode reverse		string[]
cx	Local context (see below for special Values)		see below
<i>n_subreads</i>	Number of subreads		int
<i>mapqv</i>	Mapping quality		int

Filter Property Operators

The following table shows supported Property Operator's:

Property Operator	Description	Other Names
==	Equal to	= eq
!=	Not Equal to	ne
<	Less Than	lt <
<=	Less than or equal to	lte <=
>	Greater than	gt >
>=	Greater than or equal to	gte >=
&	Contains	and
~	Does not contain	not

Possible cx Values

The following table shows Values's that are supported for the **cx** local context Property Name. The Value's can be used individually or can be combined to form a compound Value using a syntax that looks and behaves similarly to OR-ing bitflags, e.g.:

```
Value='ADAPTER_BEFORE | ADAPTER_AFTER'
```

cx Value	Description
NO_LOCAL_CONTEXT	No local context (e.g. adapters or barcodes)
ADAPTER_BEFORE	An adapter was detected before (5' to) the subread
ADAPTER_AFTER	An adapter was detected after (3' to) this subread
BARCODE_BEFORE	A barcode was detected before (5' to) this subread
BARCODE_AFTER	A barcode was detected after (3' to) this subread
FORWARD_PASS	Subread is forward on the polymerase read
REVERSE_PASS	Subread is forward on the polymerase read

Filtering and the pbi

The subsetting/filtering operation is supported efficiently by the PacBio index (*.pbi files). The contents of DataSet after applying `Filter`'s should in general be decidable using the contents of the pbi file (`rname` is the one exception in 4.0 and requires examination of the BAM file header).

Union (Merging)

Unions can be taken of DataSets with the same underlying file type (noted by the `MetaType` attribute) and with identical `Filter`'s. The output of a union (or merge) operation is a *single* new DataSet XML file containing a *single* new top-level DataSet element—notably not multiple top-level elements. For example, the SubreadSet above could be created by taking the union of two SubreadSets each containing a single BAM file:

```
<pbd:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbb:ExternalResources>
    <pbb:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0.bam"/>
  </pbb:ExternalResources>
  <pbd:Filters>
    <pbd:Filter>
      <pbb:Properties>
        <pbb:Property Name="rq" Operator="gt" Value="0.75"/>
      </pbb:Properties>
    </pbd:Filter>
  </pbd:Filters>
  <pbd:DataSetMetadata>
    <pbb:TotalLength>3000</pbb:TotalLength>
    <pbb:NumRecords>300</pbb:NumRecords>
  </pbd:DataSetMetadata>
</pbd:SubreadSet>

<pbd:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbb:ExternalResources>
    <pbb:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads1.bam"/>
  </pbb:ExternalResources>
  <pbd:Filters>
    <pbd:Filter>
      <pbb:Properties>
        <pbb:Property Name="rq" Operator="gt" Value="0.75"/>
      </pbb:Properties>
    </pbd:Filter>
  </pbd:Filters>
```

(continues on next page)

(continued from previous page)

```

<pbds:DataSetMetadata>
  <pbbase:TotalLength>2000</pbbase:TotalLength>
  <pbbase:NumRecords>200</pbbase:NumRecords>
</pbds:DataSetMetadata>
</pbds:SubreadSet>

```

Tools that merge multiple DataSets together can optionally store the original input DataSets in the DataSets tag of the output DataSet. In this way the original DataSets are maintained as “subdatasets” of the new DataSet. Merging tools should adhere to the following best practices:

- Subdatasets should be created for each input DataSet when two or more files with no subdatasets are merged or used to create a new DataSet.
- If in the process of merging a new DataSet is created, for example, wrapping a “naked” BAM file, a subdataset should be created for those inputs.
- In no case should a subdataset be present if it is identical to the containing DataSet.
- Subdatasets should be preserved when two datasets with subdatasets are merged together.
- If both have subdatasets, the lists of subdatasets should be concatenated.
- If one has no subdatasets, a subdataset should be created for that input and added to the list of subdatasets from the other input file.
- Subdatasets should be preserved during minor manipulations: Adding a filter, changing path absoluteness, adding or removing indices, references etc.
- Subdatasets should be removed during substantial transformations: alignment, producing an AlignmentSet from a SubreadSet.
- Subdatasets should be preserved during consolidation or splitting (except when splitting by subdataset).

Consolidating

Consolidating (aka Resolving) a DataSet means creating an explicit representation in the appropriate format with all filters applied. In practice, this means building a single BAM file containing all the records implied by the DataSet’s ExternalResource and Filter directives. Here is consolidated version of the SubreadSet above:

```

<?xml version="1.0" encoding="utf-8" ?>
<pbds:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource
      MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0_plus_subreads1.bam"/>
  </pbbase:ExternalResources>
  <pbds:DataSetMetadata>
    <pbbase:TotalLength>5000</pbbase:TotalLength>
    <pbbase:NumRecords>500</pbbase:NumRecords>
  </pbds:DataSetMetadata>
</pbds:SubreadSet>

```

Consolidated DataSets are useful for export of a DataSet that exists only implicitly, e.g. by filtering multiple files. They allow the user to incur the IO overhead of seeking over multiple files once at the cost of increased disk usage.

5.3.5 DataSet MetaTypes and File Extensions

The following table lists the `DataSet` subclasses defined by the XSD and their associated `MetaType` values and expected filename suffix strings. While technically the information conveyed by the `MetaType` and suffix is redundant, it is useful for some downstream consuming tools, and to be fully compliant with this specification all three should be consistent.

DataSet	DataSet MetaType	DataSet XML File Extension
SubreadSet	PacBio.DataSet.SubreadSet	.subreadset.xml
HdfSubreadSet	PacBio.DataSet.HdfSubreadSet	.hdfsubreadset.xml
AlignmentSet	PacBio.DataSet.AlignmentSet	.alignmentset.xml
BarcodeSet	PacBio.DataSet.BarcodeSet	.barcodeset.xml
ConsensusReadSet	PacBio.DataSet.ConsensusReadSet	.consensusreadset.xml
ConsensusAlignmentSet	PacBio.DataSet.ConsensusAlignmentSet	.consensusalignmentset.xml
ContigSet	PacBio.DataSet.ContigSet	.contigset.xml
ReferenceSet	PacBio.DataSet.ReferenceSet	.referenceset.xml
GmapReferenceSet	PacBio.DataSet.GmapReferenceSet	.gmapreferenceset.xml
TranscriptSet	PacBio.DataSet.TranscriptSet	.transcriptset.xml
TranscriptAlignmentSet PacBio.DataSet.TranscriptAlignmentSet .transcriptalignmentset.xml		

DataSet External Resource MetaTypes and File Extensions

The following table lists the `ExternalResource`'s expected by each `DataSet` subclass and their associated `MetaType` values and expected filename suffix strings. In some cases the `ExternalResource` can occur in multiple `DataSet` types, but only for particular parent `ExternalResource`'s elements. Note that while this specification allows `ExternalResource`'s to in some cases refer to other `DataSets`, they should in no cases refer to a `DataSet` of the same type as the parent (e.g. `SubreadSet`'s should not refer to external `SubreadSet`'s).

DataSet or Tag	ExternalResource MetaType	File Extensions
SubreadSet	PacBio.SubreadFile.SubreadBamFile	.bam
	PacBio.SubreadFile.ScrapBamFile	.bam
	PacBio.SubreadFile.HqRegionBamFile	.bam
	PacBio.SubreadFile.HqScrapBamFile	.bam
	PacBio.SubreadFile.LqRegionBamFile	.bam
	PacBio.SubreadFile.LqScrapBamFile	.bam
	PacBio.SubreadFile.PolymeraseBamFile	.bam
	PacBio.SubreadFile.PolymeraseScrapBamFile	.bam
	PacBio.SubreadFile.ChipStatsFile	.sts.xml
	PacBio.SubreadFile.ChipStatsH5File	.sts.h5
	PacBio.SubreadFile.AdapterFastaFile	.fasta
	PacBio.SubreadFile.ControlFastaFile	.fasta
HdfSubreadSet	PacBio.SubreadFile.BaxFile	.bax.h5
AlignmentSet	PacBio.AlignmentFile.AlignmentBamFile	.bam
BarcodeSet	PacBio.BarcodeFile.BarcodeFastaFile	.fasta
ConsensusReadSet	PacBio.ConsensusReadFile.ConsensusReadBamFile	.bam
ConsensusAlignmentSet	PacBio.AlignmentFile.ConsensusAlignmentBamFile	.bam
TranscriptSet	PacBio.TranscriptFile.TranscriptBamFile	.bam
TranscriptAlignmentSet	PacBio.AlignmentFile.TranscriptAlignmentBamFile	.bam
ContigSet	PacBio.ContigFile.ContigFastaFile	.fasta
ReferenceSet	PacBio.ReferenceFile.ReferenceFastaFile	.fasta
Bam ExternalResource	PacBio.Index.BamIndex	.bam.bai
	PacBio.Index.PacBioIndex	.bam.pbi
Fasta ExternalResource	PacBio.Index.SamIndex	.fasta.fai
	PacBio.Index.SaWriterIndex	.fasta.sa
	PacBio.GmapDB.GmapDBSummary	?
	PacBio.Index.NgmlrRefEncoded	.ngm
	PacBio.Index.NgmlrRefTable	.ngm

Some `ExternalResource`'s themselves contain associated `ExternalResource`'s, for example the indices associated with FASTA files. These associated files are nested within the primary `ExternalResource` element to denote their subsidiary nature.

DataSet UI Name and Time Stamped Name

The pattern for `TimeStampedName` attribute generated by production PacBio tools should be:

```
<metatype>-<yymmdd_HHmmsstt>
```

Where `MetaType` has been transformed into a lowercase, underscore separated string and the time string format directives map to the following entities: year, month, day, hour, minute, second, millisecond.

DataSet	TimeStampedName	DataSet UI Name
SubreadSet	pacbio_dataset_subreadset-<yymmdd_HHmmssttt>	BAM Data
HdfSubreadSet	pacbio_dataset_hdfsubreadset-<yymmdd_HHmmssttt>	RS II Data
AlignmentSet	pacbio_dataset_alignmentset-<yymmdd_HHmmssttt>	TBD
BarcodeSet	pacbio_dataset_barcodeaset-<yymmdd_HHmmssttt>	Barcode
ConsensusReadSet	pacbio_dataset_consensusreadset-<yymmdd_HHmmssttt>	TBD
ConsensusAlign- mentSet	pacbio_dataset_consensusalignmentset- <yymmdd_HHmmssttt>	TBD
ContigSet	pacbio_dataset_contigset-<yymmdd_HHmmssttt>	TBD
ReferenceSet	pacbio_dataset_referenceset-<yymmdd_HHmmssttt>	Reference
TranscriptSet	pacbio_dataset_transcriptset-<yymmdd_HHmmssttt>	TBD
TranscriptAlignmentSet	pacbio_dataset_transcriptalignmentset-<yymmdd_HHmmssttt>	TBD

Note that all `PacBio.Index.*ExternalResource`'s are given a `TimeStampedName` attribute with an "index-" prefix.

5.3.6 Singleton DataSet element

While technically an XML file with multiple `DataSet` elements is valid under the XSD and this spec, in practice PacBio tools follow a one `DataSet` element per XML file convention.

5.4 Support for the DataSet XML

Support exists for using the `DataSet XML` throughout the SMRT Analysis stack.

5.4.1 The dataset command-line tool

The SMRT Link CL tools in 10.0.0 provide command-line support for creating, filtering, validating, merging, splitting, consolidating and other common operations via the `dataset` command. Detailed CL documentation and example can be found in the [pbcoretools](#) documentation. Note that in the SMRT Link CL tools tarball, the command is `dataset` not `dataset.py`.

5.4.2 Core API Support

An API is provided in [pbcore](#) (Python) and [pbbam](#) (C++) that makes consuming `DataSet XML` files or the underlying files such as BAM as burden-free as possible.

5.4.3 Other PacBio command-line tools

All bioinformatics tools that consume `DataSet XML` files should be capable of producing identical results using the equivalent BAM or FASTA file generated after applying all filters. In other words, the `DataSet XML` is not required to obtain bioinformatics results. However, *support* for the `DataSet XML` is encouraged for SMRT Analysis tools, and facilitated using the above API.

5.4.4 SMRT Link and SMRT Pipe

The DataSet XML is required for display of DataSets (such as references) in SMRT Link and for chunking in the distributed pipelines using pbsmrtpipe. Moreover, for these applications the DataSetMetadata field is mandatory, not optional.

5.5 Other topics

5.5.1 Mutability and equality

To allow user editing of attributes such as Name without affecting the underlying DataSet we define the Core DataSet as the XML with the user editable attributes (Name, Description and Tags) removed (not set to "", but absent). This Core DataSet is immutable and is the entity on which identity operations are defined. As a consequence, any modifications to fields other than Name, Description or Tags requires giving the DataSet a new universal UniqueId (aka UUID). Operations such as md5 checksum should be performed on the Core DataSet unless otherwise specified.

5.5.2 I/O trade-offs

The DataSet model defers I/O operations by replacing up-front file merges with downstream I/O operations that hit many different files. This allows consumers to avoid explicit creation of files on disk and the resulting redundant storage and costly write operations. For many uses this many-file approach is preferred to explicitly creating the file on disk, but in some cases it may be desirable to incur the cost of accessing and filtering multiple files once (e.g. to reduce disk seeks for highly fragmented DataSets). Determining when the costs outweigh the benefits will need practical investigation, but regardless the Consolidate operation provides the means for using the form of DataSet that best fits a particular use case.

5.5.3 Outstanding Issues and Future Directions

- Document FASTA filters for pbcore / pbbam for releases post-4.0.
- The propagation of subdatasets in merging can result in rather large XML files with duplicated information. It is possible this duplication could be reduced using XML IDREFs from the subdatasets to information in the top level DataSet, for ExternalResources or CollectionMetadata. This should be considered as a possible future revision.

5.6 Appendix 1: Examples satisfying the motivating use cases

- Refer to a **set of subreads** in multiple BAM files:

```
<?xml version="1.0" encoding="utf-8" ?>
<pbds:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0.bam"/>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads1.bam"/>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
```

(continues on next page)

(continued from previous page)

```

        <pbbase:Properties>
          <pbbase:Property Name="rq" Operator="gt" Value="0.75"/>
        </pbbase:Properties>
      </pbdns:Filter>
    </pbdns:Filters>
    <pbdns:DataSetMetadata>
      <pbbase:TotalLength>5000</pbbase:TotalLength>
      <pbbase:NumRecords>500</pbbase:NumRecords>
    </pbdns:DataSetMetadata>
  </pbdns:SubreadSet>

```

- Refer to a **subset of subreads** by id from one or more BAM files:

```

<?xml version="1.0" encoding="utf-8" ?>
<pbdns:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0.bam"/>
  </pbbase:ExternalResources>
  <pbdns:Filters>
    <pbdns:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="qname" Operator="==" Value="m100000/0/0_100
→"/>
      </pbbase:Properties>
    </pbdns:Filter>
    <pbdns:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="qname" Operator="==" Value="m100000/1/0_200
→"/>
      </pbbase:Properties>
    </pbdns:Filter>
  </pbdns:Filters>
  <pbdns:DataSetMetadata>
    <pbbase:TotalLength>5000</pbbase:TotalLength>
    <pbbase:NumRecords>500</pbbase:NumRecords>
  </pbdns:DataSetMetadata>
</pbdns:SubreadSet>

```

- Refer to a **subset of alignments** for input to algorithms such as Arrow (e.g. on a particular chromosome, or a particular ch

– Use an *AlignmentSet* filtered by *rname*

- Perform any analysis that can be performed on an entire file of a particular data type (reads, read regions, alignments) on a **subset of that data type** without creating a new file. - *Relies on tools using common APIs to access DataSets.*

5.7 Appendix 2: Example DataSet XML files

Here are some example XML files for each of the above DataSets

Example SubreadSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:SubreadSet
  xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  xmlns:pbmeta="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="subreadset_150304_231155"
  MetaType="PacBio.DataSet.SubreadSet"
  Name="DataSet_SubreadSet"
  Tags="barcode moreTags mapping mytags"
  Version="3.0.0"
  CreatedAt="2015-01-27T09:00:01">
<pbbase:ExternalResources>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5193"
    TimeStampedName="subread_bam_150304_231155"
    MetaType="PacBio.SubreadFile.SubreadBamFile"
    Name="First Subreads BAM"
    Description="Points to an example Subreads BAM file."
    Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5194"
        TimeStampedName="bam_index_150304_231155"
        MetaType="PacBio.Index.PacBioIndex"
        ResourceId="file:///mnt/path/to/subreads0.pbi"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResources>
    <pbbase:ExternalResource
      UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5195"
      TimeStampedName="scraps_bam_150304_231155"
      MetaType="PacBio.SubreadFile.ScrapsBamFile"
      ResourceId="file:///mnt/path/to/scraps0.bam"
      Tags="Example"
      Name="First scraps BAM"
      Description="Points to an example scraps BAM file.">
      <pbbase:FileIndices>
        <pbbase:FileIndex
          UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5196"
          TimeStampedName="bam_index_150304_231155"
          MetaType="PacBio.Index.PacBioIndex"
          ResourceId="file:///mnt/path/to/scraps0.pbi"/>
        </pbbase:FileIndices>
      </pbbase:ExternalResource>
    </pbbase:ExternalResources>
  </pbbase:ExternalResource>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5197"
    TimeStampedName="subread_bam_150304_231155"
    MetaType="PacBio.SubreadFile.SubreadBamFile"
    ResourceId="file:///mnt/path/to/subreads1.bam"
    Name="Second Subreads BAM"
    Description="Points to another example Subreads BAM file."

```

(continues on next page)

(continued from previous page)

```

    Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5198"
        TimeStampedName="bam_index_150304_231155"
        MetaType="PacBio.Index.PacBioIndex"
        ResourceId="file:///mnt/path/to/subreads0.pbi"/>
      </pbbase:FileIndices>
    <pbbase:ExternalResources>
      <pbbase:ExternalResource
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5199"
        TimeStampedName="scraps_bam_150304_231155"
        MetaType="PacBio.SubreadFile.ScrapsBamFile"
        ResourceId="file:///mnt/path/to/scraps1.bam"
        Name="Second scraps BAM" Description="Points to another example_
↳scraps BAM file."
        Tags="Example">
        <pbbase:FileIndices>
          <pbbase:FileIndex
            UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5190"
            TimeStampedName="bam_index_150304_231155"
            MetaType="PacBio.Index.PacBioIndex"
            ResourceId="file:///mnt/path/to/scraps1.pbi"/>
          </pbbase:FileIndices>
        </pbbase:ExternalResource>
      </pbbase:ExternalResources>
    </pbbase:ExternalResource>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="rq" Value="0.75" Operator=">"/>
      </pbbase:Properties>
    </pbds:Filter>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="QNAME" Value="100/0/0_100" Operator="==" />
      </pbbase:Properties>
    </pbds:Filter>
  </pbds:Filters>
  <pbds:DataSetMetadata>
    <pbds:TotalLength>500000</pbds:TotalLength>
    <pbds:NumRecords>500</pbds:NumRecords>
    <pbmeta:Collections>
      <pbmeta:CollectionMetadata
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
        TimeStampedName="collection_150304_231155"
        MetaType="PacBio.CollectionMetadata"
        Context="m152720_092723_00114_c100480560100000001823075906281381_s1_p0"
        InstrumentName="RS"
        InstrumentId="43210">
        <pbmeta:InstCtrlVer>2.3.0.0.140640</pbmeta:InstCtrlVer>
        <pbmeta:SigProcVer>NRT@172.31.128.10:8082, SwVer=2300.140640, HwVer=1.0</
↳pbmeta:SigProcVer>
        <pbmeta:RunDetails>
          <pbmeta:RunId>e903682f-e502-465c-a2b6-9dd77c9f43fc</pbmeta:RunId>
          <pbmeta:Name>beta4_130726_biotin_DEV_vs_MFG_PB11K_9458p</pbmeta:Name>

```

(continues on next page)

(continued from previous page)

```

    </pbmeta:RunDetails>
    <pbmeta:WellSample Name="Well Sample 1">
      <pbmeta:PlateId>2014-12-24_141_NGAT_Igor_bisPNA Enrichment_Mag Bead_
↳ Elution Buffers</pbmeta:PlateId>
      <pbmeta:WellName>B01</pbmeta:WellName>
      <pbmeta:Concentration>10</pbmeta:Concentration>
      <pbmeta:SampleReuseEnabled>true</pbmeta:SampleReuseEnabled>
      <pbmeta:StageHotstartEnabled>true</pbmeta:StageHotstartEnabled>
      <pbmeta:SizeSelectionEnabled>true</pbmeta:SizeSelectionEnabled>
      <pbmeta:UseCount>0</pbmeta:UseCount>
      <pbsample:BioSamplePointers>
        <pbsample:BioSamplePointer>abc2df90-d44f-4a48-9f35-3b99473c68f5</
↳ pbsample:BioSamplePointer>
      </pbsample:BioSamplePointers>
    </pbmeta:WellSample>
    <pbmeta:Automation>
      <pbbase:AutomationParameters>
        <pbbase:AutomationParameter/>
      </pbbase:AutomationParameters>
    </pbmeta:Automation>
    <pbmeta:CollectionNumber>0</pbmeta:CollectionNumber>
    <pbmeta:CellIndex>0</pbmeta:CellIndex>
    <pbmeta:CellPac Barcode="100480560100000001823075906281381"/>
    <pbmeta:Primary>
      <pbmeta:AutomationName>BasecallerV1</pbmeta:AutomationName>
      <pbmeta:ConfigFileName>1-3-0_Standard_C2.xml</pbmeta:ConfigFileName>
      <pbmeta:SequencingCondition/>
      <pbmeta:OutputOptions>
        <pbmeta:ResultsFolder>Analysis_Results</pbmeta:ResultsFolder>
        <pbmeta:CollectionPathUri>rsy://mp-rsync/vol56//RS_DATA_STAGING//
↳ 2014-12-24_141_NGAT_Igor_bisPNA%20Enrichment_Mag%20Bead%20Elution%20Buffers_1094/
↳ B01_1</pbmeta:CollectionPathUri>
        <pbmeta:CopyFiles>
          <pbmeta:CollectionFileCopy>Bam</pbmeta:CollectionFileCopy>
        </pbmeta:CopyFiles>
        <pbmeta:Readout>Bases</pbmeta:Readout>
        <pbmeta:MetricsVerbosity>Minimal</pbmeta:MetricsVerbosity>
      </pbmeta:OutputOptions>
    </pbmeta:Primary>
  </pbmeta:CollectionMetadata>
</pbmeta:Collections>
<pbsample:BioSamples>
  <pbsample:BioSample
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
    TimeStampedName="biosample_150304_231155"
    MetaType="PacBio.BioSample"
    Name="consectetur purus"
    Description="Risus sit amet lectus vehicula vulputate quisque porta_
↳ accumsan venenatis."
    CreatedAt="2015-01-20T13:27:23.9271737-08:00"/>
  </pbsample:BioSamples>
<pbd:SummaryStats>
  <pbd:AdapterDimerFraction>0.1</pbd:AdapterDimerFraction>
  <pbd:ShortInsertFraction>0.1</pbd:ShortInsertFraction>
  <pbd:NumSequencingZmws>0</pbd:NumSequencingZmws>
  <pbd:ProdDist>
    <pbbase:NumBins>4</pbbase:NumBins>

```

(continues on next page)

(continued from previous page)

```

    <pbbase:BinCounts>
      <pbbase:BinCount>1576</pbbase:BinCount>
      <pbbase:BinCount>901</pbbase:BinCount>
      <pbbase:BinCount>399</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:MetricDescription>Productivity</pbbase:MetricDescription>
    <pbbase:BinLabels>
      <pbbase:BinLabel>Empty</pbbase:BinLabel>
      <pbbase:BinLabel>Productive</pbbase:BinLabel>
      <pbbase:BinLabel>Other</pbbase:BinLabel>
      <pbbase:BinLabel>NotDefined</pbbase:BinLabel>
    </pbbase:BinLabels>
  </pbds:ProdDist>
  <pbds:ReadTypeDist>
    <pbbase:NumBins>9</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>1474</pbbase:BinCount>
      <pbbase:BinCount>799</pbbase:BinCount>
      <pbbase:BinCount>4</pbbase:BinCount>
      <pbbase:BinCount>181</pbbase:BinCount>
      <pbbase:BinCount>92</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>326</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:MetricDescription>Read Type</pbbase:MetricDescription>
    <pbbase:BinLabels>
      <pbbase:BinLabel>Empty</pbbase:BinLabel>
      <pbbase:BinLabel>FullHqRead0</pbbase:BinLabel>
      <pbbase:BinLabel>FullHqRead1</pbbase:BinLabel>
      <pbbase:BinLabel>PartialHqRead0</pbbase:BinLabel>
      <pbbase:BinLabel>PartialHqRead1</pbbase:BinLabel>
      <pbbase:BinLabel>PartialHqRead2</pbbase:BinLabel>
      <pbbase:BinLabel>Multiload</pbbase:BinLabel>
      <pbbase:BinLabel>Indeterminate</pbbase:BinLabel>
      <pbbase:BinLabel>NotDefined</pbbase:BinLabel>
    </pbbase:BinLabels>
  </pbds:ReadTypeDist>
  <pbds:ReadLenDist>
    <pbbase:SampleSize>901</pbbase:SampleSize>
    <pbbase:SampleMean>4528.69384765625</pbbase:SampleMean>
    <pbbase:SampleMed>5227</pbbase:SampleMed>
    <pbbase:SampleStd>2322.8055598026981</pbbase:SampleStd>
    <pbbase:Sample95thPct>7367</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>62</pbbase:BinCount>
      <pbbase:BinCount>39</pbbase:BinCount>
      <pbbase:BinCount>36</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>
      <pbbase:BinCount>19</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>

```

(continues on next page)

(continued from previous page)

