Sample Tracking
Functional Specification

Table of Contents

The current process

Overview of CTM

The existing sample tracking system is based around a program called CTM (Closure Task Manager). The CTMs core concept is the 'CTM Reference', it is defined in the CTM documentation as:

...a unique work unit of a genome to which tasks can be assigned. A CTM reference is defined by its type, value, and status. Each reference is assigned a unique reference id. Commonly used references include BAC, Sample, Chromosome, or the whole genome.

For the influenza project the *CTM Reference Type* is sample and the *CTM Reference Value* is the Extent ID of the sample. The *CTM References* (henceforth referred to as *samples*) are grouped by assigning them '*Reference Statuses*' (henceforth referred to as *statuses*). The *statuses* are also referred to as queues or bins. However, in the CTM s*tatuses* behave more like bins than queues. Like both bins and queues; a *sample* can only be 'in' one *status* at a time. Unlike queues though; no ordering exists between the *samples* when multiple *samples* share the same s*tatus*.

When the s*tatus* of a *sample* is first assigned, or when it is changed, the new association is recorded in the *sample's CTM Reference History*. A *sample's CTM Reference History* is a list of when the *sample* changed *status* and what the *status* it changed to.

A log of comments is also collected and stored as a *CTM Reference Log* for the *sample*.

Tasks are defined in the CTM manual as:

...any unique task that is tracked by CTM. A task must be related to a defined CTM reference, and can only be created and updated by an authorized CTM user. A task is defined by its type, ***description****,* ***priority****, and status. A CTM task can be* ***assigned / reassigned*** *to any CTM user.*

Tasks, unlike *samples*, can be assigned to a user. They also have priorities and normally exist with multiple types while *CTM References* are only of type *sample*, in the influenza system. Their *Statuses* are similar to a *sample's status* and can be used in the same way. However, there is no equivalent to *CTM Reference History* for *CTM Tasks.*

As with *CTM Reference's CTM Reference Log, Tasks* have a log of the comments made about the task. This log is stored as a *CTM Task Log* for the *CTM Task*.

The CTMs GUI features a sub-set of *Reference Statuses* listed down the left side column, with a drop down menu of all the *CTM References* associated with that status. An administrator interface is provided to select which statuses to display. It is quite possibly this visual prominence that has lead to *CTM References* being used in preference to *CTM Tasks*. The absence of user assignment when using *CTM References* and *Reference Statuses* is worked around by creating statuses that are a combination

of a user and an activity, e.g. 'Edit TT'.

Samples are added to the *CTM* via an import function that requires a BAC ID or Extent ID as well as the type of *CTM Reference* to be created and the *CTM Reference Status* to associate with the newly created reference. Multiple samples can be imported using either ranges for the BAC/Extent ID or comma separated lists.

The CTM contains several built in reports, they do not support the use of compound Reference Statuses though and so aren't used. Currently direct querying of the underlying database is the only way to generate statistics such as; the number of samples, from each collection, in each CTM Queue.

ctmbacs

ctmbacs is a program used for making bulk changes to the CTM and qurying/reporting of the number of samples by status. The samples to modify are selected using either a list of BAC IDs ('-' or ',' as separator) or a file of BAC IDs (separator unknown)

When carrying out a bulk update a comment can be specified on the command line or in a file (referenced from the command line). The statuses of all the selected samples can be set using either the CTM status code or the status' name.

Selecting a sample

ctmbacs: A list of BAC IDs on the cmd line or A file of BAC IDs

JIRA: A set of samples can be defined using a search (filter), The samples to update can then be selected from within that set either by selecting all samples and then removing those that should not be moved or by selecting only the samples needed.

Reports

Number of Samples Per State

ctmbacs: Lists each status followed by the number of samples with that status.

JIRA: Report Pie Chart using a saved filter and grouping on status produces a summary total of the samples with each status.

Samples (IDs) in a State

ctmbacs: Gives a list of all the samples that have a given status. For each sample the BAC ID, status code and status name are listed.

JIRA: A filter can be restricted to only selecting samples with a single status. The filter can be saved to reuse later and a list of samples that matched can be produced using any of the samples fields and exported to Excel.

Bulk Update

Updating the state of a set of samples.

ctmbacs: The state of the selected samples can be set to the given state. There are no restrictions on when a sample can be in a particular state and so any sample can be given any state at anytime.

JIRA: The selected samples can have their statuses set, Restrictions in the workflow may make some state transitions impossible.

Adding a Comment

ctmbacs[1]: The comment can be either on the command line or in a file

JIRA: The selected samples can have a comment added.

Overview of JIRA

The JIRA system uses a number of concepts which are similar to the CTM.

CTM	JIRA	Notes
CTM Reference	Issue	One per physical sample
State	Status	In JIRA status doesn't need to be compounded with Task as Tasks can have users assigned to it. The status can be used to select a subset of the samples. The status is synonymous with the workflow's state.
Task	SubTask	Actions applied to the sample. Each sample can have multiple SubTasks. Each SubTask has a Status and a user assigned. Like a JIRA Task a SubTask's status is controlled by a workflow

1 ctmbacs can both add a comment and change the state at the same time

Approximate Workflow

A spreadsheet is provided with the Samples that contains the information needed to identify them and to submit the resulting sequences to Gen-Bank.

The physical samples are expected to arrive at JCVI labeled with their *Blinded Numbers* (<Project ID>_<Collection Code>_<Sample number>), or a shortened version called *Sample* Name(<Collection Code>_<Sample number>). If they arrive labeled differently then the alternate label is entered into the *Lot*'s spreadsheet as *Tube Label*. Aliquots of the samples are made and these are also labeled with the *Blinded number*.