```

    <pbbase:BinCount>32</pbbase:BinCount>
    <pbbase:BinCount>32</pbbase:BinCount>
    <pbbase:BinCount>40</pbbase:BinCount>
    <pbbase:BinCount>45</pbbase:BinCount>
    <pbbase:BinCount>54</pbbase:BinCount>
    <pbbase:BinCount>73</pbbase:BinCount>
    <pbbase:BinCount>77</pbbase:BinCount>
    <pbbase:BinCount>97</pbbase:BinCount>
    <pbbase:BinCount>95</pbbase:BinCount>
    <pbbase:BinCount>49</pbbase:BinCount>
    <pbbase:BinCount>17</pbbase:BinCount>
    <pbbase:BinCount>2</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>418.89999389648438</pbbase:BinWidth>
    <pbbase:MinOutlierValue>77</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>77</pbbase:MinBinValue>
    <pbbase:MaxBinValue>12225.1025390625</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>12644</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Length</
    ↳pbbase:MetricDescription>
  </pbds:ReadLenDist>
  <pbds:ReadQualDist>
    <pbbase:SampleSize>901</pbbase:SampleSize>
    <pbbase:SampleMean>0.82736450433731079</pbbase:SampleMean>
    <pbbase:SampleMed>0.83167940378189087</pbbase:SampleMed>
    <pbbase:SampleStd>0.029663275550147809</pbbase:SampleStd>
    <pbbase:Sample95thPct>0.86801999807357788</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>12</pbbase:BinCount>
      <pbbase:BinCount>8</pbbase:BinCount>
      <pbbase:BinCount>11</pbbase:BinCount>
      <pbbase:BinCount>10</pbbase:BinCount>
      <pbbase:BinCount>17</pbbase:BinCount>
      <pbbase:BinCount>11</pbbase:BinCount>
      <pbbase:BinCount>20</pbbase:BinCount>
      <pbbase:BinCount>24</pbbase:BinCount>
      <pbbase:BinCount>24</pbbase:BinCount>
      <pbbase:BinCount>34</pbbase:BinCount>
      <pbbase:BinCount>25</pbbase:BinCount>
      <pbbase:BinCount>38</pbbase:BinCount>
      <pbbase:BinCount>52</pbbase:BinCount>
      <pbbase:BinCount>33</pbbase:BinCount>
      <pbbase:BinCount>45</pbbase:BinCount>
      <pbbase:BinCount>48</pbbase:BinCount>

```

(continues on next page)

(continued from previous page)

```

        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>47</pbbase:BinCount>
        <pbbase:BinCount>69</pbbase:BinCount>
        <pbbase:BinCount>65</pbbase:BinCount>
        <pbbase:BinCount>55</pbbase:BinCount>
        <pbbase:BinCount>51</pbbase:BinCount>
        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>42</pbbase:BinCount>
        <pbbase:BinCount>29</pbbase:BinCount>
        <pbbase:BinCount>12</pbbase:BinCount>
        <pbbase:BinCount>3</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>0.0049052196554839611</pbbase:BinWidth>
    <pbbase:MinOutlierValue>0.7500004768371582</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>0.7500004768371582</pbbase:MinBinValue>
    <pbbase:MaxBinValue>0.89225196838378906</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>0.89715707302093506</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Quality</
→pbbase:MetricDescription>
  </pbds:ReadQualDist>
  <pbds:ControlReadLenDist>
    <pbbase:SampleSize>901</pbbase:SampleSize>
    <pbbase:SampleMean>4528.69384765625</pbbase:SampleMean>
    <pbbase:SampleMed>5227</pbbase:SampleMed>
    <pbbase:SampleStd>2322.8055598026981</pbbase:SampleStd>
    <pbbase:Sample95thPct>7367</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>62</pbbase:BinCount>
      <pbbase:BinCount>39</pbbase:BinCount>
      <pbbase:BinCount>36</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>
      <pbbase:BinCount>19</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>
      <pbbase:BinCount>32</pbbase:BinCount>
      <pbbase:BinCount>32</pbbase:BinCount>
      <pbbase:BinCount>40</pbbase:BinCount>
      <pbbase:BinCount>45</pbbase:BinCount>
      <pbbase:BinCount>54</pbbase:BinCount>
      <pbbase:BinCount>73</pbbase:BinCount>
      <pbbase:BinCount>77</pbbase:BinCount>
      <pbbase:BinCount>97</pbbase:BinCount>
      <pbbase:BinCount>95</pbbase:BinCount>
      <pbbase:BinCount>49</pbbase:BinCount>
      <pbbase:BinCount>17</pbbase:BinCount>
      <pbbase:BinCount>2</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>

```

(continues on next page)

(continued from previous page)

```

        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>418.89999389648438</pbbase:BinWidth>
    <pbbase:MinOutlierValue>77</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>77</pbbase:MinBinValue>
    <pbbase:MaxBinValue>12225.1025390625</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>12644</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Length</
    ↪pbbase:MetricDescription>
    </pbds:ControlReadLenDist>
    <pbds:ControlReadQualDist>
        <pbbase:SampleSize>901</pbbase:SampleSize>
        <pbbase:SampleMean>0.82736450433731079</pbbase:SampleMean>
        <pbbase:SampleMed>0.83167940378189087</pbbase:SampleMed>
        <pbbase:SampleStd>0.029663275550147809</pbbase:SampleStd>
        <pbbase:Sample95thPct>0.86801999807357788</pbbase:Sample95thPct>
        <pbbase:NumBins>30</pbbase:NumBins>
        <pbbase:BinCounts>
            <pbbase:BinCount>0</pbbase:BinCount>
            <pbbase:BinCount>12</pbbase:BinCount>
            <pbbase:BinCount>8</pbbase:BinCount>
            <pbbase:BinCount>11</pbbase:BinCount>
            <pbbase:BinCount>10</pbbase:BinCount>
            <pbbase:BinCount>17</pbbase:BinCount>
            <pbbase:BinCount>11</pbbase:BinCount>
            <pbbase:BinCount>20</pbbase:BinCount>
            <pbbase:BinCount>24</pbbase:BinCount>
            <pbbase:BinCount>24</pbbase:BinCount>
            <pbbase:BinCount>34</pbbase:BinCount>
            <pbbase:BinCount>25</pbbase:BinCount>
            <pbbase:BinCount>38</pbbase:BinCount>
            <pbbase:BinCount>52</pbbase:BinCount>
            <pbbase:BinCount>33</pbbase:BinCount>
            <pbbase:BinCount>45</pbbase:BinCount>
            <pbbase:BinCount>48</pbbase:BinCount>
            <pbbase:BinCount>57</pbbase:BinCount>
            <pbbase:BinCount>47</pbbase:BinCount>
            <pbbase:BinCount>69</pbbase:BinCount>
            <pbbase:BinCount>65</pbbase:BinCount>
            <pbbase:BinCount>55</pbbase:BinCount>
            <pbbase:BinCount>51</pbbase:BinCount>
            <pbbase:BinCount>57</pbbase:BinCount>
            <pbbase:BinCount>42</pbbase:BinCount>
            <pbbase:BinCount>29</pbbase:BinCount>
            <pbbase:BinCount>12</pbbase:BinCount>
            <pbbase:BinCount>3</pbbase:BinCount>
            <pbbase:BinCount>0</pbbase:BinCount>
            <pbbase:BinCount>1</pbbase:BinCount>
            <pbbase:BinCount>0</pbbase:BinCount>
            <pbbase:BinCount>1</pbbase:BinCount>
        </pbbase:BinCounts>

```

(continues on next page)

(continued from previous page)

```

    <pbbase:BinWidth>0.0049052196554839611</pbbase:BinWidth>
    <pbbase:MinOutlierValue>0.7500004768371582</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>0.7500004768371582</pbbase:MinBinValue>
    <pbbase:MaxBinValue>0.89225196838378906</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>0.89715707302093506</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Quality</
→pbbase:MetricDescription>
  </pbds:ControlReadQualDist>
  <pbds:MedianInsertDist>
    <pbbase:SampleSize>973</pbbase:SampleSize>
    <pbbase:SampleMean>513.21990966796875</pbbase:SampleMean>
    <pbbase:SampleMed>350</pbbase:SampleMed>
    <pbbase:SampleStd>575.83020626506971</pbbase:SampleStd>
    <pbbase:Sample95thPct>1322</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>714</pbbase:BinCount>
      <pbbase:BinCount>101</pbbase:BinCount>
      <pbbase:BinCount>32</pbbase:BinCount>
      <pbbase:BinCount>18</pbbase:BinCount>
      <pbbase:BinCount>13</pbbase:BinCount>
      <pbbase:BinCount>8</pbbase:BinCount>
      <pbbase:BinCount>5</pbbase:BinCount>
      <pbbase:BinCount>6</pbbase:BinCount>
      <pbbase:BinCount>7</pbbase:BinCount>
      <pbbase:BinCount>10</pbbase:BinCount>
      <pbbase:BinCount>10</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>2</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>4</pbbase:BinCount>
      <pbbase:BinCount>6</pbbase:BinCount>
      <pbbase:BinCount>3</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>7</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>3</pbbase:BinCount>
      <pbbase:BinCount>4</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>3</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>2</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>9</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>93.333335876464844</pbbase:BinWidth>
    <pbbase:MinOutlierValue>289</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>289</pbbase:MinBinValue>
    <pbbase:MaxBinValue>2995.666259765625</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>6278</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Median Insert</pbbase:MetricDescription>
  </pbds:MedianInsertDist>
  <pbds:InsertReadLenDist>

```

(continues on next page)

(continues on next page)

(continued from previous page)

```

        <pbbase:BinCount>12</pbbase:BinCount>
        <pbbase:BinCount>8</pbbase:BinCount>
        <pbbase:BinCount>11</pbbase:BinCount>
        <pbbase:BinCount>10</pbbase:BinCount>
        <pbbase:BinCount>17</pbbase:BinCount>
        <pbbase:BinCount>11</pbbase:BinCount>
        <pbbase:BinCount>20</pbbase:BinCount>
        <pbbase:BinCount>24</pbbase:BinCount>
        <pbbase:BinCount>24</pbbase:BinCount>
        <pbbase:BinCount>34</pbbase:BinCount>
        <pbbase:BinCount>25</pbbase:BinCount>
        <pbbase:BinCount>38</pbbase:BinCount>
        <pbbase:BinCount>52</pbbase:BinCount>
        <pbbase:BinCount>33</pbbase:BinCount>
        <pbbase:BinCount>45</pbbase:BinCount>
        <pbbase:BinCount>48</pbbase:BinCount>
        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>47</pbbase:BinCount>
        <pbbase:BinCount>69</pbbase:BinCount>
        <pbbase:BinCount>65</pbbase:BinCount>
        <pbbase:BinCount>55</pbbase:BinCount>
        <pbbase:BinCount>51</pbbase:BinCount>
        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>42</pbbase:BinCount>
        <pbbase:BinCount>29</pbbase:BinCount>
        <pbbase:BinCount>12</pbbase:BinCount>
        <pbbase:BinCount>3</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>0.0049052196554839611</pbbase:BinWidth>
    <pbbase:MinOutlierValue>0.7500004768371582</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>0.7500004768371582</pbbase:MinBinValue>
    <pbbase:MaxBinValue>0.89225196838378906</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>0.89715707302093506</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Quality</
    <pbbase:MetricDescription>
    </pbds:InsertReadQualDist>
  </pbds:SummaryStats>
</pbds:DataSetMetadata>
</pbds:SubreadSet>

```

Example CcsreadSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:ConsensusReadSet
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatasets.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="consensusreadset_150304_231155"
  MetaType="PacBio.DataSet.ConsensusReadSet"
  Name="DataSet_ConsensusReadSet"
  Tags="barcode moreTags mapping mytags"

```

(continues on next page)

(continued from previous page)

```

Version="2.3.0"
CreatedAt="2015-01-27T09:00:01">
<pbbase:ExternalResources>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5193"
    TimeStampedName="consensusread_bam_150304_231155"
    MetaType="PacBio.ConsensusReadFile.ConsensusReadBamFile"
    Name="First ConsensusRead BAM"
    Description="Points to an example ConsensusRead BAM file."
    ResourceId="file:///mnt/path/to/ccsreads0.bam"
    Tags="Example">
  <pbbase:FileIndices>
    <pbbase:FileIndex
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
      TimeStampedName="bam_index_150304_231155"
      MetaType="PacBio.Index.PacBioIndex"
      ResourceId="file:///mnt/path/to/ccsreads0.pbi"/>
    </pbbase:FileIndices>
  </pbbase:ExternalResource>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
    TimeStampedName="consensusread_bam_150304_231155"
    MetaType="PacBio.ConsensusReadFile.ConsensusReadBamFile"
    Name="Second ConsensusRead BAM"
    Description="Points to an example ConsensusRead BAM file."
    ResourceId="file:///mnt/path/to/ccsreads1.bam"
    Tags="Example">
  <pbbase:FileIndices>
    <pbbase:FileIndex
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5493"
      TimeStampedName="bam_index_150304_231155"
      MetaType="PacBio.Index.PacBioIndex"
      ResourceId="file:///mnt/path/to/ccsreads0.pbi"/>
    </pbbase:FileIndices>
  </pbbase:ExternalResource>
</pbbase:ExternalResources>
</pbds:ConsensusReadSet>

```

Example AlignmentSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:AlignmentSet
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  xmlns:pbbmeta="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatatypes.xsd"
  xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://
↪pacificbiosciences.com/PacBioDataModel.xsd"
  Tags="barcode moreTags mapping mytags"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="alignmentset_150304_231155"
  MetaType="PacBio.DataSet.AlignmentSet"
  Name="Uber alignment set"
  Version="2.3.0"
  CreatedAt="2015-01-27T09:00:01">
  <pbbase:ExternalResources>

```

(continues on next page)

(continued from previous page)

```

    <pbbase:ExternalResource
      Name="First Alignments BAM"
      Description="Points to an example Alignments BAM file."
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
      TimeStampedName="alignment_bam_150304_231155"
      MetaType="PacBio.AlignmentFile.AlignmentBamFile"
      ResourceId="file:///mnt/path/to/alignments0.bam" Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
        TimeStampedName="bam_index_150304_231155"
        MetaType="PacBio.Index.PacBioIndex"
        ResourceId="file:///mnt/path/to/alignments0.pbi"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResources>
    <pbbase:ExternalResource
      Name="Reference used to produce BAM file"
      Description="Points to the reference used to generate the above_
↪BAM file."
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5493"
      TimeStampedName="reference_fasta_150304_231155"
      MetaType="PacBio.ReferenceFile.ReferenceFastaFile"
      ResourceId="file:///mnt/path/to/reference.fasta" Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5593"
        TimeStampedName="sawriter_index_150304_231155"
        MetaType="PacBio.Index.SaWriterIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.sa"/>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5693"
        TimeStampedName="sam_index_150304_231155"
        MetaType="PacBio.Index.SamIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.fai"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResource>
  </pbbase:ExternalResources>
  <!-- Is this second BAM necessary? <pbbase:ExternalResource Name="Second_
↪Alignments BAM" Description="Points to another example Alignments BAM file, by_
↪relative path." MetaType="PacBio.AlignmentFile.AlignmentBamFile" ResourceId="file://
↪alignments1.bam" Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex MetaType="PacBio.Index.PacBioIndex" ResourceId=
↪"file:///mnt/path/to/alignments1.pbi"/>
    </pbbase:FileIndices>
  </pbbase:ExternalResource>-->

</pbbase:ExternalResources>
<!-- START use IDREF here for External Resources, or just duplicate them?
↪<pbdsets:DataSets>
  <pbdsets:DataSet
    UniqueId="ab95d0a3-94b8-4918-b3af-a3f81bbe519c"
    TimeStampedName="alignmentset_150304_231155"
    MetaType="PacBio.DataSet.AlignmentSet"
    Version="2.3.0"
    Name="HighQuality Read Alignments">

```

(continues on next page)

(continued from previous page)

```

        <pbbase:ExternalResources>
            <ExternalResource/>
        </pbbase:ExternalResources>
        <pbdbs:Filters> These Filters are in addition to those above. This
        ↪provides a means to subset and label the parent DataSet further.
            <pbdbs:Filter>
                <pbbase:Properties>
                    <pbbase:Property Name="rq" Value="0.85" Operator=">"/>
                </pbbase:Properties>
            </pbdbs:Filter>
        </pbdbs:Filters>
    </pbdbs:DataSet>
    <pbdbs:DataSet
        UniqueId="ac95d0a3-94b8-4918-b3af-a3f81bbe519c"
        TimeStampedName="alignmentset_150304_231155"
        MetaType="PacBio.DataSet.AlignmentSet"
        Version="2.3.0"
        Name="Alignments to chromosome 1">
        <pbbase:ExternalResources>
            <ExternalResource/>
        </pbbase:ExternalResources>
        <pbdbs:Filters>
            <pbdbs:Filter>
                <pbbase:Properties>
                    <pbbase:Property Name="RNAME" Value="chr1" Operator="==" />
                </pbbase:Properties>
            </pbdbs:Filter>
        </pbdbs:Filters>
    </pbdbs:DataSet>
</pbdbs:DataSets>
-->
<pbdbs:DataSetMetadata>
    <pbdbs:TotalLength>50000</pbdbs:TotalLength>
    <pbdbs:NumRecords>5000</pbdbs:NumRecords>
    <pbdbs:Provenance CreatedBy="AnalysisJob">
        <pbdbs:ParentTool Name="pbalign" Version="0.1.0" Description="pbalign
        ↪subreads.dataset.xml reference.dataset.xml"/>
    </pbdbs:Provenance>
</pbdbs:DataSetMetadata>
</pbdbs:AlignmentSet>

```

Example ReferenceSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbdbs:ReferenceSet
    xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
    xmlns:pbdbs="http://pacificbiosciences.com/PacBioDatasets.xsd"
    xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
    TimeStampedName="referenceset_150304_231155"
    MetaType="PacBio.DataSet.ReferenceSet"
    Name="DataSet_ReferenceSet"
    Tags="barcode moreTags mapping mytags"
    Version="2.3.0"
    CreatedAt="2015-01-27T09:00:01">

```

(continues on next page)

(continued from previous page)

```

<pbbase:ExternalResources>
  <pbbase:ExternalResource
    Name="First References FASTA"
    Description="Points to an example references FASTA file."
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
    TimeStampedName="referencefasta_150304_231155"
    MetaType="PacBio.ReferenceFile.ReferenceFastaFile"
    ResourceId="file:///mnt/path/to/reference.fasta"
    Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
        TimeStampedName="sawriter_index_150304_231155"
        MetaType="PacBio.Index.SaWriterIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.sa"/>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5493"
        TimeStampedName="sam_index_150304_231155"
        MetaType="PacBio.Index.SamIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.fai"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResource>
  </pbbase:ExternalResources>
<pbd:DataSetMetadata>
  <pbd:TotalLength>5000000</pbd:TotalLength>
  <pbd:NumRecords>500</pbd:NumRecords>
  <pbd:Organism>Tribble</pbd:Organism>
  <pbd:Ploidy>Diploid</pbd:Ploidy>
  <pbd:Contigs>
    <pbd:Contig
      Name="gi|229359445|emb|AM181176.4|"
      Description="Pseudomonas fluorescens SBW25 complete genome|quiver"
      Length="6722109"
      Digest="f627c795efad7ce0050ed42b942d408e"/>
    </pbd:Contigs>
  </pbd:DataSetMetadata>
</pbd:ReferenceSet>

```

Example ContigSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbd:ContigSet
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbd="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="contigset_150304_231155"
  MetaType="PacBio.DataSet.ContigSet"
  Name="DataSet_ContigSet"
  Tags="HGAP"
  Version="2.3.0"
  CreatedAt="2015-01-27T09:00:01"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe529c"

```

(continues on next page)

(continued from previous page)

```

        TimeStampedName="contig_fasta_150304_231155"
        MetaType="PacBio.ContigFile.ContigFastaFile"
        Name="First Contigs FASTA"
        Description="Points to an example contigs FASTA file e.g. from HGAP."
        ResourceId="file:///mnt/path/to/contigs.fasta" Tags="Example">
    </pbbase:ExternalResource>
</pbbase:ExternalResources>
<pbdbs:DataSetMetadata>
    <pbdbs:TotalLength>5000000</pbdbs:TotalLength>
    <pbdbs:NumRecords>500</pbdbs:NumRecords>
    <pbdbs:Contigs>
        <pbdbs:Contig
            Name="gi|229359445|emb|AM181176.4|"
            Description="Pseudomonas fluorescens SBW25 complete genome|quiver"
            Length="6722109"
            Digest="f627c795efad7ce0050ed42b942d408e"/>
        </pbdbs:Contigs>
    </pbdbs:DataSetMetadata>
</pbdbs:ContigSet>

```

Example BarcodeSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbdbs:BarcodeSet
    xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
    xmlns:pbdbs="http://pacificbiosciences.com/PacBioDatasets.xsd"
    xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
    TimeStampedName="barcodeset_150304_231155"
    MetaType="PacBio.DataSet.BarcodeSet"
    Name="DataSet_BarcodeSet"
    Tags="barcode moreTags mapping mytags"
    Version="2.3.0"
    CreatedAt="2015-01-27T09:00:01">
    <pbbase:ExternalResources>
        <pbbase:ExternalResource
            Name="First Barcodes FASTA"
            Description="Points to an example Barcodes FASTA file."
            UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe719c"
            MetaType="PacBio.BarcodeFile.BarcodeFastaFile"
            TimeStampedName="barcode_fasta_150304_231155"
            ResourceId="file:///mnt/path/to/barcode.fasta"
            Tags="Example"/>
        </pbbase:ExternalResources>
    <pbdbs:DataSetMetadata>
        <pbdbs:TotalLength>400</pbdbs:TotalLength>
        <pbdbs:NumRecords>30</pbdbs:NumRecords>
        <pbdbs:BarcodeConstruction>paired</pbdbs:BarcodeConstruction>
    </pbdbs:DataSetMetadata>
</pbdbs:BarcodeSet>

```

5.8 Appendix 3: Use cases from pre-2.0 SMRT Analysis satisfied by the DataSet XML files

- **Refer to a set of subreads in multiple bax.h5 files**
 - FOFN of bax.h5 files (for blasr)
 - input.xml of bax.h5 files (for SMRT Pipe)
- **Refer to a subset of subreads by id from one or more bas.h5 files**
 - Whitelist option to P_Filter to generate a FOFN of rgn.h5 files + FOFN of bas.h5 files
- **Refer to a set of alignments from multiple different references or movies**
 - Explicitly merge the alignments into a larger (cmp.h5) alignment file
 - Create FOFN of cmp.h5 files
- **Run algorithms such as HGAP on a subset of subreads (e.g. that align to a contaminant such as E. coli)**
 - Awkward, but supported indirectly through whitelist option to P_Filter
- **Run algorithms such as Quiver on a subset of alignments (e.g. on a particular chromosome, or a particular chromosome region)**
 - Command line options to Quiver (to e.g. specify a particular reference). Not currently supported on other algorithms.
- **Refer to a subset of alignments that obey certain criteria. In particular, the extractBy reference or accuracy functionality**
 - Explicit creation of cmp.h5 files using cmph5tools.py select.
- **Perform any analysis that can be performed on an entire file of a particular data type (reads, read regions, alignments) on a particular reference**
 - Not supported pre-2.0.

CHAPTER 6

Run Design CSV specification for PacBio

The Run Design CSV is a comma-separated file which can be imported into SMRT Link to create a run design. Each line in the CSV represents a sample.

Key	Value Example	Value Example
Experiment Name	NoRS_Standard_Edna.1	Ca
Experiment Id	325/3250057	Mu
Experiment Description	20170530_A6_Iguana_VVnC_SampleSheet_TEMPLATE	Ca
Run Name	20170530_A6_Iguana_VVnC_SampleSheet_TEMPLATE	Ca
System Name	Sequel	Mu
Run Comments	ecoliK12_pbi_March2013	Ca
Is Collection	TRUE	Mu
Sample Well	A01	Mu
Sample Name	SMS_Iguana_A6_3230046_A01_TestCase_SB_BindKit_ChemKitv2_8rxnKit	Ca
Cell No.	1	Mu
Sequencing Mode	CLR	Mu
Generate CCS Data	FALSE	Mu
Movie Time per SMRT Cell (hours)	5	Mu
Use Predictive Loading	TRUE	Mu
Loading Target (P1 + P2)	0.4	Mu
Maximum Loading Time (hours)	1.2	Mu
Sample Comment	SMS_Iguana_A6_3230046_A01_TestCase_SB_BindKit_ChemKit	Ca
Insert Size (bp)	2000	Mu
On-Plate Loading Concentration (pM)	5	Mu
Size Selection	FALSE	Mu
Stage Start	FALSE	Mu
Reuse Sample	FALSE	Mu
Template Prep Kit Box Barcode	DM1117100259100111716	Mu
DNA Control Complex Box Barcode	DM1234101084300123120	Mu
Binding Kit Box Barcode	DM1117100862200111716	Mu
Sequencing Kit Box Barcode	DM0001100861800123120	Mu

Table 1 – continued from previous page

Key	Value Example	Value
Wash Kit Box Barcode	DM2222100866100123120	Mu
Automation Name	Diffusion	Ca
Automation Parameters	ExtensionTime=double:60 ExtendFirst=boolean:True	Mu
Primary Analysis	Default	Ca
Primary Analysis Parameters	CopyFileTrace=boolean:true	Mu
Sample is Barcoded	TRUE	Mu
Barcode Set	dad4949d-f637-0979-b5d1-9777eff62008	Mu
Same Barcodes on Both Ends of Sequence	TRUE	Mu
Barcode Name	lbc1-lbc1	Mu
Bio Sample Name	sample1	Ca
Pipeline Id	pbsmrtpipe.pipelines.sa3_ds_isoseq3_with_genome	Mu
Analysis Name	sample1 analysis	Ca
Entry Points	PacBio.DataSet.BarcodeSet;eid_barcode;afe89e3f-17ca-e9b8-eae9-b701dbb1f02d	A
Task Options	isocollapse.task_options.allow_extra_5exon;boolean>false	A

6.1 General Requirements

The csv may only contain ASCII characters. Specifically, it must satisfy the regular expression:

- `/^[ \x00-\x7F]*$/g`

6.2 Required Fields

- Run Name
- Sample Well
- Sample Name
- Movie Time per SMRT Cell (hours)
- Insert Size (bp)
- Template Prep Kit Box Barcode
- Binding Kit Box Barcode
- Sequencing Kit Box Barcode

6.3 Is Collection

This field indicates whether the line is specifying a collection (TRUE), or a barcoded sample (FALSE). Collection lines should leave Barcode Names and Bio Sample Names blank. Barcoded sample lines only need to contain the Is Collection, Sample Name, the Barcode Name, and Bio Sample Name fields.

6.4 Experiment ID

Experiment IDs cannot contain the following characters: `<`, `>`, `:`, `"`, `\`, `|`, `?`, `*`, or `)`. Experiment IDs cannot start or end with a `/` and cannot have two adjacent `/`, i.e. `//`. Experiment IDs also cannot contain spaces.

6.9 Kit Barcodes

The kit barcodes are composed of three parts:

- Lot Number (ex: “DM1234”)
- Part Number (ex: “100-619-300”)
- Expiration Date (ex: “2020-12-31”)

which is used to make a single string. Using the above example, the barcode would be:

- DM1234100619300123120

Each kit must have a valid Part Number and cannot be obsolete. The list of kits can be found through a services endpoint such as:

- [server name]:[services port number]/smrt-link/bundles/chemistry-pb/active/files/definitions%2FPacBioAutomationConstraints.xml

This services endpoint will list, for each kit, the part numbers (“PartNumber”) and whether it is obsolete (“IsObsolete”). Dates must also be valid, meaning they must exist on the Gregorian calendar.

6.10 Parameters

The parameters are a “|” separated list. Each item follows the format: [parameter name]=[parameter type]:[parameter value]. Primary analysis parameters are:

- Readout
- MetricsVerbosity
- CopyFileTrace
- CopyFileBaz
- CopyFileDarkFrame
- CopyStatsH5

Acceptable parameter types are:

- String
- Int32
- UInt32
- Double
- Single
- Boolean
- DateTime

The parameter names and types are not case-sensitive.

6.11 Barcoded Sample Names

The barcoded sample names are a “|” separated list. Each item in the list follows the format: [barcode name];[biosample name] The barcode names must be contained within the specified barcodeset. A given barcode

name cannot appear more than once in the list. The biosample names can be any ASCII string but cannot contain the field separators “|” and “;”. The biosample names cannot be longer than 40 characters. A maximum of 384 barcodes is permitted per sample.

6.12 Auto Analysis fields

Auto Analysis is only supported on Sequel II. These fields include: Pipeline Id, Analysis Name, Entry Points, Task Options. You may define one analysis for each collection and bio sample. Pipeline Id, Analysis Name and Entry Points fields are required. The Task Options fields may be left empty, any task options not specified will use pipeline defaults.

7.1 PacBio legacy basecall File Format (bas.h5/bax.h5) Specification

The specification for bas.h5 and bax.h5 files can be found [here](#).

7.2 PacBio Alignment File Format (cmp.h5) Specification

7.2.1 Revision History

Version	Date	Authors	Comments
0.1	07/24/2009	Jason Chin	First draft
0.2	11/04/2009	Jason Chin	2nd draft, incorporated changes from prototype
0.3	11/17/2009	Susan Tang	Added consensus record
0.4	03/11/2009	Jason Chin, James Bullard	Added SF related spec and indexing proposal
0.5	07/06/2010	Dale Webster	Added PB Internal Format Spec
1.2rc	10/25/2010	Jason Chin, James Bullard, Dale Webster, Dimitris Iliopoulos, Ali Bashir	Major Revision before v. 1.2. Remove all reference to earlier Astro type cmp.h5. Meta-data group hierarchy changed. New attributes added. Define a few file operation behaviors. We call this document version 1.2rc to match the software release version for FCR. Preliminary support for strobe read timing information.
1.2	12/22/2010	Jason Chin	Finalize 1.2 spec , updated examples, revise the FileLog info group, remove TODO, remove “rc” in the version string.
1.3.1	03/6/2012	David Alexander, Mark Chaisson	QV record types changed. lastRow datasets removed. Converted to reStructuredText. Some material moved to Appendices
2.0.0	02/12/2013	David Alexander, James Bullard	Addition of chemistry tag information per-movie. Removal of master dataset constructs. Sortedness of a file now indicated by presence of OffsetTable.
2.1.0	08/01/2013	James Bullard	Addition of Barcode data
2.3.0	5/21/2014	David Alexander	Document revised chemistry encoding

7.2.2 File Format Versioning

The `cmp.h5` file format version is stored in the root group attribute `Version`. The version may take one of the following values:

- “1.2.0”
- “1.2.0.SF”
- “1.2.0.PB”
- “1.3.1.SF”
- “1.3.1.PB”
- “2.0.0”
- “2.1.0”
- “2.3.0”

File formats with versions ending in “.SF” (for Springfield) represent the production file formats that are produced by instruments at customer sites. File formats with versions ending in “.PB” (for PacBio) may contain additional information. *Version “X.PB” files are always usable wherever an “X.SF” file is usable; i.e. PacBio internal files contain a superset of the features required in a Springfield file, and the same formatting conventions are observed.*

7.2.3 Hierarchy Layout

In this section, we specify the general layout. At the top-level, or root group of the `cmp.h5` HDF5 file, there exist six HDF5 groups which must exist: `AlnInfo`, `RefInfo`, `MovieInfo`, `AlnGroup`, `RefGroup`, `FileLog`.

There are basically three different categories of data groups:

1. The *Info* groups contain information about particular aspects of the data contained in the file to some external references, e.g., reference sequences used for alignments, movies information for the reads, and ZMW hole numbers, etc. These groups will be referred to as info groups. (The only exception of such convention is the `FileLog` group. It should be considered as an Info group even though the group does have a “Info” suffix.)
2. The *Group* HDF5 groups contain information about how the data is stored in the file and function as key-value-pair mappings from integer IDs to character paths. Each “Group” HDF5 group will contain at least two datasets one of which will be called ID and the other will be called Path. The ID is the key used to refer to the HDF5 path stored in parallel in the Path dataset. To avoid ambiguity these groups will be referred to as mapping groups.
3. Additionally, at the top-level of the file, zero or more alignment data groups will exist—these groups contain the actual alignment data for each reference sequence and alignment group. These groups will be called data groups.

All datasets stored under the same HDF5 group irrespective of type shall always have the same number of rows or, in the case of dimensionless vectors, length.

Here we specify the minimal set of datasets in each of aforementioned groups:

1. An info group named `AlnInfo` containing information about each alignment stored in the file. The `AlnInfo` group should contain the following datasets:
 - a. `AlnIndex`: Dataset whose rows represent unique alignments and whose columns store relevant information about each alignment. The `AlnIndex` dataset has a string list attribute, `ColumnNames`, containing the names of the columns of this dataset.
 - b. (CCS only): A vector dataset `NumPasses`, of the same length as `AlnIndex`, indicating the number of CCS subreads that were used to generate the consensus read in the corresponding row of `AlnIndex`.
 - b. (Optional) Vector datasets, of the same length as `AlnIndex`, the same storing information about each alignment (e.g., `ZScore`, `SNR`, and `Edna`).
2. An info group named `RefInfo` containing information about the reference sequences used during alignment. The `RefInfo` group should contain the following datasets:
 - a. `ID`: Identifier of the record.
 - b. `FullName`: Name of the sequence as given by the FASTA file used during alignment.
 - c. `MD5`: md5 hashes of the DNA sequence used during alignment.

Note: The MD5 convention used in `cmp.h5` files differs from the standard convention in SAM files. SAM files store the “MD5 checksum of the sequence in the uppercase, with gaps and spaces removed.” *cmp.h5 files contain the MD5 checksums of the reference contig sequences as present in the reference FASTA file—case preserved, spaces and gaps intact (but newlines removed).*

- d. `Length`: The length of the DNA sequence used during alignment.
3. An info group named `MovieInfo` containing information about the movies which produced the alignments. This `MovieInfo` group should contain the following datasets:
 - a. `ID`: Identifier of the record.

- b. Name: Movie name.
- c. **FrameRate:** The camera speed in frames per second used to record the movie.
- d. Datasets encoding information about the sequencing chemistry that was used. This is encoded in one of two manners:
 - 1. Datasets `SequencingKit`, `BindingKit`, and `SoftwareVersion` represent the partnumbers read by the instrument barcode reader for each movie run, as well as the basecaller version. Decoding of this identifying “triple” for each movie is deferred to the tools that actually need to know the chemistry details—specifically, the Quiver variant/consensus calling tool and the base-modification identification tools.
 - 2. (*Versions 2.2.0 and earlier, and manual override in 2.3.0 and after*) Dataset `SequencingChemistry`, representing a canonical string representation (for example, “P4-C2”) of the chemistry. Note that this places the burden for decoding of the barcode information on the software that constructs the `cmp.h5` rather than client software.

Software that parses the `cmp.h5` format shall rely on the datasets in (1) as the canonical chemistry information, only falling back to the information in (2) if the datasets in (1) are absent.
- 4. An info group named `FileLog` containing information about the history of the file itself.
 - a. ID: Identifier of the record
 - b. Program: The name of the program that touches the file
 - c. Version: The version of the program that touches the file
 - d. Timestamp: A **W3C compatible timestamp** string of the date-time when the file is touched.
 - e. CommandLine: Detail command line string that details how the program is used
 - f. Log: The field to store any extra details
- 5. A mapping group named `RefGroup` that records the reference sequence information used in the alignments: The `RefGroup` group should contain the following datasets:
 - a. ID
 - b. Path
 - c. `RefInfoID`: `RefInfoID` refers to elements of the `/RefInfo/ID` dataset.
- 6. A mapping group named `AlnGroup` that records the different partitions of alignments. This data group should contain:
 - a. ID
 - b. Path
- 7. Zero or more data groups containing the actual alignments. The names of the groups are defined by the dataset `/RefGroup/Path`. Each reference group contains one or more alignment groups (representing alignments from some predefined grouping, such as: `SMRTcell`, `acquisition`, or `movie`, etc). The full HDF5 paths to the alignment groups including the group names are defined in the dataset `/AlnGroup/Path`. An alignment group should contain:
 - a. A single alignment array dataset named `AlnArray`
 - b. (Optional) Datasets for quality values and pulse features that can be aligned to the read bases. Detailed information about necessary datasets is defined in sections 10 and 11.

8. (Optional) User-defined datasets conforming to the conventions of simple HDF5 types and having the same length as each sibling in its containing group.

It may be helpful to inspect the output of *h5ls* applied to a 1.3.1.SF cmp.h5 file:

```
mp-f052:~ $ h5ls -r ~/Data/new_cmp5/alignments.cmp.h5
/
/AlnGroup          Group
/AlnGroup/ID       Dataset {1/Inf}
/AlnGroup/Path     Dataset {1/Inf}
/AlnInfo           Group
/AlnInfo/AlnIndex  Dataset {16866/Inf, 22/Inf}
/FileLog           Group
/FileLog/CommandLine Dataset {3/Inf}
/FileLog/ID        Dataset {3/Inf}
/FileLog/Log       Dataset {3/Inf}
/FileLog/Program   Dataset {3/Inf}
/FileLog/Timestamp Dataset {3/Inf}
/FileLog/Version   Dataset {3/Inf}
/MovieInfo         Group
/MovieInfo/FrameRate Dataset {1/Inf}
/MovieInfo/SequencingChemistry Dataset {1/Inf}
/MovieInfo/ID      Dataset {1/Inf}
/MovieInfo/Name    Dataset {1/Inf}
/RefGroup          Group
/RefGroup/ID       Dataset {1/Inf}
/RefGroup/OffsetTable Dataset {1/Inf, 3/Inf}
/RefGroup/Path     Dataset {1/Inf}
/RefGroup/RefInfoID Dataset {1/Inf}
/RefInfo           Group
/RefInfo/FullName  Dataset {1/Inf}
/RefInfo/ID        Dataset {1/Inf}
/RefInfo/Length    Dataset {1/Inf}
/RefInfo/MD5       Dataset {1/Inf}
/ref000001         Group
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0 Group
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/AlnArray_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/DeletionQV_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪DeletionTag Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/IPD_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪InsertionQV Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/MergeQV_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/PulseWidth_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪QualityValue Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪SubstitutionQV Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪SubstitutionTag Dataset {39434696/Inf}
```

7.2.4 Root Group Attributes

The following mandatory string attributes should be set in the root group:

Name	Allowed Values	Comment
Version	"1.2.0" "1.2.0.SF" "1.2.0.PB" "1.3.1.SF" "1.3.1.PB" "2.0.0"	The suffix is used to indicate whether the file includes (".SF") or does not include (".PB") several datasets useful for in-house analyses.
Read-Type	"RCCS", "CCS", "strobe", "standard", or "cDNA"	Set to "standard" by default. If the cmp.h5 is used for "RCCS" and "CCS", there will be no pulse features. Each read type will allow different sets of optional tables.
Command-Line	The command line used for generating this file.	This attribute is reserved for the initial generation. All post-initial alignment information should be stored in FileLog

7.2.5 Mapping Groups: ID, and Path datasets

Each mapping group contains at least an `ID` and `Path` dataset. The `ID` dataset contains unique positive integer values. The `Path` dataset contains proper HDF5 paths to HDF5 groups within the file. Elements of the path dataset should conform to the following regular expression (leading forward slash not included):

"[a-zA-Z+0-9]+" (all lower and upper case ASCII characters, numbers, "+", and "0-9").

The `ID`, `Path` datasets function as key-value pair mappings. The individual IDs are used in datasets to reference the relevant information stored in this particular mapping group.

The following **HDF5 DDL** defines the hdf5 data types for these data sets:

```

DATASET "ID" {
    DATATYPE  H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
}
DATASET "Path" {
    DATATYPE  H5T_STRING {
        STRSIZE H5T_VARIABLE;
        STRPAD  H5T_STR_NULLTERM;
        CSET    H5T_CSET_ASCII;
        CTYPE   H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
}

```

Two datasets are used to avoid compound types in an HDF5 file. This avoids the complication in reader/writer code implementations. If there is a mature compound type code base within the PBI development environment, compound type datasets are recommended for storing such key-value pairs.

7.2.6 RefGroup data group and /RefGroup/* datasets

The `RefGroup` mapping group provides a mapping between reference sequence identifiers (`ID`) to HDF5 paths in the file (`Path`). An example HDF5 schema can be seen above. A `RefInfoID` data set is used for pointing to the `ID` dataset in the `RefInfo` group and can be viewed as a foreign key.

The following DDL code block defines the data types for the datasets and attributes associated with `/RefGroup`:

```

GROUP "RefGroup" {
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Path" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "RefInfoID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}

```

7.2.7 AlnGroup data group: /AlnGroup/* datasets

The AlnGroup mapping group provides a mapping between alignment group identifiers (ID) to alignment group paths.

The following DDL code block defines the data types for the datasets and attributes associated with /AlnGroup:

```

GROUP "AlnGroup" {
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Path" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}

```

7.2.8 RefInfo info group and /RefInfo/* datasets

The RefInfo info group provides information about the reference sequences used during alignment. The RefInfo group contains at least 4 datasets including the ID dataset. The RefInfo/FullName provides the name of the sequence aligned to and is the full FASTA name. The RefInfo/MD5 is an MD5 hash of the reference sequence aligned to. The RefInfo/Length provides the length of the sequence aligned to.

Other sequence specific annotations can be stored as parallel datasets at this level.

The following DDL code block defines the data types for the datasets and attributes associated /RefInfo:

```
GROUP "RefInfo" {
  DATASET "FullName" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Length" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "MD5" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}
```

7.2.9 MovieInfo data group: MovieInfo/* datasets

The paired arrays MovieInfo/ID and MovieInfo/Name in the MovieInfo group are defined to indicate the source of the movies for the reads in the AlnInfo/AlnIndex dataset. This pair of arrays functions as a key-value-pair map between IDs and movie names.

The following DDL code block defines the data types for the datasets and attributes associated /MovieInfo:

```
GROUP "MovieInfo" {
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Name" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}
```

7.2.10 AlnInfo data group and the AlnArray data sets

AlnInfo data group

The first column of the `AlnIndex` can be treated as the equivalent “ID” dataset in the mapping or the info groups.

The data types of the dataset `AlnIndex` are defined as:

```

DATASET "AlnIndex" {
  DATATYPE  H5T_STD_U32LE
  DATASPACE SIMPLE { ( *, 22 ) / ( H5S_UNLIMITED, 22 ) }
}

```

AlnIndex dataset

The purpose of the `AlnIndex` dataset is to:

1. Store the information necessary to retrieve alignments from the file. This includes: path, beginning offset, and ending offset within the dataset containing the alignment. (This kind of reference to alignment is similar to that proposed by HDF5 group in the bioHDF5 specification.)
2. Store the information, e.g., the orientation (i.e., strand) of the alignment, for processing the alignment properly for downstream bioinformatics analysis and visualization.
3. Store information that can be used to indentify the original reads.
4. Store the unique unsigned 32 bit integer ID as single unique key for each individual alignment.
5. Store summary information about the alignment. For example, one can store the number of matches, mismatches, insertions, deletions, mapping quality, read level quality values, etc.

AlnIndex Dataset Columns

The 22 columns in the `AlnIndex` dataset are described in the table below.

Column Name	Meaning	Comment
AlnID	Non-zero unique 32 bit integer key for the alignment record	Each alignment should have a unique AlnID. No other assumption about the order of the AlnID should be used for data processing.
AlnGroupID	A foreign key referring to AlnGroup/ID	
MovieID	A foreign key referring to MovieInfo/ID	
RefGroupID	A foreign key referring to RefGroup/ID.	
tStart	The start position (0-based, inclusive) of the alignment target (the reference sequence)	tStart should always be less than tEnd, even when the hit is against the opposite strand.
tEnd	The end position (0-based, not-inclusive) of the alignment target (the reference sequence)	tEnd should always be greater than tStart, even when the hit is against the opposite strand.
RCRefStrand	The relative strand in the alignment. 1 for reversed reference strand; 0 for forward-forward alignment	The read base should never be reverse-complimented in the alignment array, so we only need to record if the reference bases are presented in reverse complemented strand in the file. “1” means “Yes/True” here.
HoleNumber	The HoleNumber from the bas.h5	
SetNumber		
StrobeNumber ExonNumber	Context dependent value. When the read type is Strobe, this field is the strobe number. When the read type is cDNA it will be the exon number.	
MoleculeID	An integer which is unique to all subreads from the same ZMW.	If multiple subreads are from the same physical origin, they should have the same MoleculeID and different physical origins should have different MoleculeID.
rStart	The start position (0-based, inclusive) of the read in the alignment	Regardless weather the alignment is a subread or not, the position is always relative to the original raw full read sequence.
rEnd	The end position (0-based, not-inclusive) of the read in the alignment	rEnd should always be greater than rStart.
MapQV	TBD	
nM	Number of matched base in the alignment	
nMM	Number of mis-matched base in the alignment	
nIns	Number of insertions in the read relative to the reference sequence	
nDel	Number of deletions (missing bases) in the read relative to the reference sequence	
Offset_begin	The beginning position (0-based, inclusive) of the alignment in the AlignmentArray	
Offset_end	The ending position (0-based, exclusive) the alignment in the AlignmentArray	Not including the padded zero of the alignment array.
nBackRead	Used for faster access to blocks of sorted reads	See the sorting and indexing section
nReadOverlap	Used for faster access to blocks of sorted reads	See the sorting and indexing section

The column names should be stored as an attribute `ColumnNames` that contains all names listed in “Column Name” in the table above.

7.2.11 Sequence Alignments

Binary Encoding for Alignment Pair

The *alignment array* is a one dimensional 8 bit unsigned integer array where the individual array elements represent a “read base - reference base” pair packed into one byte. The higher four bits are set by the read base and the lower four bits are set by the reference base as the following:

```
0 0 0 0 0 0 0 0
T G C A T G C A
```

For example, “T” and “T” matched alignment will be presented as 0b10001000=136. “T” vs. “G” mismatch will be represented as 0b10000100=132. Insertion of “T” in read will be 0b10000000=128. “No-call” (“N”) bases are encoded as 0b1111=15 for both read and reference.

In the *AlnArray* dataset, the encoded read base should be always the same as what has been observed by the sequencing machine without any complementation. If a read is aligned to the reverse complement strand of the reference sequence, the lower four bits represent the complemented base (i.e., the reference has been complemented).

Alignment Array

The example below shows the conversion of an alignment pair to the binary array represented as an integer:

Alignment:

```
Read Bases: ATCTT--ATC-GTTAATTA--A
Ref. Bases: A-CTCAGA-CAGTCAATTAGCA
```

Encoded Alignment Pairs:

```
AA -> 17
T- -> 128
CC -> 34
TT -> 136
TC -> 130
-A -> 1
-G -> 4
...
-C -> 2
AA -> 17
```

The final encoded array for this alignment is [17, 128, 34, 136, 130, 1, 4, 17, 128, 34, 1, 68, 136, 130, 17, 17, 136, 136, 17, 4, 2, 17, 0].

Note that zero is padded at the end of each alignment as a separator between different alignments. This will enable some analysis by simply streaming the alignment array without extra index look-ups to separate different alignments.

The alignment array is a concatenation of all encoded alignment arrays of each read and the *AlignmentIndex* dataset is used to identify the origin of each alignment.

Below is an example of the HDF5 type definition for an *AlnArray*:

```
DATASET "AlnArray" {
    DATATYPE H5T_STD_U8LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
}
```

7.2.12 Pulse Metrics and QVs

In addition to the basic and required `AlnArray` dataset present in each alignment group, pulse metrics and quality values (QVs) may be optionally provided; however if one of these features is provided for one alignment group they must be provided for all alignment groups. These optional datasets are:

- `DeletionQV`,
- `DeletionTag`,
- `InsertionQV`,
- `MergeQV`,
- `SubstitutionQV`,
- `SubstitutionTag`,
- `QualityValue`,
- `IPD`,
- `PulseWidth`,
- `StartFrame`,
- `pkmid`

Each such dataset is of the same shape as the `AlnArray` dataset in the same alignment group. Missing values (corresponding to read gaps in the alignment array) are encoded based on the type of the dataset:

Data type	Missing value encoding
<code>float32</code>	NaN
<code>int8 (char)</code>	'-' (ASCII 42)
<code>uint8</code>	255
<code>uint16</code>	65535

A missing value is present at a dataset offset if and only if that offset corresponds to a read gap in the `AlnArray`.

For the types of the pulse metric and QV datasets, see [Summary of Attributes and Datasets](#). Any offset into a pulse metric or QV dataset corresponds to the same offset in the `AlnArray`.

7.2.13 Specification for the `cmp.h5` used for automatic data analysis from the instrument

This section defines the constraints that a `cmp.h5` file should satisfy for automatic data analysis for an Springfield instrument. Such files are labeled with a root group attribute `Version` of “1.2.0.SF” or “1.3.1.SF”.

The `RefGroup/Path` for 1.2.0.SF and 1.2.0.PB `cmp.h5` files has the form of “ref%06d” (C string formatting convention). The original FASTA sequence header should be stored in the `RefInfo/FullName` dataset. Additionally, two other datasets are obligatory: `RefInfo/Length` and `RefInfo/MD5`.

The default of `AlnGroup` partition is to group alignments from the same movie that aligned to the same reference together and we use the movie filename without suffix as the default alignment group name.

7.2.14 Specification for the `cmp.h5` used for PacBio internal data analysis

In addition to all datasets specified for the standard `cmp.h5` the following additional datasets are required in internal files (“1.2.1.PB”):

1. Within the info group named “MovieInfo” containing information about the movies which produced the alignments:

- Exp: A uint32 dataset specifying the PacBio LIMS Experiment code associated with each movie in the corresponding /MovieInfo/Name dataset.
- Run: A uint32 dataset specifying the PacBio LIMS Run code associated with each movie in the corresponding /MovieInfo/Name dataset.

Data type and data space definition:

```

DATASET "/MovieInfo/Exp" {
  DATATYPE  H5T_STD_U32LE
  DATASPACE SIMPLE { ( 1 ) / ( H5S_UNLIMITED ) }
}
DATASET "/MovieInfo/Run" {
  DATATYPE  H5T_STD_U32LE
  DATASPACE SIMPLE { ( 1 ) / ( H5S_UNLIMITED ) }
}

```

2. Within the info group named AlnInfo containing information about each alignment stored in the file:

- ZScore: a float32 dataset containing the alignment significance score (“Z Score”) computed from the corresponding row of the /AlnInfo/Index table.

Data type and data space definition:

```

DATASET "/AlnInfo/ZScore" {
  DATATYPE  H5T_IEEE_F32LE
  DATASPACE SIMPLE { ( 310 ) / ( H5S_UNLIMITED ) }
}

```

3. In addition to all attributes specified for the standard cmp.h5 the following additional root level attributes are required:

Attribute name	Type	Sample values	Comment
ReportsFolder	string	“Analysis_Reports”	Contains the directory name of the Primary Analysis Reports used for this alignment
PrimaryPipeline	string	“61453”	Contains the Perforce changelist number of the Primary Analysis Pipeline used for this alignment

7.2.15 Sorting, Flattening, Merging, Splitting and Filtering Behaviors

Sorting

In order to provide fast access to cmp.h5 files, we provide sorted cmp.h5 files. These files have some additional information to quickly retrieve contiguous regions according to an indexing scheme. The most typical use case is to obtain a set of reads overlapping a particular genomic region, where the region can be a single genomic coordinate or ranges of genomic coordinates. Note that by default, *sorting* only entails the sorting of the AlnIndex dataset, and not the sorting of the alignment data itself.

A sorted cmp.h5 file has the following additional items as compared to an unsorted cmp.h5:

1. A dataset OffsetTable stored within the RefGroup mapping group giving the offsets of the reads mapped to a reference sequences in the global alignment index. The dataset is a 3 by N unsigned 32 bit unsigned integer array, where N is the total number of reference sequences in the RegGroup/ID table. The three elements of each row in the array indicate the RefID, targetStartOffset, and targetEndOffset.

The `targetStartOffset` and `targetEndOffset` give the range of the reads in the global `/AlnInfo/AlnIndex` that maps to the specific reference sequence in the first column of the dataset. The presence or absence of the `OffsetTable` dataset should be used to determine whether the file is sorted or unsorted.

2. The alignment index will have two additional columns of unsigned 32-bit integers (these could be shorter) `nBackRead` and `nReadOverlap` which gives the maximum number of reads one needs to examine to determine overlap and the actual number of reads which overlap a position, respectively. A value of -1 indicates that the field has not been filled in, whereas a value of 0 means that no further reads possibly overlap the position of interest. Here, `nBackRead > nReadOverlap` is always true.
3. In addition to sorting the `AlnIndex`, sorting and indexing can perform a “flattening” operation whereby all `AlnGroups` under each `RefGroup` are merged into a single `AlnGroup`. The name of the single `AlnGroup` can be anything, however, convention is to use the name: “rg-0001” to indicate that the sub-datasets have been merged and re-ordered. Additionally, an attribute on this group: `repacked` will be set to 1 to indicate, irrespective of the name, that the datasets have been sorted. If the length of any of the child datasets of a “repacked” alignment group would be greater than 2^{32} , then additional alignment groups are added serially, e.g., “rg-0002”, etc. An alignment will never span more than one alignment group.

Note: The time complexity of sorting a `cmp.h5` file will be on the order of $O(n \log(n))$. Additionally, the columns `nBackRead` and `nReadOverlap` need to be computed. This will be on the order of $O(\max(\text{read length}) * n)$. Access to a given start position in `cmp.h5` will be $O(\log(n))$, however, this will only produce reads having that start position. In order to obtain all reads overlapping a position, one needs to inspect the `nBackRead` to obtain the size of the slice that they should grab from the `cmp.h5` file. Retrieval, therefore, is bound by $O(n\text{BackRead} \log(n))$. The additional column, `nReadOverlap`, should allow one to obtain significantly better performance, as the search can stop once to obtained number of reads is equal to `nReadOverlap`.

Merging

Merging is performed on a list of `cmp.h5` files by selecting the first file to act as the seed and sequentially merging the rest onto the seed. If the first file in the list of files to be merged is empty then the next non-empty file is selected to act as the seed. An exact copy of the seed is made where all ID-type datasets have their entries serialized to consecutive 32 bit integers starting from 1. Merging results modifies a copy of the seed file in place. For each `cmp.h5` file in the merging list, the following steps are performed:

1. If the file is empty, its root group `Version` does not match the seed’s `Version` or does not have the same type of loaded `PulseMetrics` as the seed, it is removed from the merging list and the next file is considered.
2. Root group attributes are not merged since they are set to the seed’s Root group attributes.
3. Datasets under the seed’s `AlnInfo` Data group are extended with their counterparts from the file to be merged. The `AlnID` column of the newly added rows in the `AlnInfo/Index` is updated by resetting the old values from the merged file. The new values are set equal to a list of integers starting from the maximum `AlnID` of the seed + 1, adding 1 for each new `AlnID` from the merged file.
4. Datasets under the seed’s `RefInfo`, `MovieInfo`, `AlnGroup` and `RefGroup` data groups are extended only with new entries from their counterparts in the file to be merged. If new `RefInfo/ID`, `RefGroup/ID` or `MovieInfo/ID` entries are created, they are mapped back to their respective columns in the `AlnInfo/Index`.

After going through the entire list of files to be merged, the `FileLog` attribute from the Root group attributes is modified (TBD).

Splitting

The current splitting behavior is implementation specific and associated with a single use case, i.e., processing of `.cmp.h5` files involved in Edna analysis- type workflows. It is our aim to generalize the splitting behavior to accom-

modate more use cases when those become available.

A master cmp.h5 file is split into an N number of cmp.h5 files where N is equal to the number of RefInfo/ID entries in the master file. Consequently, each new cmp.h5 file contains all data associated with a single reference sequence. This is done by:

1. Creating N copies of the master cmp.h5 file and sequentially selecting a RefInfo/ID entry to become the only entry for each copied file, unique amongst the group.
2. Resizing all datasets belonging to AlnInfo, RefInfo, MovieInfo, AlnGroup and RefGroup by deleting all entries that are not associated with the chosen reference sequence. Splitting maintains the values of all ID-type fields and data fields in the AlnInfo/Index rows.
3. Maintaining the size and content of the AlnArray and PulseMetric-type datasets in the new files as the ones in the master.

7.2.16 Barcode Information

In addition to the aforementioned core alignment information, the cmp.h5 file can be used to store optional datasets containing barcode annotation on alignments. The pattern leveraged to store this annotation demonstrates a general mechanism to extend the information stored in the cmp.h5 file for downstream applications.

In the case of barcoding, we wish to label alignments according to their barcode so that other applications can leverage this information when computing statistics over sets of alignments, e.g., consensus calling within sample. To this end, a parallel dataset to /AlnInfo/AlnIndex is created. The Barcode dataset is 32-bit integer matrix with the same number of rows as the AlnIndex dataset and 5 columns storing scoring and labeling information.

The Barcode dataset contains the total number of barcodes scored for this molecule (count), the index of the top-scoring barcode (index1), the score of the top-scoring barcode (score1), the index of the 2nd-highest scoring barcode (index2) and its score (score2). These columns are named in the attribute ColumnNames of the Barcode dataset.

The index1 and index2 are foreign-keys into the BarcodeInfo/ID dataset. Analogous to the other *Info datasets, the BarcodeInfo/ID and BarcodeInfo/Name are used to retrieve the human-readable name of the barcode.

7.2.17 Summary of Attributes and Datasets

Versions prior to 2.0.0 are described in the Appendices.

File Version 2.0.0 contents:

Parent Group	HDF5 data	Resource Name	Data type	Shape	
/	ATTR	CommandLine	VLEN_STR	None	required
/	ATTR	Index	VLEN_STR	(3,)	optional
/	ATTR	ReadType	VLEN_STR	None	required
/	ATTR	Version	VLEN_STR	None	required
/AlnGroup	DS	ID	uint32	1	required
/AlnGroup	DS	Path	VLEN_STR	1	required
/AlnInfo	DS	AlnIndex	uint32	22	required
/AlnInfo	ATTR	ColumnNames	VLEN_STR	22	required
/FileLog	DS	CommandLine	VLEN_STR	1	required
/FileLog	DS	ID	uint32	1	required
/FileLog	DS	Log	VLEN_STR	1	required

Continued on next page

Table 1 – continued from previous page

Parent Group	HDF5 data	Resource Name	Data type	Shape	
/FileLog	DS	Program	VLEN_STR	1	required
/FileLog	DS	Timestamp	VLEN_STR	1	required
/FileLog	DS	Version	VLEN_STR	1	required
/MovieInfo	DS	ID	uint32	1	required
/MovieInfo	DS	Name	VLEN_STR	1	required
/MovieInfo	DS	FrameRate	float32	1	required
/MovieInfo	DS	SequencingChemistry	VLEN_STR	1	required
/ref*/*	DS	AlnArray	uint8	1	required
/ref*/*	DS	QualityValue	uint8	1	optional
/ref*/*	DS	DeletionQV	uint8	1	optional
/ref*/*	DS	InsertionQV	uint8	1	optional
/ref*/*	DS	MergeQV	uint8	1	optional
/ref*/*	DS	SubstitutionQV	uint8	1	optional
/ref*/*	DS	SubstitutionTag	char	1	optional
/ref*/*	DS	DeletionTag	char	1	optional
/ref*/*	DS	IPD	uint16	1	optional
/ref*/*	DS	PulseWidth	uint16	1	optional
/ref*/*	DS	PulseIndex	uint32	1	optional
/RefGroup	DS	ID	uint32	1	required
/RefGroup	DS	OffsetTable	uint32	3	optional
/RefGroup	DS	Path	VLEN_STR	1	required
/RefGroup	DS	RefInfoID	uint32	1	required
/RefInfo	DS	FullName	VLEN_STR	1	required
/RefInfo	DS	ID	uint32	1	required
/RefInfo	DS	Length	uint32	1	required
/RefInfo	DS	MD5	VLEN_STR	1	required
/BarcodeInfo	DS	ID	uint32	1	optional
/BarcodeInfo	DS	ID	uint32	1	required
/BarcodeInfo	DS	Name	VLEN_STR	1	required

CHAPTER 8

APIs available

We occasionally make changes to these file format specifications so we recommend using PacBio-authored APIs to access these file types.

- C++: [pbbam](#)
- Python: [pbcore](#)

For completeness, here is the PacBio data model XSD.

9.1 PacBioDataModel

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2016 (x64) (http://www.altova.com) by
efayad@pacificbiosciences.com (Pacific Biosciences) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://
pacificbiosciences.com/PacBioDataModel.xsd" xmlns:pbbase="http://pacificbiosciences.
com/PacBioBaseDataModel.xsd" xmlns:pbpn="http://pacificbiosciences.com/
PacBioPartNumbers.xsd" xmlns:pbdmeta="http://pacificbiosciences.com/
PacBioCollectionMetadata.xsd" xmlns:pbds="http://pacificbiosciences.com/
PacBioDatasets.xsd" xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.
xsd" targetNamespace="http://pacificbiosciences.com/PacBioDataModel.xsd"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioCollectionMetadata.
xsd" schemaLocation="PacBioCollectionMetadata.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioDatasets.xsd"
schemaLocation="PacBioDatasets.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioPartNumbers.xsd"
schemaLocation="PacBioPartNumbers.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
schemaLocation="PacBioSampleInfo.xsd"/>
  <xs:element name="Assay" type="AssayType"/>
  <xs:element name="ChipLayout">
    <xs:annotation>
      <xs:documentation>Part of the RunResources; specifies a
ChipLayout which is compatible with the collection protocols defined on the plate</
xs:documentation>
```

(continues on next page)

(continued from previous page)

```

</xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element ref="Validation" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="Name" type="xs:string" use="required"/>
  <xs:attribute name="PartNumber" type="xs:string" use="required
↪"/>

  <xs:attribute name="Quantity">
    <xs:annotation>
      <xs:documentation>The number of cells_
↪required, of a particular part number</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
</xs:element>
<xs:element name="CompatibleChipLayouts">
  <xs:annotation>
    <xs:documentation>A set of Chip Layouts deemed compatible_
↪with the current plate</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="ChipLayout" maxOccurs="unbounded"/>
      <xs:element ref="Validation" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="CompatibleSequencingKits">
  <xs:annotation>
    <xs:documentation>A set of reagent kits deemed compatible_
↪with the current plate</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="pbbn:SequencingKit"/>
      <xs:element ref="RequiredTips"/>
      <xs:element ref="EstimatedTotalRunTime"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="EstimatedTotalRunTime">
  <xs:annotation>
    <xs:documentation>The total amount of time the run is_
↪estimated to require. A confidence value (defaulted to 90%) indicates the degree_
↪of certainty associated with the estimate</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Validation" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Value" type="xs:string" use="required"/>
    <xs:attribute name="Confidence" type="xs:int" use="required"/>
  </xs:complexType>
</xs:element>
<xs:element name="Events" type="pbbase:RecordedEventType"/>
<xs:element name="Input" type="pbbase:InputOutputDataType"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="Output" type="pbbase:InputOutputDataType"/>
<xs:element name="Parameter">
  <xs:annotation>
    <xs:documentation>A variable, as a name/value pair,
↪ associated with a protocol (one of Collection, Primary, and Secondary)</
↪ xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Validation" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Name" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value=
↪ "AverageReadLength"/>
          <xs:enumeration value="AcquisitionTime
↪ "/>
          <xs:enumeration value="InsertSize"/>
          <xs:enumeration value="ReuseComplex"/>
          <xs:enumeration value="StageHS"/>
          <xs:enumeration value="JobId"/>
          <xs:enumeration value="JobName"/>
          <xs:enumeration value=
↪ "NumberOfCollections"/>
          <xs:enumeration value="StrobeByTime"/>
          <xs:enumeration value="UsedControl"/>
          <xs:enumeration value="Use2ndLook"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="Value" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>
<xs:element name="PacBioDataModel">
  <xs:annotation>
    <xs:documentation>PacBio Data Model root element</
↪ xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ExperimentContainer" type=
↪ "ExperimentContainerType"/>
      <xs:any minOccurs="0">
        <xs:annotation>
          <xs:documentation>By using the "any"
↪ element we can extend (after "ProjectContainer") the content of "PacBioDataModel"
↪ with any element</xs:documentation>
        </xs:annotation>
      </xs:any>
    </xs:sequence>
    <xs:attribute name="Version" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation>An optional identifier
↪ denoting the revision of this particular entity</xs:documentation>
      </xs:annotation>
    </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

        </xs:complexType>
      </xs:element>
      <xs:element name="RequiredSMRTCells">
        <xs:annotation>
          <xs:documentation>Part of the RunResources; specifies the_
↪required number of SMRT cells</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Quantity" type="xs:int" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="RequiredTips">
        <xs:annotation>
          <xs:documentation>Part of the RunResources; specifies the_
↪required number of tips via two attributes, Left and Right</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Left" type="xs:int" use="required"/>
          <xs:attribute name="Right" type="xs:int" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="RunResources">
        <xs:annotation>
          <xs:documentation>This is an output field specifying the_
↪requirements for the run, e.g. number of tips, estimated run time, etc.</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="CompatibleSequencingKits" maxOccurs=
↪"unbounded"/>
            <xs:element ref="CompatibleChipLayouts"/>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SampleComment">
        <xs:annotation>
          <xs:documentation>A general sample description</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType mixed="true">
          <xs:sequence>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Value" type="xs:string"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="Validation">
        <xs:annotation>
          <xs:documentation xml:lang="en">

```

(continues on next page)

(continued from previous page)

A validation type which is an element/part of every other element in the schema. It is used to communicate validation issues as part of the output.

```

</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attribute name="IsValid" type="xs:boolean" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Indicates whether or not the element is valid. The assumption is that the
          Validation element is omitted unless the element is invalid, in which
          case,
            the Validation element would describe the problem.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="ID" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          An identifier which can be used by client applications to translate/map
          to a human decipherable message.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Source" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          This is the element which has experienced a validation issue.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="PlateId"/>
        <xs:enumeration value="PlateDefinition"
          </xs:enumeration value="SchemaVersion"/
          <xs:enumeration value="DefType"/>
          <xs:enumeration value="Owner"/>
          <xs:enumeration value="CreatedBy"/>
          <xs:enumeration value="Comments"/>
          <xs:enumeration value="OutputPath"/>
          <xs:enumeration value="Collections"/>
          <xs:enumeration value="Collection"/>
          <xs:enumeration value=
            <xs:enumeration value="RunResources"/>
            <xs:enumeration value=
              <xs:enumeration value="ChipLayout"/>
              <xs:enumeration value=
                <xs:enumeration value="SequencingKit"/
                <xs:enumeration value="RequiredTips"/>
                <xs:enumeration value=
          </xs:enumeration value="EstimatedTotalRunTime"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:complexType>
  </xs:annotation>
</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

    <xs:enumeration value=
    <xs:enumeration value=
    <xs:enumeration value="Basecaller"/>
    <xs:enumeration value=
    <xs:enumeration value="WellNo"/>
    <xs:enumeration value="SampleName"/>
    <xs:enumeration value="Barcode"/>
    <xs:enumeration value="AcquisitionTime
    <xs:enumeration value="InsertSize"/>
    <xs:enumeration value="ReuseComplex"/>
    <xs:enumeration value="StageHS"/>
    <xs:enumeration value=
    <xs:enumeration value="Confidence"/>
    <xs:enumeration value="SampleComment"/
    <xs:enumeration value="StrobeByTime"/>
  </xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="ElementPath" type="xs:string" use=
  <xs:annotation>
    <xs:documentation xml:lang="en">
      An optional string attribute which holds the path to the offending
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="SupplementalInfo" type="xs:string" use=
  <xs:annotation>
    <xs:documentation xml:lang="en">
      An optional string attribute which holds extraneous information.
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:complexType name="AssayType">
  <xs:complexContent>
    <xs:extension base="pbbase:DataEntityType">
      <xs:sequence>
        <xs:element ref="SubreadSets"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ExperimentContainerType">
  <xs:annotation>
    <xs:documentation>A composite object type that can encompass
  </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element ref="SubreadSets"/>
  </xs:sequence>
</xs:complexType>

```

multiple runs, possibly across multiple instruments.

(continues on next page)

(continued from previous page)

One use case may be that a user may have a large genome they'd like to sequence, and it may take multiple runs on multiple instruments, to get enough data. Another use case may be that a user has multiple samples of the same phenotype which they would like to analyze in a similar fashion/automation, and as such these samples are run as part of one experiment.

The experiment object is intended to be packagable, such that the metadata of all acquisitions within is contained.

```

</xs:documentation>
</xs:annotation>
<xs:complexContent>
  <xs:extension base="pbbase:BaseEntityType">
    <xs:sequence>
      <xs:element name="InvestigatorName" type=
        "xs:string" minOccurs="0">
          <xs:annotation>
            <xs:documentation>An optional
PI name</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="CreatedDate" type="xs:date">
          <xs:annotation>
            <xs:documentation>
Automatically generated creation date</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Runs" minOccurs="0">
          <xs:annotation>
            <xs:documentation>Multiple
acquisitions from different instrument runs</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Run
" type="RunType" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="DataSets" minOccurs="0">
          <xs:annotation>
            <xs:documentation>Pointers to
various data elements associated with the acquisitions</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element ref=
"pbds:DataSet" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="RecordedEvents" minOccurs="0
">
          <xs:annotation>
            <xs:documentation>Journal of
metrics, system events, or alarms that were generated during this container's
lifetime</xs:documentation>
          </xs:annotation>
        </xs:complexType>
    </xs:sequence>
  </xs:extension>
</xs:complexContent>

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
          <xs:element name=
↪ "RecordedEvent" type="pbbase:RecordedEventType" minOccurs="0" maxOccurs="unbounded">
↪ <xs:annotation>
↪ <xs:documentation>Journal of metrics, system events, or alarms that were generated
↪ during this container's lifetime</xs:documentation>
          </
↪ xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="BioSamples" minOccurs="0">
    <xs:complexType>
      <xs:sequence>
        <xs:element name=
↪ "BioSample" type="pbsample:BioSampleType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:sequence>
<xs:attribute name="ExperimentId" type="xs:string"/>
<xs:attribute name="TimeStampedName" type="xs:string">
  <xs:annotation>
    <xs:documentation>This is NOT
↪ intended to be used as a unique field. For uniqueness, use UniqueId. In order to
↪ not utilize customer provided names, this attribute may be used as an alternative
↪ means of Human Readable ID, e.g. instrumentId-Run-150304_231155</xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="RunType">
  <xs:annotation>
    <xs:documentation>A run is defined as a set of one or more
↪ data collections acquired in sequence on an instrument. A run specifies the wells
↪ and SMRT Cells to include in the sequencing run, along with the collection and
↪ analysis automation to use for the selected wells and cells.
  </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="pbbase:StrictEntityType">
      <xs:sequence>
        <xs:element name="Outputs" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref=
↪ "Output" minOccurs="0" maxOccurs="unbounded"/>
              <xs:element ref=
↪ "SubreadSets" minOccurs="0">
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
</xs:annotation>

```

↪ <xs:documentation>When a run definition is created, the placeholder for (continues on next page)
 ↪ is created as well. As part of that, the collection metadata will reside within
 ↪ this output placeholder.

(continued from previous page)

```

There will be one SubreadSet placeholder created per collection.</xs:documentation>
    </
    <xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Inputs" minOccurs="0">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Input
    <xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element ref="Assay" minOccurs="0">
    <xs:annotation>
        <xs:documentation>A
    <predefined set of collection definitions for the purpose of conducting a known run
    <type</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element ref="RunResources" minOccurs="0"/>
    <xs:element name="RecordedEvents" minOccurs="0
    <xs:annotation>
        <xs:documentation>Journal of
    <metrics, system events, or alarms that were generated during this run's lifetime</
    <xs:documentation>
        </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name=
    <"RecordedEvent" type="pbbase:RecordedEventType" minOccurs="0" maxOccurs="unbounded">
    <xs:annotation>
    <xs:documentation>Journal of metrics, system events, or alarms that were generated
    <during this run's lifetime.
In the case of Primary generating the DataSet containing the sts.xml, this
    <RecordedEvent object should be a pointer to the DataSet object generated.</
    <xs:documentation>
        </
    <xs:annotation>
        </xs:element>
    </xs:sequence>
    </xs:complexType>
    </xs:element>
</xs:sequence>
<xs:attribute name="Status" type=
    <"pbbase:SupportedRunStates"/>
    <xs:attribute name="InstrumentId" type="xs:string">
        <xs:annotation>
            <xs:documentation>World unique id
    <assigned by PacBio. </xs:documentation>
        </xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        </xs:attribute>
        <xs:attribute name="InstrumentName" type="xs:string">
            <xs:annotation>
                <xs:documentation>Friendly name_
→ assigned by customer</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="CreatedBy" type="xs:string">
            <xs:annotation>
                <xs:documentation>Who created the run.
→ </xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="StartedBy" type="xs:string">
            <xs:annotation>
                <xs:documentation>Who started the run.
→ Could be different from who created it. </xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="WhenStarted">
            <xs:annotation>
                <xs:documentation>Date and time of_
→ when the overall run was started. </xs:documentation>
            </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:dateTime"/>
            </xs:simpleType>
        </xs:attribute>
    </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:element name="CollectionReferences">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="CollectionMetadataRef" type=
→ "xs:IDREF" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SubreadSets">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="pbds:SubreadSet" maxOccurs="unbounded
→ "/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>

```

9.2 PacBioAutomationConstraints

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 4 sp1 (x64) (http://www.altova.com) by_
→ efayad@pacificbiosciences.com (Pacific Biosciences) -->

```

(continues on next page)

(continued from previous page)

```

<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioAutomationConstraints.xsd"
  xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xi="http://www.w3.org/2003/
  XInclude" xmlns:xm="http://www.w3.org/XML/1998/namespace" xmlns:pbbase="http://
  pacificbiosciences.com/PacBioBaseDataModel.xsd" xmlns:pbrk="http://
  pacificbiosciences.com/PacBioReagentKit.xsd" xmlns:pbpn="http://pacificbiosciences.
  com/PacBioPartNumbers.xsd" targetNamespace="http://pacificbiosciences.com/
  PacBioAutomationConstraints.xsd" elementFormDefault="qualified" id=
  "PacBioAutomationConstraints">
  <xs:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation=
  "http://www.w3.org/2001/03/xml.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
  schemaLocation="PacBioReagentKit.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioPartNumbers.xsd"
  schemaLocation="PacBioPartNumbers.xsd"/>
  <xs:element name="PacBioAutomationConstraints" type=
  "PacBioAutomationConstraintsType">
    <xs:annotation>
      <xs:documentation>The root element of the Automation
    Constraints </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="AutomationConstraint" type="pbbase:AutomationConstraintType"
  />
  <xs:complexType name="PacBioAutomationConstraintsType">
    <xs:sequence>
      <xs:element name="AutomationConstraints">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="AutomationConstraint
  " maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element ref="pbpn:PacBioPartNumbers" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Version" type="xs:string"/>
    <xs:attribute name="Description" type="xs:string"/>
  </xs:complexType>
</xs:schema>

```

9.3 PacBioBaseDataModel

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2016 (x64) (http://www.altova.com) by
  efayad@pacificbiosciences.com (Pacific Biosciences) -->
<!--W3C XML Schema generated by XMLSpy v2014 sp1 (x64) (http://www.altova.com)-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://
  pacificbiosciences.com/PacBioBaseDataModel.xsd" targetNamespace="http://
  pacificbiosciences.com/PacBioBaseDataModel.xsd" elementFormDefault="qualified"
  attributeFormDefault="unqualified">

```

(continues on next page)

(continued from previous page)

```

<xs:complexType name="AnalogType">
  <xs:complexContent>
    <xs:extension base="BaseEntityType">
      <xs:sequence>
        <xs:element name="Spectrum">
          <xs:annotation>
            <xs:documentation>A vector of
↪probabilities, given in the order of increasing filter-bin wavelength, that light
↪emitted by the analog will fall in the corresponding filter bin of the instrument
↪detection system. By convention, the values are normalized to sum to 1.</
↪xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name=
↪"Values">
↪<xs:complexType>
↪<xs:sequence>
↪<xs:element name="Value" type="xs:float" maxOccurs="unbounded">
↪<xs:annotation>
↪<xs:documentation>There should be as many values as specified in the Number of
↪Filter Bins attribute.
Each value is a probability, in the range of [0, 1].</xs:documentation>
↪</xs:annotation>
↪</xs:element>
↪
↪</xs:sequence>
↪
↪</xs:complexType>
↪
↪</xs:element>
↪</xs:sequence>
↪<xs:attribute name=
↪"NumberFilterBins" type="xs:int" use="required">
↪<xs:annotation>
↪<xs:documentation>number of bins describing the spectrum, green to red</
↪xs:documentation>
↪</xs:annotation>
↪</xs:attribute>
↪</xs:complexType>
↪</xs:element>
↪<xs:element name="RelativeAmplitude" type=
↪"xs:float">
↪<xs:annotation>
↪<xs:documentation>Relative
↪intensity of emission vs. a reference analog using standardized metrology - e.g.,
↪relative to the amplitude of the "542" analog as measured by the mean DWS pkMid on
↪the Astro instrument.</xs:documentation>
↪</xs:annotation>
↪</xs:element>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="IntraPulseXsnCV" type=
↪ "xs:float">
    <xs:annotation>
        <xs:documentation>The 1-sigma_
↪ fractional variation of the intra-pulse signal, independent of any Shot noise_
↪ associated with that signal</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="InterPulseXsnCV" type=
↪ "xs:float">
    <xs:annotation>
        <xs:documentation>The 1-sigma_
↪ fractional variation, pulse-to-pulse, of the mean signal level (i.e., the pkMid).</
↪ xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="DiffusionXsnCV" type=
↪ "xs:float"/>
</xs:sequence>
<xs:attribute name="Base" type="SupportedNucleotides">
    <xs:annotation>
        <xs:documentation>The base label, A,
↪ C, T, or G</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="Nucleotide" type="xs:string">
    <xs:annotation>
        <xs:documentation>The type and number_
↪ of nucleotides on a given analog. e.g. (dT6P)6</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="Wavelength" type="xs:float">
    <xs:annotation>
        <xs:documentation>The peak emission_
↪ wavelength associated with the dye label in nm.</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="CompoundID" type="xs:string">
    <xs:annotation>
        <xs:documentation>Identification code_
↪ of the final compound. The suffix 'N' should be used to distinguish these values_
↪ from enzyme identifiers. e.g. 5031N</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="LotID" type="xs:string">
    <xs:annotation>
        <xs:documentation>Identification code_
↪ for the build of the final compound, written as initials/date, where date is_
↪ written as YYYY-MM-DD. e.g. js/2014-06-30</xs:documentation>
    </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="BaseEntityType">
    <xs:annotation>
        <xs:documentation>This is the base element type for all types_
↪ in this data model</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

</xs:annotation>
<xs:sequence>
  <xs:element name="Extensions" minOccurs="0">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ExtensionElement"
↳minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:sequence>
<xs:attribute name="Name" type="xs:string">
  <xs:annotation>
    <xs:documentation>A short text identifier; uniqueness
↳not necessary</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Description" type="xs:string">
  <xs:annotation>
    <xs:documentation>A long text description of the
↳object</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Tags" type="xs:string">
  <xs:annotation>
    <xs:documentation>A set of keywords assigned to the
↳object to help describe it and allow it to be found via search</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Format" type="xs:string">
  <xs:annotation>
    <xs:documentation>Optional, but recommended. The
↳MIME-Type of the referenced file. See http://www.iana.org/assignments/media-types/
↳media-types.xhtml for examples</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="ResourceId" type="xs:anyURI">
  <xs:annotation>
    <xs:documentation>A uniform resource identifier used
↳to identify a "web" resource. e.g. svc://run/acquisition/alignment/gridding</
↳xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Version" type="xs:string">
  <xs:annotation>
    <xs:documentation>An optional identifier denoting the
↳revision of this particular entity</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="CreatedAt">
  <xs:annotation>
    <xs:documentation>Timestamp designating the creation
↳of this object, relative to UTC; millisecond precision is expected.</
↳xs:documentation>
  </xs:annotation>
<xs:simpleType>
  <xs:restriction base="xs:dateTime"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="ModifiedAt">
        <xs:annotation>
          <xs:documentation>Timestamp designating the
↪modification of this object, relative to UTC; millisecond precision is expected.</
↪xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  <xs:complexType name="StrictEntityType">
    <xs:annotation>
      <xs:documentation>This is the base element type for all types
↪in this data model</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="BaseEntityType">
        <xs:attribute name="UniqueId" use="required">
          <xs:annotation>
            <xs:documentation>A unique identifier,
↪such as a GUID - likely autogenerated</xs:documentation>
          </xs:annotation>
          <xs:simpleType>
            <xs:restriction base="xs:ID">
              <xs:pattern value="[a-fA-F0-9]
↪{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="MetaType" type="xs:string" use=
↪"required">
          <xs:annotation>
            <xs:documentation>Controlled
↪Vocabulary, meant as a means to group similar entities; the type of the object, e.g.
↪Instrument Run, Secondary Run, Assay, Sample, Barcode, Alignment File, Alarm,
↪Exception, Metric, SystemEvent, etc.</xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="TimeStampedName" type="xs:string"
↪use="required">
          <xs:annotation>
            <xs:documentation>This is NOT
↪intended to be used as a unique field. For uniqueness, use UniqueId. In order to
↪not utilize customer provided names, this attribute may be used as an alternative
↪means of Human Readable ID, e.g. instrumentId-Run-150304_231155</xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="ConfigSetAnalog">
    <xs:annotation>
      <xs:documentation>An unlimited number of analogs listed for
↪the purposes of hosting in a configuration file. e.g. a list of all possible
↪analogs on the system</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="AnalogType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="DataEntity" type="DataEntityType"/>
    <xs:complexType name="DataEntityType">
        <xs:annotation>
            <xs:documentation>Extends BaseEntityType and adds a value_
↪element. The intent is to have only one of the value elements exist at any point_
↪in time; however, this is not enforced.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="BaseEntityType">
                <xs:sequence>
                    <!--xs:element name="EncodedValue" type=
↪"xs:base64Binary" nillable="true" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>A complex_
↪data type element, such as an image, file, binary object, etc.</xs:documentation>
                        </xs:annotation>
                    </xs:element-->
                    <xs:element name="Checksum" type="xs:string"
↪minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>small-size_
↪datum of the attached value for the purpose of detecting errors or modification_
↪which may have been introduced during its transmission or storage</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="ValueDataType" type=
↪"SupportedDataTypes" default="Object">
                    <xs:annotation>
                        <xs:documentation>The datatype of the_
↪simple or encoded value. If not specified, a string is assumed.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="SimpleValue" type=
↪"xs:anySimpleType">
                    <xs:annotation>
                        <xs:documentation>A simple data type_
↪element, such as a string, int, float, etc.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="MetaType" type="xs:string">
                    <xs:annotation>
                        <xs:documentation>Controlled_
↪Vocabulary, meant as a means to group similar entities; the type of the object, e.g.
↪Instrument Run, Secondary Run, Assay, Sample, Barcode, Alignment File, Alarm,_
↪Exception, Metric, SystemEvent, etc.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="TimeStampedName" type="xs:string">
                    <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

                                <xs:documentation>This is NOT
↳intended to be used as a unique field.  For uniqueness, use UniqueId.  In order to
↳not utilize customer provided names, this attribute may be used as an alternative
↳means of Human Readable ID, e.g. instrumentId-Run-150304_231155</xs:documentation>
                                </xs:annotation>
                                </xs:attribute>
                                </xs:extension>
                                </xs:complexContent>
</xs:complexType>
<xs:element name="DataPointers">
    <xs:annotation>
        <xs:documentation>Pointer list to UniqueIds in the system</
↳xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="DataPointer" type="xs:IDREF"
↳minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="DNABarcode">
        <xs:annotation>
            <xs:documentation>Composite of uuid, sequence, and name</
↳xs:documentation>
            </xs:annotation>
            <xs:complexContent>
                <xs:extension base="BaseEntityType">
                    <xs:attribute name="DNASequence">
                        <xs:annotation>
                            <xs:documentation>This is the sample
↳'s DNA barcode</xs:documentation>
                        </xs:annotation>
                    </xs:attribute>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
    <xs:element name="DyeSetAnalog">
        <xs:annotation>
            <xs:documentation>A set of four analogs, one for each of the
↳nucleotides, grouped together for the purposes of a single experiment.</
↳xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:complexContent>
                    <xs:extension base="AnalogType"/>
                </xs:complexContent>
            </xs:complexType>
        </xs:element>
    <xs:element name="ExternalResources">
        <xs:annotation>
            <xs:documentation>Pointers to data that do not reside inside
↳the parent structure</xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="ExternalResource" maxOccurs=
↳"unbounded"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:complexType name="InputOutputDataType">
      <xs:complexContent>
        <xs:extension base="StrictEntityType"/>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="IndexedDataType">
      <xs:complexContent>
        <xs:extension base="InputOutputDataType">
          <xs:sequence>
            <xs:element name="FileIndices" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name=
↪ "FileIndex" type="InputOutputDataType" maxOccurs="unbounded">

↪ <xs:annotation>

↪ <xs:documentation>e.g. index for output files, allowing one to find information in
↪ the output file</xs:documentation>

                  </
↪ <xs:annotation>

                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element ref="ExternalResources" minOccurs=
↪ "0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AutomationType">
      <xs:complexContent>
        <xs:extension base="BaseEntityType">
          <xs:sequence>
            <xs:element name="AutomationParameters"
↪ minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element ref=
↪ "AutomationParameter" maxOccurs="unbounded"/>

                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element ref="Defaults" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="PartNumber" type="xs:string" use=
↪ "optional">
            <xs:annotation>
              <xs:documentation>Defines a part
↪ number, mainly for use in defining incompatibility with other PB kit PNs, if
↪ necessary</xs:documentation>
            </xs:annotation>
          </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

        <xs:attribute name="IsRestricted" type="xs:boolean"
↪use="optional" default="false">
            <xs:annotation>
                <xs:documentation>Allows for an
↪automation to be marked for internal use or by admin users only</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="IsObsolete" type="xs:boolean" use=
↪"optional" default="false">
            <xs:annotation>
                <xs:documentation>Allows for an
↪automation to be marked as obsolete</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="IsDefault" type="xs:boolean" use=
↪"optional" default="false">
            <xs:annotation>
                <xs:documentation>Allows for an
↪automation to be marked as a default for a kit with which it's compatible</
↪xs:documentation>
            </xs:annotation>
        </xs:attribute>
    </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:element name="AutomationParameter" type="DataEntityType">
    <xs:annotation>
        <xs:documentation>One or more collection parameters, such as
↪MovieLength, InsertSize, UseStageStart, IsControl, etc..</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:complexType name="IncompatiblePairType">
    <xs:annotation>
        <xs:documentation>Describes a bidirectional incompatibility
↪between part numbers.

By default, any PN is compatible for use with other PNs in the system. In order to
↪exclude the usage of one or more PNs with this one, the pairwise incompatible PNs
↪are listed here.</xs:documentation>
    </xs:annotation>
</xs:complexContent>
    <xs:extension base="BaseEntityType">
        <xs:attribute name="PartA" type="xs:string" use=
↪"required">
            <xs:annotation>
                <xs:documentation>An automation or
↪kit Part Number that's incompatible with Part Number B</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="PartB" type="xs:string" use=
↪"required">
            <xs:annotation>
                <xs:documentation>An automation or
↪kit Part Number that's incompatible with Part Number A</xs:documentation>
            </xs:annotation>
        </xs:attribute>
    </xs:extension>

```

(continues on next page)

(continued from previous page)

```

        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AutomationConstraintType">
        <xs:annotation>
            <xs:documentation>This data type defines constraints that an
↪automation has. The information here, along with the availability of an
↪exclusionary list of automations in the PartNumberType, allows for defining a
↪robust compatibility matrix.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="BaseEntityType">
                <xs:sequence>
                    <xs:element name="Automations" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>Names of
↪automations that are all similarly constrained</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name=
↪"Automation" type="AutomationType" maxOccurs="unbounded"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="SupportsCellReuse" type=
↪"xs:boolean">
                    <xs:annotation>
                        <xs:documentation>Does this
↪automation support cell reuse?</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="SupportsStageStart" type=
↪"xs:boolean">
                    <xs:annotation>
                        <xs:documentation>Does this
↪automation support hot-start on the stage?</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="MaxCollectionsPerCell" type=
↪"xs:int">
                    <xs:annotation>
                        <xs:documentation>If cell reuse is
↪supported (i.e. above attribute is true) how many times can the cell be reused?</
↪xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="MinMovieLength" type="xs:int">
                    <xs:annotation>
                        <xs:documentation>Minimum length of
↪movie acquisition</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="MaxMovieLength" type="xs:int">
                    <xs:annotation>
                        <xs:documentation>Maximum length of
↪movie acquisition</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="MinInsertSize" type="xs:int">
        <xs:annotation>
          <xs:documentation>Minimum recommended_
↪insert size</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="MaxInsertSize" type="xs:int">
        <xs:annotation>
          <xs:documentation>Maximum recommended_
↪insert size</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="PartNumberType">
  <xs:annotation>
    <xs:documentation>Generic representation of a supply kit.
    If the part number has an NFC associated with it, the contents of the NFC may be_
↪encoded here.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataEntityType">
      <xs:sequence>
        <xs:element ref="Defaults" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="PartNumber" type="xs:string">
        <xs:annotation>
          <xs:documentation>The kit part number
↪</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="LotNumber" type="xs:string">
        <xs:annotation>
          <xs:documentation>The kit lot number</
↪xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="Barcode" type="xs:string">
        <xs:annotation>
          <xs:documentation>The kit barcode;_
↪used for tracking purposes.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="ExpirationDate" type="xs:date">
        <xs:annotation>
          <xs:documentation>The kit's shelf life
↪</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="IsObsolete" type="xs:boolean"_
↪default="false"/>
      <xs:attribute name="IsRestricted" type="xs:boolean"_
↪default="false"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="RecordedEventType">
      <xs:annotation>
        <xs:documentation>Metrics, system events, alarms, and logs_
↳may utilize this type</xs:documentation>
      </xs:annotation>
      <xs:complexContent>
        <xs:extension base="DataEntityType">
          <xs:attribute name="Context" type="xs:string">
            <xs:annotation>
              <xs:documentation>The part of the_
↳system in effect when the event was recorded</xs:documentation>
            </xs:annotation>
          </xs:attribute>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="SequencingChemistry">
      <xs:annotation>
        <xs:documentation>A container for a set of analogs</
↳xs:documentation>
      </xs:annotation>
      <xs:complexContent>
        <xs:extension base="DataEntityType">
          <xs:sequence>
            <xs:element name="DyeSet">
              <xs:complexType>
                <xs:complexContent>
                  <xs:extension base=
↳"BaseEntityType">
                    <xs:sequence>
                      <xs:element name="Analog">
                        <xs:complexType>
                          <xs:sequence>
                            <xs:element name="Analog" type="AnalogType" maxOccurs="4"/>
                          </xs:sequence>
                        </xs:complexType>
                      </xs:element>
                    </xs:sequence>
                  </xs:extension>
                </xs:complexContent>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="PacBioSequencingChemistry">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>Root element for document containing the
↪ container of analog set, SequencingChemistryConfig</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="ChemistryConfig"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:complexType name="SequencingChemistryConfig">
        <xs:annotation>
          <xs:documentation>A container for a set of analogs</
↪ xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="DataEntityType">
            <xs:sequence>
              <xs:element name="Analog">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪ "Analog" type="AnalogType" maxOccurs="4"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="DefaultLaserSetPoint" type=
↪ "xs:float">
                <xs:annotation>
                  <xs:documentation>The laser
↪ power at the input couple, needed to achieve predefined performance requirements
↪ based on a median 'golden' SMRT Cell.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="SNRCut" type="xs:float">
                <xs:annotation>
                  <xs:documentation>The SNRCut
↪ is applied in PPA (baz2bam) as a read-quality filter.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="TargetSNR">
                <xs:complexType>
                  <xs:attribute name="SNR_A"
↪ type="xs:float" use="required"/>
                  <xs:attribute name="SNR_C"
↪ type="xs:float" use="required"/>
                  <xs:attribute name="SNR_G"
↪ type="xs:float" use="required"/>
                  <xs:attribute name="SNR_T"
↪ type="xs:float" use="required"/>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:complexType name="StatsContinuousDistType">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>Continuous distribution class</
↪xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="BaseEntityType">
            <xs:sequence>
              <xs:element name="SampleSize" type="xs:int"/>
              <xs:element name="SampleMean" type="xs:float"/
↪>
              <xs:element name="SampleMed" type="xs:float"/>
              <xs:element name="SampleStd" type="xs:float"/>
              <xs:element name="Sample95thPct" type=
↪"xs:float"/>
              <xs:element name="NumBins" type="xs:int"/>
              <xs:element name="BinCounts">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪"BinCount" type="xs:int" maxOccurs="unbounded"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="BinWidth" type="xs:float"/>
              <xs:element name="MinOutlierValue" type=
↪"xs:float" minOccurs="0"/>
              <xs:element name="MinBinValue" type="xs:float"
↪" minOccurs="0"/>
              <xs:element name="MaxBinValue" type="xs:float"
↪" minOccurs="0"/>
              <xs:element name="MaxOutlierValue" type=
↪"xs:float" minOccurs="0"/>
              <xs:element name="MetricDescription" type=
↪"xs:string"/>
            </xs:sequence>
            <xs:attribute name="Channel" type="xs:string"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="StatsDiscreteDistType">
        <xs:annotation>
          <xs:documentation>Discrete distribution class</
↪xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="BaseEntityType">
            <xs:sequence>
              <xs:element name="NumBins" type="xs:int"/>
              <xs:element name="BinCounts">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪"BinCount" type="xs:int" maxOccurs="unbounded"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="MetricDescription" type=
↪"xs:string"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:element name="BinLabels">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name=
↪ "BinLabel" type="xs:string" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="StatsTimeSeriesType">
    <xs:annotation>
        <xs:documentation>Time series (for time-dependent metrics)</
↪ xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="BaseEntityType">
            <xs:sequence>
                <xs:element name="TimeUnits" type="xs:string"/
↪ >
                <xs:element name="ValueUnits" type="xs:string"
↪ "/>
                <xs:element name="StartTime" type="xs:float"/>
                <xs:element name="MeasInterval" type="xs:float"
↪ "/>
                <xs:element name="Values" minOccurs="0">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Val
↪ " type="xs:float" minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="SupplyKitBinding">
    <xs:annotation>
        <xs:documentation>A more specific binding kit representation
↪ (includes SupplyKit fields). </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="PartNumberType">
            <xs:sequence>
                <xs:element name="Control" type=
↪ "SupplyKitControl" minOccurs="0">
                    <xs:annotation>
                        <xs:documentation>Defines the
↪ binding kit internal control name. Present when used, otherwise not used if not
↪ defined. </xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="IsControlUsed" type=
↪ "xs:boolean" minOccurs="0">

```

(continues on next page)

(continued from previous page)

```

                                <xs:annotation>
                                    <xs:documentation>True if the
↪control was used during run, otherwise false. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SupplyKitCellPack">
                <xs:annotation>
                    <xs:documentation>Represents the package of cells. </
↪xs:documentation>
                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="PartNumberType">
                        <xs:sequence>
                            <xs:element name="ChipLayout" type="xs:string
↪" minOccurs="0">
                                <xs:annotation>
                                    <xs:documentation>Defines the
↪internal chip layout name, if any. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="SupportsCellReuse">
                            <xs:annotation>
                                <xs:documentation>If
↪SupportsCellReuse is true, it can be used for regular sequencing as well as in a
↪reuse scenario.</xs:documentation>
                            </xs:annotation>
                        </xs:attribute>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SupplyKitControl">
                <xs:annotation>
                    <xs:documentation>Represents the DNA control complex. </
↪xs:documentation>
                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="PartNumberType">
                        <xs:sequence>
                            <xs:element name="InternalControlName" type=
↪"xs:string" minOccurs="0">
                                <xs:annotation>
                                    <xs:documentation>Defines the
↪internal control name, if any. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="CustomSequence" type=
↪"xs:string" nillable="true" minOccurs="0"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SupplyKitTemplate">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>A more specific template kit representation
↪ (includes SupplyKit fields). </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="PartNumberType">
            <xs:sequence>
              <xs:element name="LeftAdaptorSequence" type=
↪ "xs:string" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>Left
↪ adapter DNA sequence.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="LeftPrimerSequence" type=
↪ "xs:string" minOccurs="0"/>
              <xs:element name="RightAdaptorSequence" type=
↪ "xs:string" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>Right
↪ adapter DNA sequence. If not specified, a symmetric adapter model is inferred,
↪ where the left adapter sequence is used wherever needed.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="RightPrimerSequence" type=
↪ "xs:string" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>Right
↪ primer sequence. If not specified, a symmetric model is inferred, where the left
↪ primer sequence is used wherever needed.</xs:documentation>
                </xs:annotation>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="MinInsertSize" type="xs:int">
              <xs:annotation>
                <xs:documentation>Minimum recommended
↪ insert size</xs:documentation>
              </xs:annotation>
            </xs:attribute>
            <xs:attribute name="MaxInsertSize" type="xs:int">
              <xs:annotation>
                <xs:documentation>Maximum recommended
↪ insert size</xs:documentation>
              </xs:annotation>
            </xs:attribute>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:simpleType name="SupportedAcquisitionStates">
        <xs:restriction base="xs:string">
          <xs:enumeration value="Ready"/>
          <xs:enumeration value="Initializing"/>
          <xs:enumeration value="SocketDiagnostics"/>
          <xs:enumeration value="Acquiring"/>
          <xs:enumeration value="Aligning"/>
          <xs:enumeration value="Aligned"/>
          <xs:enumeration value="Aborting"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:sequence>
  </xs:element>
</xs:sequence>
</xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        <xs:enumeration value="Aborted"/>
        <xs:enumeration value="Failed"/>
        <xs:enumeration value="Completing"/>
        <xs:enumeration value="Complete"/>
        <xs:enumeration value="Calibrating"/>
        <xs:enumeration value="Unknown"/>
        <xs:enumeration value="Pending"/>
        <xs:enumeration value="ReadyToCalibrate"/>
        <xs:enumeration value="CalibrationComplete"/>
        <xs:enumeration value="ReadyToAcquire"/>
        <xs:enumeration value="FinishingAnalysis"/>
        <xs:enumeration value="PostPrimaryPending"/>
        <xs:enumeration value="PostPrimaryAnalysis"/>
        <xs:enumeration value="TransferPending"/>
        <xs:enumeration value="TransferringResults"/>
        <xs:enumeration value="Error"/>
        <xs:enumeration value="Stopped"/>
        <xs:enumeration value="TransferFailed"/>
        <xs:enumeration value="InPrep"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="SupportedDataTypes">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Int16"/>
        <xs:enumeration value="Int32"/>
        <xs:enumeration value="Int64"/>
        <xs:enumeration value="UInt16"/>
        <xs:enumeration value="UInt32"/>
        <xs:enumeration value="UInt64"/>
        <xs:enumeration value="Boolean"/>
        <xs:enumeration value="Single"/>
        <xs:enumeration value="Double"/>
        <xs:enumeration value="String"/>
        <xs:enumeration value="DateTime"/>
        <xs:enumeration value="Int16_1D"/>
        <xs:enumeration value="Int32_1D"/>
        <xs:enumeration value="Int64_1D"/>
        <xs:enumeration value="UInt16_1D"/>
        <xs:enumeration value="UInt32_1D"/>
        <xs:enumeration value="UInt64_1D"/>
        <xs:enumeration value="Boolean_1D"/>
        <xs:enumeration value="Single_1D"/>
        <xs:enumeration value="Double_1D"/>
        <xs:enumeration value="String_1D"/>
        <xs:enumeration value="DateTime_1D"/>
        <xs:enumeration value="Int16_2D"/>
        <xs:enumeration value="Int32_2D"/>
        <xs:enumeration value="Int64_2D"/>
        <xs:enumeration value="UInt16_2D"/>
        <xs:enumeration value="UInt32_2D"/>
        <xs:enumeration value="UInt64_2D"/>
        <xs:enumeration value="Boolean_2D"/>
        <xs:enumeration value="Single_2D"/>
        <xs:enumeration value="Double_2D"/>
        <xs:enumeration value="String_2D"/>
        <xs:enumeration value="DateTime_2D"/>
        <xs:enumeration value="XML"/>
    </xs:restriction>

```

(continues on next page)

(continued from previous page)

```

        <xs:enumeration value="JSON"/>
        <xs:enumeration value="Object"/>
        <xs:enumeration value="Other"/>
        <xs:enumeration value="Unknown"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="SupportedNucleotides">
    <xs:restriction base="xs:string">
        <xs:enumeration value="A"/>
        <xs:enumeration value="C"/>
        <xs:enumeration value="T"/>
        <xs:enumeration value="G"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="SupportedRunStates">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Ready"/>
        <xs:enumeration value="Idle"/>
        <xs:enumeration value="System Test"/>
        <xs:enumeration value="Starting"/>
        <xs:enumeration value="Running"/>
        <xs:enumeration value="Aborting"/>
        <xs:enumeration value="Aborted"/>
        <xs:enumeration value="Terminated"/>
        <xs:enumeration value="Completing"/>
        <xs:enumeration value="Complete"/>
        <xs:enumeration value="Paused"/>
        <xs:enumeration value="Unknown"/>
    </xs:restriction>
</xs:simpleType>
<xs:complexType name="UserDefinedFieldsType">
    <xs:sequence>
        <xs:element name="DataEntities" type="DataEntityType"
↳maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="ValueDataType" type="SupportedDataTypes"/>
<!-- The base supply kit definition -->
<xs:element name="ExtensionElement" type="xs:anyType">
    <xs:annotation>
        <xs:documentation>A generic element whose contents are
↳undefined at the schema level. This is used to extend the data model.</
↳xs:documentation>
    </xs:annotation>
</xs:element>
<xs:complexType name="FilterType">
    <xs:sequence>
        <xs:element name="Properties">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="Property" maxOccurs=
↳"unbounded">
                        <xs:complexType>
                            <xs:attribute name=
↳"Name" type="xs:string" use="required"/>
                            <xs:attribute name=
↳"Value" type="xs:string" use="required"/>

```

(continues on next page)

(continued from previous page)

```

<xs:attribute name=
↪ "Operator" type="xs:string" use="required"/>
<xs:attribute name=
↪ "Feature" type="xs:string" use="optional"/>
<xs:attribute name=
↪ "Assignment" type="xs:string" use="optional"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:element name="ExternalResource">
  <xs:annotation>
    <xs:documentation>for example, an output file could be the_
↪ BAM file, which could be associated with multiple indices into it.</
↪ xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="IndexedDataType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="KeyValueMap">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Items">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Item" type=
↪ "MapItemType" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
        <xs:unique name="Item">
          <xs:selector xpath="Item"/>
          <xs:field xpath="Key"/>
        </xs:unique>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:complexType name="MapType">
  <xs:sequence>
    <xs:element ref="KeyValueMap"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="MapItemType">
  <xs:sequence>
    <xs:element name="Key" type="xs:ID"/>
    <xs:element name="Value" type="xs:anyType"/>
    <xs:element name="Description" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="ChemistryConfig" type="SequencingChemistryConfig"/>
<xs:element name="Defaults" type="DefaultsType" nillable="true">

```

(continues on next page)

(continued from previous page)

```

    <xs:annotation>
      <xs:documentation>Default paramaters and filters which may be applied to
↪PartNumber types in order to constrain them via parameterization</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:complexType name="DefaultsType">
    <xs:annotation>
      <xs:documentation>A data type that allows the definition of default
↪paramaters and filters. This structure may be applied to PartNumber types in order
↪to constrain them via parameterization</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="AutomationParameters" nillable="true" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="AutomationParameter" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Filters" nillable="true" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Filter" type="FilterType" maxOccurs=
↪"unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

9.4 PacBioCollectionMetadata

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2016 (x64) (http://www.altova.com) by
↪efayad@pacificbiosciences.com (Pacific Biosciences) -->
<?xml-stylesheet type="application/xml" href="metadata2html.xslt"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
↪xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.
↪com/PacBioBaseDataModel.xsd" xmlns:pbsample="http://pacificbiosciences.com/
↪PacBioSampleInfo.xsd" xmlns:pbrk="http://pacificbiosciences.com/PacBioReagentKit.xsd
↪" targetNamespace="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
↪elementFormDefault="qualified" id="PacBioCollectionMetadata">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↪schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
↪schemaLocation="PacBioSampleInfo.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
↪schemaLocation="PacBioReagentKit.xsd"/>
  <!-- The root element for the metadata structure -->
  <xs:element name="PacBioCollectionMetadata">
    <xs:annotation>
      <xs:documentation>Root element of a standalone
↪CollectionMetadata file.</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="CollectionMetadata"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Collections">
        <xs:annotation>
            <xs:documentation>A set of acquisition definitions</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="CollectionMetadata" maxOccurs=
↪"unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="CollectionMetadata">
        <xs:annotation>
            <xs:documentation>Root-level element for the metadata. The
↪purpose of which is to contain pertinent instrument information related to the
↪conditions present during a movie acquisition. It also serves to provide key
↪pieces of information for integration with primary and secondary analysis. This
↪file is associated with 1 movie. </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="pbbase:StrictEntityType">
                    <xs:sequence>
                        <xs:element name="InstCtrlVer" type=
↪"xs:string" minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
↪Instrument control software version. </xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element name="SigProcVer" type=
↪"xs:string" minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
↪Signal processing software version. </xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element ref="RunDetails"
↪minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
↪Container for instrument run-related information. </xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element ref="Movie" minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
↪Container for movie related information. This is populated by primary analysis.</
↪xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element ref="WellSample">
        <xs:annotation>
          <xs:documentation>
↪ Container for sample related information, specific to the plate well. This could
↪ be a pooled sample with many barcodes, associated with many biological samples.

If multiple collections are utilizing the same physical plate well, all these
↪ collections should share the same WellSample UniqueId.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Automation" type=
↪ "pbbase:AutomationType">
          <xs:annotation>
            <xs:documentation>
↪ Defines the collection workflow (e.g., robotic movement, movie acquisition) for a
↪ particular cell. </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="CollectionNumber"
↪ type="xs:int" minOccurs="0">
          <xs:annotation>
            <xs:documentation>
↪ Collection number for this plate well. Sample from one plate well or tube can be
↪ distributed to more than one cell. </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="CellIndex" type=
↪ "xs:int" minOccurs="0">
          <xs:annotation>
            <xs:documentation>The
↪ zero-based index of this particular cell within the cell tray. Likely to be in the
↪ range of [0-3]</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="SetNumber" type=
↪ "xs:unsignedShort" minOccurs="0">
          <xs:annotation>
            <xs:documentation>
↪ Formerly known as the look number. 1 - N. Defaults to 1. 0 if the look is unknown.
↪ </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="CellPac" type=
↪ "pbbase:SupplyKitCellPack" minOccurs="0">
          <xs:annotation>
            <xs:documentation>The
↪ SMRT cell packaging supply information. </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="ControlKit" type=
↪ "pbbase:SupplyKitControl" minOccurs="0">
          <xs:annotation>
            <xs:documentation>
↪ Defines the DNA control used for this experiment. </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:element>
    </xs:element>
  </xs:element>

```

(continues on next page)

(continued from previous page)

```

</xs:element>
<xs:element name="TemplatePrepKit"
↳type="pbbase:SupplyKitTemplate" minOccurs="0">
    <xs:annotation>
        <xs:documentation>
↳Defines the template (sample) prep kit used for this experiment. Can be used to get
↳back to the primary and adapter used. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="BindingKit" type=
↳"pbbase:SupplyKitBinding" minOccurs="0">
    <xs:annotation>
        <xs:documentation>The
↳binding kit supply information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="SequencingKitPlate"
↳type="pbrk:SupplyKitSequencing" minOccurs="0">
    <xs:annotation>
        <xs:documentation>The
↳sequencing kit supply information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="Primary" minOccurs="0
↳">
    <xs:annotation>
        <xs:documentation>The
↳primary analysis-specific information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="Secondary" minOccurs=
↳"0">
    <xs:annotation>
        <xs:documentation>The
↳secondary analysis-specific information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="UserDefinedFields"
↳minOccurs="0" maxOccurs="1">
    <xs:annotation>
        <xs:documentation>A
↳set of custom key-value pairs, set by a user when building a run definition. </
↳xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="ExpirationData"
↳minOccurs="0">
    <xs:annotation>
        <xs:documentation>The
↳information on whether consumables were expired. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="ComponentVersions"
↳nillable="true" minOccurs="0">
    <xs:annotation>
        <xs:documentation>
↳Subcomponents involved in the generation of the data</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

</xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element
↳name="VersionInfo" type="pbbase:BaseEntityType" maxOccurs="unbounded">

↳<xs:annotation>

↳<xs:documentation>Each component should list its name and version attribute</
↳xs:documentation>

</
↳xs:annotation>

</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="Context" type="xs:string">
  <xs:annotation>
    <xs:documentation>Replace
↳with TimeStampedName</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Status" type=
↳"pbbase:SupportedAcquisitionStates"/>
  <xs:attribute name="InstrumentId" type=
↳"xs:string">
    <xs:annotation>
      <xs:documentation>World
↳unique id assigned by PacBio. </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="InstrumentName" type=
↳"xs:string">
    <xs:annotation>
      <xs:documentation>Friendly
↳name assigned by customer</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="RunDetails">
  <xs:annotation>
    <xs:documentation>Information related to an instrument run.
↳A run can contain multiple chips, wells, and movies. </xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TimeStampedName" type="xs:string">
        <xs:annotation>
          <xs:documentation>A unique identifier
↳for this run. Format is r[sid]_[iname]_[ts]. Where [id] is a system generated id
↳and [iname] is the instrument name and [ts] is a timestamp YYMMDD Example: r000123_
↳00117_100713 </xs:documentation>
        </xs:annotation>

```

(continues on next page)

(continued from previous page)

```

</xs:element>
<xs:element name="Name" type="xs:string" minOccurs="0"
↳ ">
    <xs:annotation>
        <xs:documentation>Assigned name for a
↳ run, which consists of multiple wells. There is no constraint on the uniqueness of
↳ this data. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="CreatedBy" type="xs:string"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Who created the run.
↳ </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="WhenCreated" type="xs:dateTime"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Date and time of
↳ when the overall run was created in the system. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="StartedBy" type="xs:string"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Who started the run.
↳ Could be different from who created it. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="WhenStarted" type="xs:dateTime"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Date and time of
↳ when the overall run was started. </xs:documentation>
    </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Movie">
    <xs:annotation>
        <xs:documentation>A movie corresponds to one acquisition for
↳ a chip, one set (look) and one strobe. </xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="WhenStarted" type="xs:dateTime">
                <xs:annotation>
                    <xs:documentation>Date and time of
↳ when this movie acquisition started. </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="DurationInSec" type="xs:int"
↳ default="0">
                <xs:annotation>
                    <xs:documentation>The actual length
↳ of the movie acquisition (in seconds), irrespective of the movie duration specified
↳ by an automation parameter. </xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

```

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element name="Number" type="xs:int" default="0">
        <xs:annotation>
          <xs:documentation>The number of this_
↪movie within the set (i.e., look). This is unique when combined with the 'SetNumber
↪'. </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ExpirationData">
  <xs:annotation>
    <xs:documentation>Container for the expired consumable data.
↪</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TemplatePrepKitPastExpiration" type=
↪"xs:int">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the template prep kit was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="BindingKitPastExpiration" type=
↪"xs:int">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the binding kit was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CellPacPastExpiration" type="xs:int
↪">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the cell pac was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CellPacPastExpiration" type="xs:int
↪">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the eight pac was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="SequencingKitPastExpiration" type=
↪"xs:int">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the reagent kit was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="SequencingTube0PastExpiration" type=
↪"xs:int">
        <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

                                <xs:documentation>Number of days past
↪expiration the reagent tube 0 was (if at all). </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="SequencingTubelPastExpiration" type=
↪"xs:int">
                                <xs:annotation>
                                <xs:documentation>Number of days past
↪expiration the reagent tube 1 was (if at all). </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                </xs:sequence>
                                </xs:complexType>
                                </xs:element>
                                <xs:element name="WellSample">
                                <xs:annotation>
                                <xs:documentation>Container for the sample related data. </
↪xs:documentation>
                                </xs:annotation>
                                <xs:complexType>
                                <xs:complexContent>
                                <xs:extension base="pbbase:BaseEntityType">
                                <xs:sequence>
                                <xs:element name="WellName" type=
↪"xs:string">
                                <xs:annotation>
                                <xs:documentation>
↪Identifies which well this sample came from (e.g., coordinate on a plate). </
↪xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="Concentration" type=
↪"xs:double">
                                <xs:annotation>
                                <xs:documentation>
↪Sample input concentration. </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="InsertSize" type=
↪"xs:int">
                                <xs:annotation>
                                <xs:documentation>
↪Length of the sheared template, e.g. 500, 2000, 30000</xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="SampleReuseEnabled"
↪type="xs:boolean">
                                <xs:annotation>
                                <xs:documentation>
↪Whether or not complex reuse is enabled for this well. </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="StageHotstartEnabled
↪" type="xs:boolean">
                                <xs:annotation>
                                <xs:documentation>
↪Whether or not hotstart at the stage is enabled for this well. </xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
    </xs:element>
    <xs:element name="SizeSelectionEnabled"
        type="xs:boolean">
        <xs:annotation>
            <xs:documentation>
                Whether or not size selection is enabled for this well. </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="UseCount" type="
            xs:int">
            <xs:annotation>
                <xs:documentation>
                    Count of usages for this batch of complex. </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="DNAControlComplex"
                type="xs:string" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>
                        Indicating what kind (if any) control was used. </xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="SampleBarcodeInfo"
                    type="pbbase:DataEntityType" minOccurs="0">
                    <xs:annotation>
                        <xs:documentation>
                            When utilizing DNA barcoding, store the list of sample barcodes in this element.</
                        <xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element ref="
                        pbsample:BioSamplePointers" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>
                                Back references to other sample object UniqueIds
                                This is optional since we will likely not have a link back to samples created in the
                                calculator.</xs:documentation>
                            </xs:annotation>
                        </xs:element>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <!-- Primary analysis-related fields.-->
    <xs:element name="Primary">
        <xs:annotation>
            <xs:documentation>Container for the primary analysis related
            data. </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="SampleTrace" minOccurs="0"
                maxOccurs="1">
                <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

                                <xs:documentation>Tag to indicate
↳that the trace file will be sampled. </xs:documentation>
                                </xs:annotation>
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name=
↳"TraceSamplingFactor" type="xs:float">
                                            <xs:annotation>
↳<xs:documentation>Percentage of traces to sample. </xs:documentation>
                                            </xs:annotation>
                                        </xs:element>
                                        <xs:element name=
↳"FullPulseFile" type="xs:boolean">
                                            <xs:annotation>
↳<xs:documentation>Whether full or sampled pulse file is transferred if requested. </
↳xs:documentation>
                                            </xs:annotation>
                                        </xs:element>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="AutomationName" type="xs:string">
                                <xs:annotation>
                                    <xs:documentation>Name of primary
↳analysis protocol. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="ConfigFileName" type="xs:string">
                                <xs:annotation>
                                    <xs:documentation>Name of primary
↳analysis config file. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="SequencingCondition" type="xs:string
↳">
                                <xs:annotation>
                                    <xs:documentation>A sequencing
↳condition tag to be used by primary analysis, e.g., to select basecaller
↳calibration or training parameters. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="OutputOptions">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name=
↳"ResultsFolder" type="xs:string">
                                            <xs:annotation>
↳<xs:documentation>NOTE: not for customers. A sub-folder under the CollectionPath
↳created by Primary Analysis. This is a field that will be updated by the primary
↳analysis pipeline. The default (as created by homer) should be set to 'Reports_Sms
↳' for now. Consumers of the data should be aware that they will find collection
↳metadata (and trace files if acquisition is so-configured) at the CollectionPathUri,
↳and all primary analysis results in the sub-folder PrimaryResultsFolder. </
↳xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element name=
↪ "CollectionPathUri" type="xs:anyURI">
        <xs:annotation>
↪ <xs:documentation>User-specified location of where the results should be copied,
↪ after an analysis has been completed. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CopyFiles">
        <xs:complexType>
          <xs:sequence>
↪ <xs:element name="CollectionFileCopy" type="PapOutputFile" maxOccurs="unbounded">
↪ <xs:annotation>
↪ <xs:documentation>Defines the set of files to be copied to the CollectionPathUri. 1
↪ or more. </xs:documentation>
↪ </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Readout">
      <xs:annotation>
↪ <xs:documentation>BazIO Readout option; valid values are Bases (default) and Pulses
↪ </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
↪ <xs:restriction base="xs:string">
↪ <xs:enumeration value="Pulses"/>
↪ <xs:enumeration value="Bases"/>
↪ <xs:enumeration value="Bases_Without_QVs"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  <xs:element name=
↪ "MetricsVerbosity" default="Minimal">
    <xs:annotation>
↪ <xs:documentation>BazIO MetricsVerbosity option; valid values are Minimal (default),
↪ High, and None</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
↪ <xs:restriction base="xs:string">

```

(continues on next page)

(continued from previous page)

```

↪<xs:enumeration value="Minimal"/>
↪<xs:enumeration value="High"/>
↪<xs:enumeration value="None"/>
↪<xs:restriction>
                                                    </
                                                    </xs:simpleType>
</xs:element>
<xs:element name=
↪"TransferResource" minOccurs="0">
                                                    <xs:annotation>
↪<xs:documentation>Transfer Resource (optional for now, but will be made required_
↪when ICS implements this)</xs:documentation>
                                                    </xs:annotation>
                                                    <xs:complexType>
                                                    <xs:sequence>

↪<xs:element name="Id">
↪<xs:annotation>
↪<xs:documentation>Id of the Transfer Resource that is unique to the Scheme Type. A_
↪tuple of (TransferScheme, Id) will be globally unique</xs:documentation>
↪</xs:annotation>
↪<xs:simpleType>
↪<xs:restriction base="xs:string">
↪<xs:pattern value="([a-zA-Z0-9_\-])*/>
↪</xs:restriction>
↪</xs:simpleType>
                                                    </
↪<xs:element>
↪<xs:element name="TransferScheme">
↪<xs:annotation>
↪<xs:documentation>Transfer Scheme type (this should be an enum Scheme of rsync, srs_
↪or nfs)</xs:documentation>
↪</xs:annotation>
↪<xs:simpleType>
↪<xs:restriction base="xs:string">
↪<xs:enumeration value="RSYNC"/>
↪<xs:enumeration value="SRS"/>

```

(continues on next page)

(continued from previous page)

```

↪<xs:enumeration value="NFS"/>
↪</xs:restriction>
↪</xs:simpleType>
↪xs:element>
↪<xs:element name="Name" type="xs:string">
↪<xs:annotation>
↪<xs:documentation>Display Name of the Transfer Resource</xs:documentation>
↪</xs:annotation>
↪xs:element>
↪<xs:element name="Description" type="xs:string">
↪<xs:annotation>
↪<xs:documentation>Description of the Transfer Resource</xs:documentation>
↪</xs:annotation>
↪xs:element>
↪<xs:element name="DestPath" type="xs:string">
↪<xs:annotation>
↪<xs:documentation>Remote Root Destination Path of the Transfer Resource</
↪xs:documentation>
↪</xs:annotation>
↪xs:element>
↪<xs:element name="RelativePath" type="xs:string" minOccurs="0">
↪<xs:annotation>
↪<xs:documentation>Remote Relative Path of the Transfer Resource</xs:documentation>
↪</xs:annotation>
↪xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

```

(continues on next page)

(continued from previous page)

```

</xs:element>
<!-- Secondary Analysis-related fields.-->
<xs:element name="Secondary">
  <xs:annotation>
    <xs:documentation>Container for the primary analysis related
↳data. </xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AutomationName" type="xs:string">
        <xs:annotation>
          <xs:documentation>The secondary
↳analysis protocol name specified in the sample sheet. Ignored by secondary. </
↳xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="AutomationParameters" minOccurs="0">
        <xs:annotation>
          <xs:documentation>The parameters for
↳secondary analysis specified in the sample sheet. Ignored by secondary. </
↳xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name=
↳"AutomationParameter" type="pbbase:DataEntityType" minOccurs="0" maxOccurs=
↳"unbounded">
              <xs:annotation>
                <xs:documentation>One or more secondary analysis parameters, such as JobName,
↳Workflow, etc..</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="CellCountInJob" type="xs:int">
        <xs:annotation>
          <xs:documentation>The number of cells
↳in this secondary analysis job, identified by the secondary analysis parameter
↳'JobName'. Supports automated secondary analysis. </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Contains custom key/value pairs defined by the customer -->
<xs:element name="UserDefinedFields" type="pbbase:UserDefinedFieldsType">
  <xs:annotation>
    <xs:documentation>A set of key-value pairs specified by a
↳user via the run input mechanism. Note that uniqueness of keys is not enforced here,
↳and so may contain duplicate keys. </xs:documentation>
  </xs:annotation>
</xs:element>
  <xs:element name="KeyValue">
    <xs:annotation>
      <xs:documentation>One custom, possibly non-unique, key-value
↳pair. </xs:documentation>
    </xs:annotation>
  </xs:element>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="key" type="xs:string" use=
↪ "required">
                        <xs:annotation>
                            <xs:documentation>Key
↪ (attribute) and Value (element content). </xs:documentation>
                        </xs:annotation>
                    </xs:attribute>
                    <xs:attribute name="label" type="xs:string"
↪ use="optional"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:simpleType name="PapOutputFile">
        <xs:annotation>
            <xs:documentation>Defines a list of available file output
↪ types from primary output that can be copied out to the CollectionPathUri. The
↪ types Pulse, Base, Fasta, and Fastq are for legacy use only.</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="Trace"/>
            <xs:enumeration value="Fasta"/>
            <xs:enumeration value="Baz"/>
            <xs:enumeration value="Bam"/>
            <xs:enumeration value="DarkFrame"/>
            <xs:enumeration value="StatsH5"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>

```

9.5 PacBioDatasets

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2015 rel. 3 sp1 (x64) (http://www.altova.com) by Lakhvir Rai
↪ (Pacific Biosciences) -->
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd" xmlns:xs="http://
↪ www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↪ PacBioBaseDataModel.xsd" xmlns:pbmeta="http://pacificbiosciences.com/
↪ PacBioCollectionMetadata.xsd" xmlns:pbsample="http://pacificbiosciences.com/
↪ PacBioSampleInfo.xsd" targetNamespace="http://pacificbiosciences.com/PacBioDatasets.
↪ xsd" elementFormDefault="qualified" id="PacBioDatasets">
    <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↪ schemaLocation="PacBioBaseDataModel.xsd"/>
    <xs:import namespace="http://pacificbiosciences.com/PacBioCollectionMetadata.
↪ xsd" schemaLocation="PacBioCollectionMetadata.xsd"/>
    <xs:import namespace="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
↪ schemaLocation="PacBioSampleInfo.xsd"/>
    <xs:element name="AlignmentSet">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>DataSets for aligned subreads and CCS reads.
↪</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="AlignmentSetType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="BarcodeSet">
        <xs:annotation>
          <xs:documentation>DataSets of Barcodes. Basically a thin_
↪metadata layer on top of the barcode FASTA.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="BarcodeSetType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="ConsensusAlignmentSet">
        <xs:annotation>
          <xs:documentation>DataSets of aligned CCS reads.</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="AlignmentSetType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="ConsensusReadSet">
        <xs:annotation>
          <xs:documentation>DataSets of CCS reads (typically in_
↪unaligned BAM format).</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="DataSetType">
              <xs:sequence>
                <xs:element name="DataSetMetadata"
↪type="DataSetMetadataType"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="Contigs">
        <xs:annotation>
          <xs:documentation>List of contigs in a ContigSet</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="pbbase:BaseEntityType">
              <xs:sequence>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="Contig" minOccurs="0"
  maxOccurs="unbounded">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension
        base="pbbase:BaseEntityType">
        <xs:attribute name="Length" use="required"/>
        <xs:attribute name="Digest" use="required"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="ContigSet">
  <xs:annotation>
    <xs:documentation>DataSets of contigs sequences. Basically a
    thin metadata layer on top of a contigs FASTA (e.g. from HGAP).</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ContigSetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="DataSet" type="DataSetType"/>
<xs:element name="DataSetRoot" type="DataSetRootType"/>
<xs:element name="HdfSubreadSet">
  <xs:annotation>
    <xs:documentation>DataSets of subreads in bax.h5 or bas.h5
    format.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ReadSetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="ReferenceSet">
  <xs:annotation>
    <xs:documentation>DataSets of reference sequences. Replaces
    the reference.info.xml.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ContigSetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="GmapReferenceSet">
  <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        <xs:documentation>DataSets of reference sequences, with GMAP
↪indices.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="ContigSetType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="SubreadSet">
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="SubreadSetType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="Subsets">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Subset" type="SubsetType" maxOccurs=
↪"unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="AlignmentSetMetadataType">
        <xs:complexContent>
            <xs:extension base="DataSetMetadataType">
                <xs:sequence>
                    <xs:element name="Aligner" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AlignmentSetType">
        <xs:annotation>
            <xs:documentation>Type for DataSets consisting of aligned
↪subreads and CCS reads.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="DataSetType">
                <xs:sequence>
                    <xs:element name="DataSetMetadata" type=
↪"AlignmentSetMetadataType" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="BarcodeSetMetadataType">
        <xs:complexContent>
            <xs:extension base="DataSetMetadataType">
                <xs:sequence>
                    <xs:element name="BarcodeConstruction" type=
↪"xs:string"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

<xs:complexType name="BarcodeSetType">
  <xs:annotation>
    <xs:documentation>Type for the Barcode DataSet.</
↪xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataSetType">
      <xs:sequence>
        <xs:element name="DataSetMetadata" type=
↪"BarcodeSetMetadataType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ContigSetMetadataType">
  <xs:complexContent>
    <xs:extension base="DataSetMetadataType">
      <xs:sequence>
        <xs:element name="Organism" type="xs:string" ↪
↪minOccurs="0"/>
        <xs:element name="Ploidy" type="xs:string" ↪
↪minOccurs="0"/>
        <xs:element ref="Contigs"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ContigSetType">
  <xs:annotation>
    <xs:documentation>Type for a Contig DataSet.</
↪xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataSetType">
      <xs:sequence>
        <xs:element name="DataSetMetadata" type=
↪"ContigSetMetadataType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="DataSetMetadataType">
  <xs:annotation>
    <xs:documentation>Extend this type to provide DataSetMetadata ↪
↪element in each DataSet.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="TotalLength" type="xs:long"/>
    <xs:element name="NumRecords" type="xs:int"/>
    <xs:element name="Provenance" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name=
↪"CommonServicesInstanceId" type="xs:IDREF" minOccurs="0"/>
          <xs:element name="CreatorUserId" type=
↪"xs:IDREF" minOccurs="0"/>
          <xs:element name="ParentJobId" type=
↪"xs:IDREF" minOccurs="0"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="ParentTool" type=
↪ "pbbase:BaseEntityType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="CreatedBy" use="required">
        <xs:simpleType>
            <xs:restriction base=
↪ "xs:string">
                <xs:enumeration value=
↪ "Instrument"/>
                <xs:enumeration value=
↪ "User"/>
                <xs:enumeration value=
↪ "AnalysisJob"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:complexType name="DataSetRootType">
    <xs:choice>
        <xs:element ref="AlignmentSet" minOccurs="0"/>
        <xs:element ref="BarcodeSet" minOccurs="0"/>
        <xs:element ref="ConsensusAlignmentSet" minOccurs="0"/>
        <xs:element ref="ConsensusReadSet" minOccurs="0"/>
        <xs:element ref="ContigSet" minOccurs="0"/>
        <xs:element ref="HdfSubreadSet" minOccurs="0"/>
        <xs:element ref="ReferenceSet" minOccurs="0"/>
        <xs:element ref="SubreadSet" minOccurs="0"/>
    </xs:choice>
</xs:complexType>
<xs:complexType name="DataSetType">
    <xs:complexContent>
        <xs:extension base="pbbase:StrictEntityType">
            <xs:sequence>
                <xs:element ref="pbbase:ExternalResources"/>
                <xs:element name="Filters" minOccurs="0">
                    <xs:annotation>
                        <xs:documentation>The set of
↪ filters defined here apply to the resident data set. Should DataSet subsets be
↪ created out of this parent DataSet, each sub-DataSet may contain its own filters.</
↪ xs:documentation>
                    </xs:annotation>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="DataSets" minOccurs="0">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref=
↪ "Filter" type="pbbase:FilterType" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="DataSets" minOccurs="0">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref=
↪ "DataSet" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:sequence>

```

(continues on next page)

(continued from previous page)

```

                                </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:extension>
                    </xs:complexContent>
                </xs:complexType>
                <xs:complexType name="ReadSetMetadataType">
                    <xs:complexContent>
                        <xs:extension base="DataSetMetadataType">
                            <xs:sequence>
                                <xs:element ref="pbmeta:Collections"
minOccurs="0"/>
                                <xs:element ref="pbsample:BioSamples"
minOccurs="0"/>
                                <xs:element name="SummaryStats" minOccurs="0">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name=
"AdapterDimerFraction" type="xs:float"/>
                                            <xs:element name=
"ShortInsertFraction" type="xs:float"/>
                                            <xs:element name=
"NumSequencingZmws" type="xs:int"/>
                                            <xs:element name=
"ProdDist" type="pbbase:StatsDiscreteDistType"/>
                                            <xs:element name=
"ReadTypeDist" type="pbbase:StatsDiscreteDistType"/>
                                            <xs:element name=
"ReadLenDist" type="pbbase:StatsContinuousDistType"/>
                                            <xs:element name=
"ReadQualDist" type="pbbase:StatsContinuousDistType"/>
                                            <xs:element name=
"ControlReadLenDist" type="pbbase:StatsContinuousDistType"/>
                                            <xs:element name=
"ControlReadQualDist" type="pbbase:StatsContinuousDistType"/>
                                            <xs:element name=
"MedianInsertDist" type="pbbase:StatsContinuousDistType"/>
                                            <xs:element name=
"InsertReadLenDist" type="pbbase:StatsContinuousDistType"/>
                                            <xs:element name=
"InsertReadQualDist" type="pbbase:StatsContinuousDistType"/>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:extension>
                    </xs:complexContent>
                </xs:complexType>
                <xs:complexType name="ReadSetType">
                    <xs:annotation>
                        <xs:documentation>Type for DataSets consisting of unaligned
subreads and CCS reads DataSets</xs:documentation>
                    </xs:annotation>
                    <xs:complexContent>
                        <xs:extension base="DataSetType">
                            <xs:sequence>
                                <xs:element name="DataSetMetadata" type=
"ReadSetMetadataType" minOccurs="0"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SubreadSetMetadataType">
    <xs:complexContent>
      <xs:extension base="DataSetMetadataType">
        <xs:sequence>
          <xs:element name="AverageSubreadLength" type=
↪ "xs:int"/>
          <xs:element name="AverageSubreadQuality" type=
↪ "xs:float"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SubreadSetType">
    <xs:complexContent>
      <xs:extension base="ReadSetType"/>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SubsetType">
    <xs:complexContent>
      <xs:extension base="pbbase:StrictEntityType">
        <xs:sequence>
          <xs:element name="Filters" minOccurs="0">
            <xs:annotation>
              <xs:documentation>The set of ↵
↪ filters defined here apply to the resident data set. Should DataSet subsets be ↵
↪ created out of this parent DataSet, each sub-DataSet may contain its own filters.</
↪ xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element name=
↪ "Filter" type="pbbase:FilterType" maxOccurs="unbounded"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element ref="pbbase:DataPointers" ↵
↪ minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>

```

9.6 PacBioDeclData

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="UOM" type="xs:string"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="Name" type="xs:anyURI"/>
<xs:element name="Scope" type="xs:string"/>
<xs:element name="Value" type="xs:string"/>
<xs:element name="Version" type="T_Version"/>
<xs:element name="Category" type="xs:string"/>
<xs:element name="DeclData" type="T_DeclData"/>
<xs:element name="Specifier" type="T_Specifier"/>
<xs:element name="ValueType" type="xs:string"/>
<xs:element name="DeclDataSet" type="T_DeclDataSet"/>
<xs:element name="Description" type="xs:string"/>
<xs:element name="ComplexValue" type="xs:string"/>
<xs:element name="DefaultValue" type="xs:string"/>
<xs:element name="Classification" type="xs:string"/>
<xs:complexType name="T_DeclData">
  <xs:sequence>
    <xs:element ref="Classification"/>
    <xs:element ref="Name"/>
    <xs:element ref="Description"/>
    <xs:element ref="Category"/>
    <xs:element ref="DefaultValue"/>
    <xs:element ref="Value"/>
    <xs:element ref="ValueType"/>
    <xs:element ref="UOM"/>
    <xs:element ref="Scope"/>
    <xs:element ref="Version"/>
    <xs:element ref="Specifier"/>
    <xs:element ref="ComplexValue"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_Specifier">
</xs:complexType>
<xs:complexType name="T_DeclDataSet">
  <xs:sequence>
    <xs:element ref="DeclData" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:simpleType name="T_Version">
  <xs:restriction base="xs:decimal">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

9.7 PacBioPartNumbers

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 3 (x64) (http://www.altova.com) by
efayad@pacificbiosciences.com (Pacific Biosciences) -->
<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioPartNumbers.xsd" xmlns:xs=
"http://www.w3.org/2001/XMLSchema" xmlns:xi="http://www.w3.org/2003/XInclude"
xmlns:xml="http://www.w3.org/XML/1998/namespace" xmlns:pbbase="http://
pacificbiosciences.com/PacBioBaseDataModel.xsd" xmlns:pbdm="http://
pacificbiosciences.com/PacBioDataModel.xsd" xmlns:pbrk="http://pacificbiosciences.com/PacBioReagentKit.xsd" targetNamespace="http://pacificbiosciences.com/
PacBioPartNumbers.xsd" elementFormDefault="qualified" id="PacBioPartNumbers">

```

(continues on next page)

(continued from previous page)

```

<xs:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation=
↪ "http://www.w3.org/2001/03/xml.xsd"/>
<xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↪ schemaLocation="PacBioBaseDataModel.xsd"/>
<xs:import namespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
↪ schemaLocation="PacBioReagentKit.xsd"/>
<xs:element name="PacBioPartNumbers" type="PacBioPartNumbersType">
  <xs:annotation>
    <xs:documentation>The root element of the Part Numbers </
↪ xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="SequencingKit" type="pbrk:SupplyKitSequencing"/>
<xs:element name="BindingKit" type="pbbase:SupplyKitBinding"/>
<xs:element name="TemplatePrepKit" type="pbbase:SupplyKitTemplate"/>
<xs:element name="ControlKit" type="pbbase:SupplyKitControl"/>
<xs:element name="CellPackKit" type="pbbase:SupplyKitCellPack"/>
<xs:element name="OtherKit">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="pbbase:PartNumberType">
        <xs:attribute name="MaxCollections" type=
↪ "xs:int"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:simpleType name="PartTypes">
  <xs:restriction base="xs:string">
    <xs:enumeration value="WFA"/>
    <xs:enumeration value="BDK"/>
    <xs:enumeration value="TPK"/>
    <xs:enumeration value="SQK"/>
    <xs:enumeration value="CPK"/>
    <xs:enumeration value="OSE"/>
    <xs:enumeration value="CMO"/>
    <xs:enumeration value="Other"/>
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="PacBioPartNumbersType">
  <xs:sequence>
    <xs:element ref="pbbase:KeyValueMap"/>
    <xs:element name="Automations" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Automation" type=
↪ "pbbase:AutomationType" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="BindingKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the binding kit part
↪ numbers. A list of incompatible part numbers and automations is available to
↪ specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
          <xs:element ref="BindingKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="TemplatePrepKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the sample prep kit
↪part numbers. A list of incompatible part numbers and automations is available to
↪specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="TemplatePrepKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SequencingKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the sequencing kit
↪part numbers. A list of incompatible part numbers and automations is available to
↪specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="SequencingKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="ControlKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the DNA control
↪complex part numbers. A list of incompatible part numbers and automations is
↪available to specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="ControlKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="CellPackKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the cell tray part
↪numbers. A list of incompatible part numbers and automations is available to
↪specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="CellPackKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        </xs:element>
        <xs:element name="OtherKits" minOccurs="0">
            <xs:annotation>
                <xs:documentation>A placeholder for
↳ miscellaneous parts, such as OS Enzyme tubes</xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="OtherKit" maxOccurs=
↳ "unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="IncompatibleParts" minOccurs="0">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="IncompatiblePart"
↳ type="pbbase:IncompatiblePairType" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:schema>

```

9.8 PacBioPrimaryMetrics

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!--
*****
**
**  File Name: PipeStats.xsd
**          XML schema defining pipeline statistics
**
**  Programmer: Aseneth Lopez, Phil McClueg
**
**  $Id$
**
*****
**  History:
**
**      2010/03/11 pmclurg - created to correspond to new Functional Specs
**
*****
-->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioPipelineStats.xsd" xmlns:xs=
↳ "http://www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↳ PacBioBaseDataModel.xsd" targetNamespace="http://pacificbiosciences.com/
↳ PacBioPipelineStats.xsd" elementFormDefault="qualified" id="PacBioPipelineStats">
    <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↳ schemaLocation="PacBioBaseDataModel.xsd"/>
<!--

```

(continues on next page)

(continued from previous page)

```

Movie Metrics and Classification

-->
    <!--
MovieContext
-->
    <xs:element name="MovieName" type="xs:string"/>
    <!--
MovieLength
-->
    <xs:element name="MovieLength" type="xs:double"/>
    <!--
NumDroppedFrames
-->
    <xs:element name="NumFramesDropped" type="xs:double"/>
    <!--
NumSequencingZmws
-->
    <xs:element name="NumSequencingZmws" type="xs:double"/>
    <!--
TraceFileSize
-->
    <xs:element name="TraceFileSize" type="xs:long"/>
    <!--
PulseFileSize
-->
    <xs:element name="PulseFileSize" type="xs:long"/>
    <!--
BaseFileSize
-->
    <xs:element name="BaseFileSize" type="xs:long"/>
    <!--
AdapterDimerFraction
-->
    <xs:element name="AdapterDimerFraction" type="xs:double"/>
    <!--
ShortInsertFraction
-->
    <xs:element name="ShortInsertFraction" type="xs:double"/>
    <!--
FractionFailedZmwClippedLow
-->
    <xs:element name="FailedZmwClippedLowFraction" type="xs:double"/>
    <!--
FractionFailedZmwClippedHigh
-->
    <xs:element name="FailedZmwClippedHighFraction" type="xs:double"/>
    <!--
TotalBaseFraction
-->
    <xs:element name="TotalBaseFractionValue" type="xs:double"/>
    <xs:complexType name="TotalBaseFraction">
        <xs:sequence>
            <xs:element ref="TotalBaseFractionValue"/>
        </xs:sequence>
        <xs:attribute name="Channel" type="xs:string" use="required"/>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

    <xs:element name="TotalBaseFractionPerChannel" type="TotalBaseFraction"/>
    <!--
PkMidCV
-->
    <xs:element name="PkMidCVValue" type="xs:double"/>
    <xs:complexType name="PkMidCV">
        <xs:sequence>
            <xs:element ref="PkMidCVValue"/>
        </xs:sequence>
        <xs:attribute name="Channel" type="xs:string" use="required"/>
    </xs:complexType>
    <xs:element name="PkMidCVPerChannel" type="PkMidCV"/>
    <!--
Total number of counts in the sample
-->
    <xs:element name="SampleSize" type="xs:int">
        <xs:annotation>
            <xs:documentation>Number of data values in the sample</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Mean of the sample
-->
    <xs:element name="SampleMean" type="xs:double">
        <xs:annotation>
            <xs:documentation>Mean of the sample</xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Median of the sample
-->
    <xs:element name="SampleMed" type="xs:double">
        <xs:annotation>
            <xs:documentation>Median of the sample</xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Standard Deviation of the sample
-->
    <xs:element name="SampleStd" type="xs:double">
        <xs:annotation>
            <xs:documentation>StandardDeviation of the sample</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
95th Percentile of the sample
-->
    <xs:element name="Sample95thPct" type="xs:double">
        <xs:annotation>
            <xs:documentation>95th Percentile of the sample</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Number of Bins in Histogram

```

(continues on next page)

(continued from previous page)

```

-->
    <xs:element name="NumBins" type="xs:int">
        <xs:annotation>
            <xs:documentation>Number of Bins in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Bin Counts in Histogram
-->
    <xs:element name="BinCount" type="xs:int">
        <xs:annotation>
            <xs:documentation>Bin Counts in Histogram</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Bin Width in Histogram
-->
    <xs:element name="BinWidth" type="xs:double">
        <xs:annotation>
            <xs:documentation>Bin Width in Histogram</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Histogram outlier minimum value
-->
    <xs:element name="MinOutlierValue" type="xs:double">
        <xs:annotation>
            <xs:documentation>Minimum Outlier Value in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Histogram minimum bin value (less outliers)
-->
    <xs:element name="MinBinValue" type="xs:double">
        <xs:annotation>
            <xs:documentation>Minimum Value (less outliers) in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Histogram maximum bin value (less outliers)
-->
    <xs:element name="MaxBinValue" type="xs:double">
        <xs:annotation>
            <xs:documentation>Maximum Value (less outliers) in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Histogram outlier maximum value
-->
    <xs:element name="MaxOutlierValue" type="xs:double">
        <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        <xs:documentation>Maximum Value in Histogram</
↪xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    "Human-Readable" description of movie metric
    -->
    <xs:element name="MetricDescription" type="xs:string">
        <xs:annotation>
            <xs:documentation>Description of metric</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Bin Labels for discrete distributions
    -->
    <xs:element name="BinLabel" type="xs:string">
        <xs:annotation>
            <xs:documentation>Bin Label for Discrete Distribution</
↪xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Continuous distribution class
    <xs:complexType name="ContinuousDist">
        <xs:sequence>
            <xs:element ref="SampleSize"/>
            <xs:element ref="SampleMean"/>
            <xs:element ref="SampleMed"/>
            <xs:element ref="SampleStd"/>
            <xs:element ref="Sample95thPct"/>
            <xs:element ref="NumBins"/>
            <xs:element ref="BinCount" minOccurs="1" maxOccurs="unbounded
↪"/>
            <xs:element ref="BinWidth"/>
            <xs:element ref="MinOutlierValue"/>
            <xs:element ref="MinBinValue"/>
            <xs:element ref="MaxBinValue"/>
            <xs:element ref="MaxOutlierValue"/>
            <xs:element ref="MetricDescription"/>
        </xs:sequence>
        <xs:attribute name="Channel" type="xs:string"/>
    </xs:complexType> -->
    <!--
    Discrete distribution class
    <xs:complexType name="DiscreteDist">
        <xs:sequence>
            <xs:element ref="NumBins"/>
            <xs:element ref="BinCount" minOccurs="1" maxOccurs="unbounded
↪"/>
            <xs:element ref="MetricDescription"/>
            <xs:element ref="BinLabel" minOccurs="1" maxOccurs="unbounded
↪"/>
        </xs:sequence>
    </xs:complexType> -->
    <!--
    Time series (for time-dependent metrics)
    <xs:complexType name="TimeSeries">

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
            <xs:element name="TimeUnits" type="xs:string"/>
            <xs:element name="ValueUnits" type="xs:string"/>
            <xs:element name="StartTime" type="xs:float"/>
            <xs:element name="MeasInterval" type="xs:float"/>
            <xs:element name="Val" type="xs:float" minOccurs="0"
↪maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType> -->
    <!--
Specific Distributions
-->
    <!--
Control Read Length Distribution
-->
        <xs:element name="ControlReadLenDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Control Read Quality Distribution
-->
        <xs:element name="ControlReadQualDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Baseline Level Distribution - all holes
-->
        <xs:element name="BaselineLevelDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Baseline Standard Deviation Distribution - all holes
-->
        <xs:element name="BaselineStdDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Movie Read Quality Distribution - all holes
-->
        <xs:element name="MovieReadQualDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Productivity Distribution - all holes
-->
        <xs:element name="ProdDist" type="pbbase:StatsDiscreteDistType"/>
    <!--
ReadType Distribution
-->
        <xs:element name="ReadTypeDist" type="pbbase:StatsDiscreteDistType"/>
    <!--
Pulse Rate Distribution - productive holes
-->
        <xs:element name="PulseRateDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Mean Pulse Width Distribution - productive holes
-->
        <xs:element name="PulseWidthDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Base Rate (global) Distribution - productive holes, HQ regions
-->
        <xs:element name="BaseRateDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Mean Base Width Distribution - productive holes, HQ regions
-->
        <xs:element name="BaseWidthDist" type="pbbase:StatsContinuousDistType"/>
    <!--

```

(continues on next page)

(continued from previous page)

```

Mean Base IPD Distribution - productive holes, HQ regions
-->
    <xs:element name="BaseIpdDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Local Base Rate Distribution - productive holes, HQ regions.
I.e., an estimate of the polymerase rate, excluding pauses.
-->
    <xs:element name="LocalBaseRateDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Number of unfiltered basecalls Distribution - productive holes
-->
    <xs:element name="NumUnfilteredBasecallsDist" type=
→ "pbbase:StatsContinuousDistType"/>
    <!--
Polymerase Read Length Distribution - productive holes
-->
    <xs:element name="ReadLenDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Polymerase Read Quality Distribution - productive holes
-->
    <xs:element name="ReadQualDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Distribution of the ratio (Polymerase Read Len)/(Num Unfiltered Basecalls)
-->
    <xs:element name="HqBaseFractionDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Polymerase Read-Mean Base Quality Value Distribution - productive holes
-->
    <xs:element name="RmBasQvDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Best Estimate (Insert) Read Length Distribution
-->
    <xs:element name="InsertReadLenDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Best Estimate (Insert) Read Quality (Predicted Accuracy) Distribution
-->
    <xs:element name="InsertReadQualDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Local Yield Distribution - % of productive holes locally
-->
    <xs:element name="LocalYieldDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Local Snr Distribution - snr per channel locally
-->
    <xs:element name="LocalSnrDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Trace Clipped Fraction Distribution - Total % traces clipped per channel
-->
    <xs:element name="TraceClippedFractionDist" type=
→ "pbbase:StatsContinuousDistType"/>
    <!--
Trace Low Clipped Fraction Distribution - % traces clipped low per channel
-->
    <xs:element name="TraceLowClippedFractionDist" type=
→ "pbbase:StatsContinuousDistType"/>
    <!--
Trace High Clipped Fraction Distribution - % traces clipped high per channel

```

(continues on next page)

(continued from previous page)

```

-->
  <xs:element name="TraceHighClippedFractionDist" type=
↪ "pbbase:StatsContinuousDistType"/>
  <!--
  Pausiness Distribution - % ipds above a threshold
-->
    <xs:element name="PausinessDist" type="pbbase:StatsContinuousDistType"/>
    <!--
  Adapter Dimer Distribution - % distribution of mean insert length
-->
    <xs:element name="MedianInsertDist" type="pbbase:StatsContinuousDistType"/>
    <!--
  PkMid CV Distribution - % distribution of coefficient of variation of PkMid
-->
    <xs:element name="PkMidCVDist" type="pbbase:StatsContinuousDistType"/>
    <!--
  SNR Distribution - productive holes
-->
    <xs:element name="SnrDist" type="pbbase:StatsContinuousDistType"/>
    <!--
  HQ Region SNR Distribution - robust estimate of SNR distribution over productive_
↪ holes
-->
    <xs:element name="HqRegionSnrDist" type="pbbase:StatsContinuousDistType"/>
    <!--
  HqBasPkMid Distribution - productive holes
-->
    <xs:element name="HqBasPkMidDist" type="pbbase:StatsContinuousDistType"/>
    <!--
  BaselineLevelSequencing Distribution - productive holes
-->
    <xs:element name="BaselineLevelSequencingDist" type=
↪ "pbbase:StatsContinuousDistType"/>
    <!--
  BaselineLevelAntihole Distribution - productive holes
-->
    <xs:element name="BaselineLevelAntiholeDist" type=
↪ "pbbase:StatsContinuousDistType"/>
    <!--
  BaselineLevelAntimirror Distribution - productive holes
-->
    <xs:element name="BaselineLevelAntimirrorDist" type=
↪ "pbbase:StatsContinuousDistType"/>
    <!--
  BaselineLevelFiducial Distribution - productive holes
-->
    <xs:element name="BaselineLevelFiducialDist" type=
↪ "pbbase:StatsContinuousDistType"/>
    <!--
  Subsystem Failure Metrics, Added in Software v1.3.1
-->
    <xs:element name="MaxPauseFractionVsT" type="xs:double"/>
    <xs:element name="TMaxPauseFraction" type="xs:int"/>
    <xs:element name="MaxSlopePauseFractionVsT" type="xs:double"/>
    <xs:element name="TMaxSlopePauseFraction" type="xs:int"/>
    <xs:element name="MaxBaseRateRatioVsT" type="xs:double"/>
    <xs:element name="TMaxBaseRateRatio" type="xs:int"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="MaxSlopeBaseRateRatioVsT" type="xs:double"/>
<xs:element name="TMaxSlopeBaseRateRatio" type="xs:int"/>
<xs:element name="SgnMaxSlopeBaseRateRatio" type="xs:int"/>
<xs:element name="BaseRateChngStrtToEnd" type="xs:double"/>
<xs:element name="YieldCvOverRegions" type="xs:double"/>
<xs:element name="YieldChngCntrToEdge" type="xs:double"/>
<xs:element name="SnrRatioEdgeToCntr_0" type="xs:double"/>
<xs:element name="SnrRatioEdgeToCntr_2" type="xs:double"/>
<xs:element name="PauseFractionVsT" type="pbbase:StatsTimeSeriesType"/>
<xs:element name="BaseRateRatioVsT" type="pbbase:StatsTimeSeriesType"/>
<!--
Other v1.3.1 Metric Additions
-->
<xs:element name="IsReadsFraction" type="xs:double"/>
<!--
v2.0 Additions - spectral diagnostics
-->
<xs:element name="SpectralDiagRRDist" type="pbbase:StatsContinuousDistType"/>
<!--
PipeStats class for PipelineStats XML
-->
<xs:element name="PipeStats">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="MovieName" minOccurs="1" maxOccurs="1"
↪"/>
      <xs:element ref="MovieLength" minOccurs="1" maxOccurs="
↪"1"/>
      <xs:element ref="NumFramesDropped" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="NumSequencingZmws" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="TraceFileSize" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="PulseFileSize" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="BaseFileSize" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="AdapterDimerFraction" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="ShortInsertFraction" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="IsReadsFraction" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="FailedZmwClippedLowFraction"
↪minOccurs="1" maxOccurs="1"/>
      <xs:element ref="FailedZmwClippedHighFraction"
↪minOccurs="1" maxOccurs="1"/>
      <xs:element ref="ControlReadLenDist" minOccurs="0"
↪maxOccurs="1"/>
      <xs:element ref="ControlReadQualDist" minOccurs="0"
↪maxOccurs="1"/>
      <xs:element ref="ProdDist" minOccurs="1" maxOccurs="1"
↪"/>
      <xs:element ref="ReadTypeDist" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="TotalBaseFractionPerChannel"
↪minOccurs="4" maxOccurs="4"/>

```

(continues on next page)

(continued from previous page)

```

↪maxOccurs="4"/>
↪maxOccurs="4"/>
↪maxOccurs="4"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪"1"/>
↪maxOccurs="1"/>
↪minOccurs="1" maxOccurs="1"/>
↪"1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪"1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="4"/>
↪"4" maxOccurs="4"/>
↪minOccurs="4" maxOccurs="4"/>
↪minOccurs="4" maxOccurs="4"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪>
↪maxOccurs="4"/>
↪maxOccurs="4"/>
↪minOccurs="4" maxOccurs="4"/>
↪"4" maxOccurs="4"/>

<xs:element ref="PkMidCVPerChannel" minOccurs="4"
<xs:element ref="BaselineLevelDist" minOccurs="4"
<xs:element ref="BaselineStdDist" minOccurs="4"
<xs:element ref="MovieReadQualDist" minOccurs="1"
<xs:element ref="PulseRateDist" minOccurs="1"
<xs:element ref="PulseWidthDist" minOccurs="1"
<xs:element ref="BaseRateDist" minOccurs="1"
<xs:element ref="BaseWidthDist" minOccurs="1"
<xs:element ref="BaseIpdDist" minOccurs="1" maxOccurs=
<xs:element ref="LocalBaseRateDist" minOccurs="1"
<xs:element ref="NumUnfilteredBasecallsDist"
<xs:element ref="ReadLenDist" minOccurs="1" maxOccurs=
<xs:element ref="ReadQualDist" minOccurs="1"
<xs:element ref="HqBaseFractionDist" minOccurs="1"
<xs:element ref="RmBasQvDist" minOccurs="1" maxOccurs=
<xs:element ref="InsertReadLenDist" minOccurs="0"
<xs:element ref="InsertReadQualDist" minOccurs="0"
<xs:element ref="LocalYieldDist" minOccurs="1"
<xs:element ref="LocalSnrDist" minOccurs="4"
<xs:element ref="TraceClippedFractionDist" minOccurs=
<xs:element ref="TraceLowClippedFractionDist"
<xs:element ref="TraceHighClippedFractionDist"
<xs:element ref="PausinessDist" minOccurs="1"
<xs:element ref="MedianInsertDist" minOccurs="1"
<xs:element ref="SnrDist" minOccurs="4" maxOccurs="4"/
<xs:element ref="HqRegionSnrDist" minOccurs="4"
<xs:element ref="HqBasPkMidDist" minOccurs="4"
<xs:element ref="BaselineLevelSequencingDist"
<xs:element ref="BaselineLevelAntiholeDist" minOccurs=

```

(continues on next page)

(continued from previous page)

```
<xs:minOccurs="4" maxOccurs="4"/>  
    <"4" maxOccurs="4"/>  
  
    <maxOccurs="4"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <"1" maxOccurs="1"/>  
  
    <" " maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <"1" maxOccurs="1"/>  
  
    <" " maxOccurs="1"/>  
  
    <"1" maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    <maxOccurs="1"/>  
  
    </xs:sequence>  
    <xs:attribute name="Version" type="xs:string" use="optional">  
        <xs:annotation>  
            <xs:documentation>An optional identifier_  
denoting the revision of this particular entity</xs:documentation>  
        </xs:annotation>  
    </xs:attribute>  
</xs:element>  
</xs:schema>
```

9.9 PacBioReagentKit

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 4 sp1 (x64) (http://www.altova.com) by
refavad@pacificbiosciences.com (Pacific Biosciences) -->
```

(continues on next page)

(continued from previous page)

```

<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioReagentKit.xsd" xmlns:xs="http://
↳/www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↳PacBioBaseDataModel.xsd" xmlns:pbdm="http://pacificbiosciences.com/PacBioDataModel.
↳xsd" targetNamespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
↳elementFormDefault="qualified" id="PacBioReagentKit">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↳schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:element name="PacBioReagentKit">
    <xs:annotation>
      <xs:documentation>The root element of the reagent kit
↳standalone file</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ReagentKit"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Reagent" type="ReagentType"/>
  <xs:element name="ReagentKit" type="ReagentKitType"/>
  <xs:element name="ReagentTube" type="ReagentTubeType"/>
  <xs:element name="ReagentPlateRow" type="ReagentPlateRowType"/>
  <xs:complexType name="SupplyKitSequencing">
    <xs:annotation>
      <xs:documentation>A more specific template kit representation
↳ (includes SupplyKit fields). </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="pbbase:PartNumberType">
        <xs:sequence>
          <xs:element name="ReagentAutomationName" type=
↳"xs:string" minOccurs="0">
            <xs:annotation>
              <xs:documentation>The reagent-
↳mixing protocol used. </xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="ReagentTubes" type=
↳"SupplyKitSequencing" minOccurs="0" maxOccurs="2">
            <xs:annotation>
              <xs:documentation>Tubes
↳associated with the reagent kit - can have up to two; don't forget to set the
↳location, 0 or 1</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="SequencingChemistry" type=
↳"pbbase:SequencingChemistry" minOccurs="0"/>
          <xs:element name="SequencingKitDefinition"
↳type="ReagentKitType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="Location" type="xs:int" default="0
↳">
          <xs:annotation>
            <xs:documentation>The location of the
↳supply kit - for a reagent plate, it could be 0 or 1, and for a tube it could be 0
↳or 1</xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        </xs:attribute>
        <xs:attribute name="MaxCollections" type="xs:int"
↪default="8">

            <xs:annotation>
                <xs:documentation>The number of
↪collections this supply kit is capable of</xs:documentation>
            </xs:annotation>
        </xs:attribute>
    </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:attribute name="ReagentKey" type="ReagentKey" use=
↪"required"/>
            <xs:attribute name="PlateColumn" type="xs:string" use=
↪"required"/>
            <xs:attribute name="Volume" type="xs:int" use=
↪"required"/>
            <xs:attribute name="DeadVolume" type="xs:int" use=
↪"required"/>
            <xs:attribute name="ActiveInHour" type="xs:int" use=
↪"required"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentKitType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:sequence>
                <xs:element name="Reagents">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref=
↪"Reagent" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="ReagentTubes">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref=
↪"ReagentTube" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="ReagentPlateRows">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref=
↪"ReagentPlateRow" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:element ref="pbbase:ChemistryConfig"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element ref="pbbase:Defaults" minOccurs="0"/>
↪"/>
<xs:element name="Automations" nillable="true"
↪" minOccurs="0">
    <xs:annotation>
        <xs:documentation>Automations_
↪that are deemed compatible with this kit. Parameters specified within an_
↪automation will override a parameter with the same name and data type specified in_
↪the above Defaults section</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name=
↪"Automation" type="pbbase:AutomationType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="ProductCode" type="xs:string"/>
<xs:attribute name="PlateType" type="xs:string"/>
<xs:attribute name="ActiveInHour" type="xs:int"/>
<xs:attribute name="BasesPerSecond" type="xs:decimal"/
↪>
    <xs:attribute name="AcquisitionCount" type="xs:int"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentTubeType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:attribute name="ProductCode" type="xs:string" use=
↪"required"/>
            <xs:attribute name="ReagentKey" type="ReagentKey" use=
↪"required"/>
            <xs:attribute name="Volume" type="xs:short" use=
↪"required"/>
            <xs:attribute name="UsageVolume" type="xs:double" use=
↪"required"/>
            <xs:attribute name="UsageKey" type="xs:string" use=
↪"required"/>
            <xs:attribute name="DeadVolume" type="xs:short" use=
↪"required"/>
            <xs:attribute name="ActiveInHour" type="xs:int" use=
↪"required"/>
            <xs:attribute name="TubeWellType" type="TubeSize" use=
↪"required"/>
            <xs:attribute name="ReagentTubeType" type=
↪"TubeLocation" use="required"/>
            <xs:attribute name="InitialUse" type="xs:dateTime"
↪use="optional"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentPlateRowType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:attribute name="PlateRow" type="xs:string" use=
↪"required"/>

```

(continues on next page)

(continued from previous page)

```

<xs:attribute name="InitialUse" type="xs:dateTime"
↪use="optional"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:simpleType name="TubeLocation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ReagentTube0"/>
    <xs:enumeration value="ReagentTube1"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="TubeSize">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DeepTube"/>
    <xs:enumeration value="ShallowTube"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="ReagentKey">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Base"/>
    <xs:enumeration value="DTT"/>
    <xs:enumeration value="DilBuffer"/>
    <xs:enumeration value="MineralOil"/>
    <xs:enumeration value="MIXED_DilBuffer"/>
    <xs:enumeration value="MIXED_OS"/>
    <xs:enumeration value="OSbuffer"/>
    <xs:enumeration value="OSenzyme"/>
    <xs:enumeration value="PhospholinkedNT"/>
    <xs:enumeration value="SABuffer"/>
    <xs:enumeration value="Spike"/>
    <xs:enumeration value="Streptavidin"/>
    <xs:enumeration value="SubstrateOS"/>
    <xs:enumeration value="TSQ"/>
    <xs:enumeration value="WashBuffer"/>
    <xs:enumeration value="WettingBuffer"/>
    <xs:enumeration value="PCA"/>
    <xs:enumeration value="PCD"/>
    <xs:enumeration value="Analog"/>
    <xs:enumeration value="Sample"/>
    <xs:enumeration value="PEGDilBuffer"/>
    <xs:enumeration value="ExtraBuffer"/>
    <xs:enumeration value="PrewetBuffer"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

9.10 PacBioRightsAndRoles

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2015 rel. 3 (x64) (http://www.altova.com) by
↪efayad@pacificbiosciences.com (Pacific Biosciences) -->
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioRightsAndRoles.xsd" xmlns:xs=
↪"http://www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↪PacBioBaseDataModel.xsd" targetNamespace="http://pacificbiosciences.com/
↪PacBioRightsAndRoles.xsd" elementFormDefault="qualified" id="PacBioRightsAndRoles">
(continues on next page)

```

(continued from previous page)

```

<xs:import namespace="http://pacifichiosciences.com/PacBioBaseDataModel.xsd"
↳schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:element name="RnR">
    <xs:annotation>
      <xs:documentation>Rights and Roles root element</
↳xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Roles" minOccurs="0"/>
        <xs:element ref="AccessRights" minOccurs="0"/>
        <xs:element ref="UserIdentities" minOccurs="0"/>
        <xs:element ref="Projects" minOccurs="0">
          <xs:annotation>
            <xs:documentation>A project is of
↳type AccessRight and can be defined the same way, defined by a URI (ResoureId
↳attribute) and access controlled by using the AccessDisabled attribute, set to
↳'false' to grant users access.
A user (with a role, which has access rights) has one more projects with which they
↳are associated. Via this association, one or more users may own a project, and one
↳or more users may be granted access to a project, simply by being associated with
↳it.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="AuditableEvents" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="CFRp11Enabled" type="xs:boolean" use=
↳"optional" default="false">
        <xs:annotation>
          <xs:documentation>Define whether or not
↳21CFRp11 capabilities are turned on or off (default)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="LDAPUri" type="xs:anyURI">
        <xs:annotation>
          <xs:documentation>LDAP server URI used to
↳authenticate users. If none is specified, local user/password information is used
↳to determine access.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="UserIdentityType">
    <xs:complexContent>
      <xs:extension base="pbbase:StrictEntityType">
        <xs:sequence>
          <xs:element ref="Person">
            <xs:annotation>
              <xs:documentation>An entity
↳to store persons' names and contact info, to associate to users</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element ref="UserPassword">
            <xs:annotation>
              <xs:documentation>An entity
↳storing hashed password and salt - cached here in case LDAP is not availabel to
↳authenticate, or the user does not exist in LDAP (i.e. adhoc user).</
↳xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element ref="ProjectReferences" minOccurs=
↪ "0">

        <xs:annotation>
          <xs:documentation>List of ↪
↪ projects that the user has access to</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="UserName" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:maxLength value="50"/>
          <xs:minLength value="1"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="TokenId" type="xs:string">
      <xs:annotation>
        <xs:documentation>A token to use for ↪
↪ authentication, instead of constantly pinging LDAP with user name/password</
↪ xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="UseLDAPUri" type="xs:boolean">
      <xs:annotation>
        <xs:documentation>If the LDAP server ↪
↪ is specified at the root element, we use the UserName to authenticate against that -
↪ i.e. no password info is necessary to be stored.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Email">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:maxLength value="96"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="MessageAddress">
      <xs:annotation>
        <xs:documentation>e.g. a phone number ↪
↪ to text a message, a twitter handle...</xs:documentation>
      </xs:annotation>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="AccessGrantDate" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:dateTime">
          <xs:minInclusive value="1000-
↪ 01-01T00:00:00"/>
          <xs:maxInclusive value="9999-
↪ 12-31T23:59:59"/>

```

(continues on next page)

(continued from previous page)

```

<xs:pattern value="\p{Nd}{4}-
↪\p{Nd}{2}-\p{Nd}{2}T\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="RoleReference" type="xs:IDREF"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="RoleType">
  <xs:complexContent>
    <xs:extension base="pbbase:StrictEntityType"/>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="AccessRightType">
  <xs:complexContent>
    <xs:extension base="pbbase:StrictEntityType">
      <xs:attribute name="InternalResourceAddress" use=
↪"required">
        <xs:annotation>
          <xs:documentation>e.g. svc://admin
This should support a wildcard specification, such that an entire hierarchy can be_
↪disabled via http://*/Analysis/*</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:maxLength value="50"/>
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="ComponentName">
        <xs:annotation>
          <xs:documentation>e.g. System_
↪Administration</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:maxLength value="255"/>
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="AccessDisabled" type="xs:boolean"
↪use="optional" default="false">
        <xs:annotation>
          <xs:documentation>When the object is_
↪created, by default it is to disable access to a function. This should be set to
↪'false' in order to explicitly enable a function.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="Operation" type="xs:string">
        <xs:annotation>
          <xs:documentation>Define an optional_
↪operation to the resource identifier, e.g. GET, PUT, POST</xs:documentation>
        </xs:annotation>
      </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

<xs:attribute name="RequiresAudit" type="xs:boolean"
↪use="required"/>
<xs:attribute name="RequiresESig" type="xs:boolean"
↪use="required">
    <xs:annotation>
        <xs:documentation>An ESig may only be
↪required, if an audit is required.</xs:documentation>
    </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="AuditableEventType">
    <xs:complexContent>
        <xs:extension base="pbbase:StrictEntityType">
            <xs:sequence>
                <xs:element ref="EventToken">
                    <xs:annotation>
                        <xs:documentation>A handoff
↪event token to be used between system components - maybe superceded by the user's
↪tokenId</xs:documentation>
                    </xs:annotation>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="DateCreated" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:dateTime">
                        <xs:minInclusive value="1000-
↪01-01T00:00:00"/>
                        <xs:maxInclusive value="9999-
↪12-31T23:59:59"/>
                        <xs:pattern value="\p{Nd}{4}-
↪\p{Nd}{2}-\p{Nd}{2}T\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="AuditEventType" use="required">
                <xs:annotation>
                    <xs:documentation>e.g. NewUserCreated
↪</xs:documentation>
                </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="255"/>
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="OldValue">
                <xs:simpleType>
                    <xs:restriction base="xs:base64Binary"
↪">
                    <xs:maxLength value=
↪"2147483647"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

        <xs:attribute name="NewValue" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:base64Binary"
                    <xs:maxLength value=
↪ ">
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="Reason">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="255"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ESig">
            <xs:simpleType>
                <xs:restriction base="xs:base64Binary"
                    <xs:maxLength value=
↪ ">
                    </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="UserIdentityReference" type=
↪ "xs:IDREF"/>
    </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="ProjectReferenceType">
    <xs:attribute name="ProjectId" type="xs:IDREF" use="required">
        <xs:annotation>
            <xs:documentation>Reference to a project ID</
↪ xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="IsOwner" type="xs:boolean">
        <xs:annotation>
            <xs:documentation>Designate if the user has ownership
↪ of this project; note that multiple users may have ownership of the same project.</
↪ xs:documentation>
        </xs:annotation>
    </xs:attribute>
</xs:complexType>
<xs:element name="UserIdentity" type="UserIdentityType">
    <xs:annotation>
        <xs:documentation>A user entity relating to a person
↪ (optional) and role;</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="Role">
    <xs:annotation>
        <xs:documentation>The role a user would play in the system, e.
↪ g. Admin, tech</xs:documentation>
    </xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="RoleType">
              <xs:sequence>
                <xs:element ref="AccessRightReferences"
↳ " minOccurs="0"/>
                <xs:element ref=
↳ "UserIdentityReferences" minOccurs="0"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="AccessRight">
        <xs:annotation>
          <xs:documentation>Define the functions that a role is capable
↳ of accessing</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="AccessRightType">
              <xs:sequence>
                <xs:element ref="RoleReferences"/>
                <xs:element ref=
↳ "AuditableEventReferences" minOccurs="0"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="AuditableEvent" type="AuditableEventType">
        <xs:annotation>
          <xs:documentation>An auditable record is created if required
↳ for an access right</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="Project" type="AccessRightType">
        <xs:annotation>
          <xs:documentation>A project is of type AccessRight and can be
↳ defined the same way, defined by a URI (ResourceId attribute) and access controlled
↳ by using the AccessDisabled attribute, set to 'false' to grant users access.

A user (with a role, which has access rights) has one more projects with which
↳ they're associated. Via this association, one or more users may own a project, and
↳ one or more users may be granted access to a project, simply by being associated
↳ with it.
</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="EventToken">
        <xs:annotation>
          <xs:documentation>A handoff event token to be used between
↳ system components o</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:attribute name="Token">
            <xs:simpleType>

```

(continues on next page)

(continued from previous page)

```

        <xs:restriction base="xs:base64Binary">
            <xs:maxLength value="255"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:element name="Person">
    <xs:annotation>
        <xs:documentation>a table to store persons' full names, to
→associate to users</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:attribute name="FirstName">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="50"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="MiddleName">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="50"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="LastName" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="50"/>
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="PhoneNumber" type="xs:string"/>
        <xs:attribute name="EMail" type="xs:string"/>
        <xs:attribute name="NotifyBySMS" type="xs:boolean"/>
        <xs:attribute name="NotifyByEmail" type="xs:boolean"/>
    </xs:complexType>
</xs:element>
<xs:element name="UserPassword">
    <xs:annotation>
        <xs:documentation>a table storing hashed password and salt</
→xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:attribute name="PasswordHash" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="255"/>
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="Salt" use="required">
            <xs:simpleType>

```

(continues on next page)

(continued from previous page)

```

        <xs:restriction base="xs:string">
            <xs:maxLength value="255"/>
            <xs:minLength value="1"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="Expired" type="xs:boolean" use="required"/>
</>

    </xs:complexType>
</xs:element>
<xs:element name="Roles">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more Role elements
    </xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Role" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>The role a user
would play in the system, e.g. Admin, tech, scientist, PI, etc.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="AccessRights">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more AccessRight
elements</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="AccessRight" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>Define the
functions that a role is capable of accessing
                </xs:documentation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="UserIdentities">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more UserIdentity
elements</xs:documentation>
    </xs:annotation>
    <xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
          <xs:element ref="UserIdentity" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="AuditableEvents">
      <xs:annotation>
        <xs:documentation>An aggregation of one or more
↪ AuditableEvent elements</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="AuditableEvent" maxOccurs="unbounded
↪ "/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Projects">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="Project" maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation>An grouping of
↪ datasets and analysis jobs, with user access controls

```

The Project type extends AccessRight, allowing a project to be defined by a URI
↪ (ResoureId attribute) and access controlled by using the AccessDisabled attribute,
↪ set to 'false' to grant users access.</xs:documentation>

```

            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="AccessRightReferences">
      <xs:annotation>
        <xs:documentation>List of IDREFs to AccessRight element UUIDs
↪ </xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element name="AccessRightReference" type="xs:IDREF
↪ " maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation>IDREF to an
↪ AccessRight element UUID</xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="UserIdentityReferences">
      <xs:annotation>
        <xs:documentation>List of IDREFs to UserIdentity element UUIDs
↪ </xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>

```

(continues on next page)

(continued from previous page)

```

        <xs:element name="UserIdentityReference" type=
↪ "xs:IDREF" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>IDREF to a
↪ UserIdentity element UUID</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="AuditableEventReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to AuditableEvent element
↪ UUIDs</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="AuditableEventReference" type=
↪ "xs:IDREF" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>IDREF to an
↪ AuditableEvent element UUID</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="RoleReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to Role element UUIDs</
↪ xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="RoleReference" type="xs:IDREF"
↪ maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>IDREF to a Role
↪ element UUID</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="ProjectReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to DataUserGroup element
↪ UUIDs</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ProjectReference" type=
↪ "ProjectReferenceType" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>List of 1 or more
↪ projects that the user has access to, or ownership of.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

```

(continues on next page)

(continued from previous page)

```

        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

9.11 PacBioSampleInfo

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 3 (x64) (http://www.altova.com) by
efayad@pacificbiosciences.com (Pacific Biosciences) -->
<?xml-stylesheet type="application/xml" href="metadata2html.xslt"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioSampleInfo.xsd" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
PacBioBaseDataModel.xsd" targetNamespace="http://pacificbiosciences.com/
PacBioSampleInfo.xsd" elementFormDefault="qualified" id="PacBioSampleInfo">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
schemaLocation="PacBioBaseDataModel.xsd"/>
  <!-- The root element for the metadata structure -->
  <xs:complexType name="BarcodedSampleType">
    <xs:annotation>
      <xs:documentation>This is a data type to hold a barcoded
biological sample, or a raw biological sample - so, barcode is optional.</
xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="BioSampleType">
        <xs:sequence>
          <xs:element name="Barcodes" minOccurs="0">
            <xs:annotation>
              <xs:documentation>A list of
biological samples associated with the biological sample</xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element name=
"Barcode" type="pbbase:DNABarcode" minOccurs="0" maxOccurs="unbounded">
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:sequence>
            <xs:documentation>A sequence of barcodes associated with the biological sample</
xs:documentation>
            </xs:sequence>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="BioSampleType">

```

(continues on next page)

(continued from previous page)

```

<xs:annotation>
  <xs:documentation>The actual biological sample; this could be
↳prep'd, or in original form; could be bound, or annealed...</xs:documentation>
</xs:annotation>
<xs:complexContent>
  <xs:extension base="pbbase:StrictEntityType">
    <xs:sequence>
      <xs:element ref="BioSamples" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="DateReceived" type="xs:dateTime">
      <xs:annotation>
        <xs:documentation>Date the sample was
↳received by the lab</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Organism" type="xs:string">
      <xs:annotation>
        <xs:documentation>e.g. HIV, E.coli</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Reference" type="xs:string">
      <xs:annotation>
        <xs:documentation>Name of reference,
↳or pointer to one at e.g. NCBI RefSeq</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="DNAType" type="xs:string">
      <xs:annotation>
        <xs:documentation>shotgun library,
↳amplicon, etc.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Concentration" type="xs:float">
      <xs:annotation>
        <xs:documentation>in ng/uL, e.g. 250</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="QuantificationMethod" type=
↳"xs:string">
      <xs:annotation>
        <xs:documentation>e.g. Qubit</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="SMRTBellConcentration" type=
↳"xs:float">
      <xs:annotation>
        <xs:documentation>in ng/uL, e.g. 4.5</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="SMRTBellQuantificationMethod"
↳type="xs:string">
      <xs:annotation>
        <xs:documentation>e.g. Qubit</
↳xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        </xs:attribute>
        <xs:attribute name="BufferName" type="xs:string">
            <xs:annotation>
                <xs:documentation>e.g. Tris HCl</
↪xs:documentation>

            </xs:annotation>
            </xs:attribute>
            <xs:attribute name="SamplePrepKit" type="xs:string">
                <xs:annotation>
                    <xs:documentation>e.g. SMRTbell_
↪Template Prep Kit</xs:documentation>

            </xs:annotation>
            </xs:attribute>
            <xs:attribute name="TargetLibrarySize" type="xs:string
↪">

                <xs:annotation>
                    <xs:documentation>2000, 10000, 20000</
↪xs:documentation>

                </xs:annotation>
                </xs:attribute>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="BioSamplePointers">
        <xs:annotation>
            <xs:documentation>Back references to other BarcodedSampleType_
↪object UniqueIds which utilize this sample</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:annotation>
                    <xs:documentation>The choice is to either_
↪point to a group of barcoded samples or a single biological sample</
↪xs:documentation>

                </xs:annotation>
                <xs:choice>
                    <xs:element ref="BarcodedSamplePointers"/>
                    <xs:element name="BioSamplePointer" type=
↪"xs:IDREF">

                        <xs:annotation>
                            <xs:documentation>Pointer to_
↪a single biological sample</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:choice>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="BarcodedSamplePointers">
        <xs:annotation>
            <xs:documentation>Back references to other BarcodedSampleType_
↪object UniqueIds which utilize this sample</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BarcodedSamplePointer" type=
↪"xs:IDREF" maxOccurs="unbounded">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
            <xs:documentation>Pointer to a group
↳ of barcoded samples</xs:documentation>
        </xs:annotation>
    </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="BioSamples">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="BioSample" minOccurs="0" maxOccurs=
↳ "unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="BioSample" type="BioSampleType"/>
</xs:schema>

```

9.12 PacBioSeedingData

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Dye" type="T_Dye"/>
    <xs:element name="Dyes" type="T_Dyes"/>
    <xs:element name="User" type="T_User"/>
    <xs:element name="DependOn" type="T_DependOn"/>
    <xs:element name="Protocol" type="T_Protocol"/>
    <xs:element name="PlateType" type="T_PlateType"/>
    <xs:element name="Protocols" type="T_Protocols"/>
    <xs:element name="PlateTypes" type="T_PlateTypes"/>
    <xs:element name="DefaultUsers" type="T_DefaultUsers"/>
    <xs:element name="HiveMetadata" type="T_HiveMetadata"/>
    <xs:element name="ProtocolType" type="T_ProtocolType"/>
    <xs:element name="ReminderTask" type="T_ReminderTask"/>
    <xs:element name="ProtocolTypes" type="T_ProtocolTypes"/>
    <xs:element name="ReminderTasks" type="T_ReminderTasks"/>
    <xs:element name="DependencyItem" type="T_DependencyItem"/>
    <xs:element name="ReagentKitType" type="T_ReagentKitType"/>
    <xs:element name="ReagentKitTypes" type="T_ReagentKitTypes"/>
    <xs:element name="InstDBSeedingData" type="T_InstDBSeedingData"/>
    <xs:element name="ProductPartNumber" type="T_ProductPartNumber"/>
    <xs:element name="ProtocolParameter" type="T_ProtocolParameter"/>
    <xs:element name="PipelineDependency" type="T_PipelineDependency"/>
    <xs:element name="ProductPartNumbers" type="T_ProductPartNumbers"/>
    <xs:element name="CompatibleReagentKit" type="T-CompatibleReagentKit"/>
    <xs:element name="ProtocolTypeParameter" type="T_ProtocolTypeParameter"/>
    <xs:element name="ProtocolTypeParameters" type="T_ProtocolTypeParameters"/>
    <xs:complexType name="T_Dye">
        <xs:attribute name="Base" use="required" type="AT_37"/>
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:attribute name="Nucleotide" use="required" type="AT_5"/>
        <xs:attribute name="WaveLength" use="required" type="AT_5"/>
        <xs:attribute name="DyeConcentration" use="required" type="AT_5"/>
    </xs:complexType>
    <xs:complexType name="T_Dyes">
        <xs:sequence>
            <xs:element ref="Dye" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="T_User">
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="Role" use="required" type="xs:string"/>
        <xs:attribute name="Email" use="required" type="xs:string"/>
        <xs:attribute name="LastName" use="required" type="xs:string"/>
        <xs:attribute name="Password" use="required" type="xs:string"/>
        <xs:attribute name="FirstName" use="required" type="xs:string"/>
        <xs:attribute name="Description" use="required" type="xs:string"/>
        <xs:attribute name="PhoneNumber" use="required" type="AT_5"/>
    </xs:complexType>
    <xs:complexType name="T_DependOn">
        <xs:attribute name="BindingKit" use="required" type="xs:int"/>
        <xs:attribute name="SequencingKit" use="required" type="xs:int"/>
    </xs:complexType>
    <xs:complexType name="T_Protocol">
        <xs:choice minOccurs="0">
            <xs:element ref="CompatibleReagentKit" maxOccurs="unbounded"/>
            <xs:element ref="ProtocolParameter"/>
        </xs:choice>
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
        <xs:attribute name="InternalOnly" use="required" type="xs:boolean"/>
        <xs:attribute name="ProtocolType" use="required" type="xs:string"/>
        <xs:attribute name="PythonModule" use="required" type="xs:string"/>
        <xs:attribute name="hasPreRunTasks" use="required" type="xs:boolean"/>
        <xs:attribute name="hasPostRunTasks" use="required" type="xs:boolean"/>
        <xs:attribute name="compatibleBindingKits" type="xs:string"/>
        <xs:attribute name="compatibleChipLayouts" type="xs:string"/>
        <xs:attribute name="compatibleSamplePrepKits" type="xs:string"/>
    </xs:complexType>
    <xs:complexType name="T_PlateType">
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="NumOfWells" use="required" type="xs:short"/>
        <xs:attribute name="Description" use="required" type="xs:string"/>
        <xs:attribute name="MaxWellVolume" use="required" type="xs:decimal"/>
        <xs:attribute name="DeadWellVolume" use="required" type="xs:decimal"/>
    </xs:complexType>
    <xs:complexType name="T_Protocols">
        <xs:sequence>
            <xs:element ref="Protocol" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="T_PlateTypes">
        <xs:sequence>
            <xs:element ref="PlateType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

<xs:complexType name="T_DefaultUsers">
  <xs:sequence>
    <xs:element ref="User" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_HiveMetadata">
  <xs:attribute name="HiveTableList" use="required" type="xs:string"/>
  <xs:attribute name="MetadataVersion" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolType">
  <xs:attribute name="Name" use="required" type="xs:string"/>
  <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
  <xs:attribute name="Description" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ReminderTask">
  <xs:attribute name="Name" use="required" type="xs:string"/>
  <xs:attribute name="Interval" use="required" type="xs:string"/>
  <xs:attribute name="StartTime" use="required" type="xs:string"/>
  <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
  <xs:attribute name="Description" use="required" type="xs:string"/>
  <xs:attribute name="DelegateName" use="required" type="xs:anyURI"/>
  <xs:attribute name="IsVisibleInUI" use="required" type="xs:boolean"/>
  <xs:attribute name="ReminderOffset" use="required" type="AT_11"/>
  <xs:attribute name="IsPreventsRunStart" use="required" type=
↪ "xs:boolean"/>
</xs:complexType>
<xs:complexType name="T_ProtocolTypes">
  <xs:sequence>
    <xs:element ref="ProtocolType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_ReminderTasks">
  <xs:sequence>
    <xs:element ref="ReminderTask" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_DependencyItem">
  <xs:sequence>
    <xs:element ref="DependOn"/>
  </xs:sequence>
  <xs:attribute name="ConfigFileName" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ReagentKitType">
  <xs:attribute name="Name" use="required" type="xs:string"/>
  <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
  <xs:attribute name="Description" use="required" type="xs:string"/>
  <xs:attribute name="InternalOnly" use="required" type="xs:boolean"/>
</xs:complexType>
<xs:complexType name="T_ReagentKitTypes">
  <xs:sequence>
    <xs:element ref="ReagentKitType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_InstDBSeedingData">
  <xs:sequence>
    <xs:element ref="DefaultUsers"/>
    <xs:element ref="ProtocolTypes"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:element ref="ProtocolTypeParameters"/>
        <xs:element ref="PlateTypes"/>
        <xs:element ref="ReagentKitTypes"/>
        <xs:element ref="Protocols"/>
        <xs:element ref="Dyes"/>
        <xs:element ref="ProductPartNumbers"/>
        <xs:element ref="PipelineDependency"/>
        <xs:element ref="ReminderTasks"/>
        <xs:element ref="HiveMetadata"/>
    </xs:sequence>
    <xs:attribute name="Version" use="required" type="xs:decimal"/>
    <xs:attribute name="Description" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProductPartNumber">
    <xs:attribute name="Name" use="required" type="xs:string"/>
    <xs:attribute name="DyeNames" type="xs:string"/>
    <xs:attribute name="ChipLayout" type="xs:string"/>
    <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
    <xs:attribute name="PartNumber" use="required" type="xs:int"/>
    <xs:attribute name="ControlName" type="xs:string"/>
    <xs:attribute name="Description" use="required" type="xs:string"/>
    <xs:attribute name="ProductName" use="required" type="xs:string"/>
    <xs:attribute name="PolymeraseSpeed" type="xs:decimal"/>
    <xs:attribute name="PolymeraseKinetic" type="xs:short"/>
    <xs:attribute name="ControlDisplayName" type="xs:string"/>
    <xs:attribute name="LeftPrimerSequence" type="xs:string"/>
    <xs:attribute name="LeftAdaptorSequence" type="xs:string"/>
    <xs:attribute name="RightPrimerSequence" type="xs:string"/>
    <xs:attribute name="RightAdaptorSequence" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolParameter">
    <xs:attribute name="Name" use="required" type="xs:string"/>
    <xs:attribute name="Value" use="required" type="xs:string"/>
    <xs:attribute name="ValueType" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_PipelineDependency">
    <xs:sequence>
        <xs:element ref="DependencyItem" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="T_ProductPartNumbers">
    <xs:sequence>
        <xs:element ref="ProductPartNumber" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="T_CompatibleReagentKit">
    <xs:attribute name="PPN" use="required" type="xs:int"/>
    <xs:attribute name="MixingProtocol" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolTypeParameter">
    <xs:attribute name="Name" use="required" type="xs:string"/>
    <xs:attribute name="MethodName" use="required" type="xs:string"/>
    <xs:attribute name="PythonModule" use="required" type="xs:string"/>
    <xs:attribute name="ProtocolTypes" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolTypeParameters">
    <xs:sequence>

```

(continues on next page)

(continued from previous page)

```
<xs:element ref="ProtocolTypeParameter" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:simpleType name="AT_5">
  <xs:restriction base="xs:string">
    <xs:enumeration value=""/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="AT_11">
  <xs:restriction base="xs:byte">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="AT_37">
  <xs:restriction base="xs:string">
    <xs:enumeration value="A"/>
    <xs:enumeration value="C"/>
    <xs:enumeration value="G"/>
    <xs:enumeration value="T"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```