The samples, depending on the virus and the form of sample, may undergo several preliminary extraction, cleaning and amplification stages before they are ready for the sequencing pipeline. In the case of influenza the samples are MRT-PCRed.

When the samples are ready to be added to the system the data from the provided spreadsheet is uploaded into the GLK using a program called FLURP (RURP for non-influenza projects). During the samples insertion into the GLK it is assigned a BAC ID.

CTM: Once the insertions are complete FLURP returns a list of the samples that have been inserted, giving several IDs for each. One of the IDs is the BAC ID which is used to add the samples to the CTM. Within the CTM the newly added samples/References are given the status of 'Received'.

JIRA: With JIRA FLURP will create the Issues at the same time as uploading into the GLK. The final screen of IDs from FLURP will still be returned but the data is not needed to seed JIRA.[E1]

The samples while in this state may undergo further preparation, specific to the sequencing technologies being used. When ready a JIRA ticket is created for the sequencing team. Attached to the ticket are a JLIMS manifest[2] and a work-order. There are separate projects for each technology, SOLEXASEQ, 454-SEQ and SLW (Sanger Lab Work-orders). A ticket is created for each technology used. The samples are then handed over for sequencing.

CTM: At this point the status is changed to 'Sequencing' or 'Re-sequencing'. For Sanger sequencing the number of expected reads is entered into a command line script.

JIRA: JIRA provides more fine grained monitoring of the work. sub-tasks are created based on the sample type and sequencing type. Examples are; SISPA, RT-PCR and Sequencing. Each of these is initially unassigned and users will assign themselves the tasks. A search can be carried out of the unassigned sub-tasks to ensure that nothing is missed. Each task is only assigned to one person and so avoids duplication of effort.

E1 **Enhancement 1 - Create JIRA tasks when data in uploaded in FLURP**

2 Speak to Dan about what the script does

Sequencing is a form of sub-task. Once assigned it is the assignees responsibility to ensure that the sample is submitted to sequencing and that once sequencing is complete the sub-task is moved to the completed step. Moving the sub-task to complete will send an email to the Sample's assignee to inform them that the sub-task has finished.[E2]

Once the sequencing data is returned the samples can enter the 'Assembly' state. The assembly process varies depending on the technologies used to generate the data.

CTM: If the data is from purely Sanger sequencing then VAPOR is used. Each night the number of expected reads is compared to the number of reads received back for each of the waiting samples. Any that have at least the expected number of reads are then automatically assembled. Once assembly has completed the sample's CTM status is automatically changed to 'Assembly'. VAPOR is also responsible for calling autoTasker on the sample's newly assembled contig(s).

JIRA: When sequencing is complete the sub-tasks are updated, by their owners, to indicate that sequencing has completed. If only Sanger Sequencing was used then the assembler is called automatically in much the same way that VAPOR currently does. When assembly is automatically carried out the sample is moved to 'Review Assembly' once the assembler finishes.[E3] [E4]

The processing of 454, illumina and mixed technology data is more manual. Once the sequencing is complete the JIRA ticket is updated, by sequencing. A VHTNGS (Viral High Throughput Next Generation Sequencing) request is then created. The VHTNGS project is used for keeping track of the Viral-IFX groups tasks and bugs in software produced by the group. There currently is not separate type for Assembly requests, likewise there is little consistency in what information the requests contain. The closing of the JIRA ticket indicates that the assembly has completed.

CTM: Once the assembly has been done the sample must be manually moved to the 'Initial Manual Edit' state and autoTasker run.

JIRA: If the assembly was initiated by JIRA and was successful then the sample is automatically moved to the 'Review Assembly' step. Otherwise the sample waits at the assembly step to be manually moved on.

CTM: The results of autoTasker are then used manually to decide what work, if any, is needed before the sample can be submitted. If work is required it is moved first to a state such as 'Initial Manual Edit' from which a particular worker moves it to 'their' queue ,e.g. 'Edit TT', to indicate that they are working on it. If the sample needs further sequencing it can be moved to a particular user's closure status, 'Closure TT', to carry out the preparation of the sample, such

---

E2 **Enhancement 2 - Send email on workflow transition**

E3 **Enhancement 3 - Run command on workflow transition (VAPOR, autotasker)**
This may also involve communicating with the grid.

E4 **Enhancement 4 - Transition workflow from external program (VAPOR)**

as selection/design of primers. Once ready the sample procedure as occurred for Sequencing is repeated only with the status set to 'Resequencing'.

JIRA: Starting the 'Review Assembly' step increments the 'Assembly Versions' field and calls autotasker. The results from autotasker are summarized into the 'Number of Tasks' field. The number in the 'Number of Tasks' field is the number of closure reactions needed, a score of 0 does not mean that the sample is ready for validation. The sample waits at the 'Review Assembly' Step to be moved (to Lab Processing, Annotate, Close Sample or unresolved). [E3] [E5] [E6]

If the sample requires work it is moved to the Closure or Laboratory Work steps where sub-tasks are created to represent the work. Closure has different sub-tasks than Laboratory work but they are used in the same way.

Once the sub-tasks are complete the sample returns to the 'Review Assembly' step, optionally passing through the 'Assembly' step if required. Passing through the assembly step will increment the 'Assembly Versions' field. The sample will then remain in the 'Review Assembly' step for manual assigning. [E5]

Samples that are not complete, but that have at least one closed segment may be submitted as 'Draft'.

CTM: While in the 'Validation' state or any of the closure states a sample may be marked as Draft by having its state set to Draft Submission and a field in the GLK (is_draft) set to true. Note: At the moment is_draft is only actually set after the sample has been submitted.

JIRA: The 'Submission Type' field can be set at any point after assembly to indicate Draft status. The GLK should be updated, setting 'is_draft' to true. [E7]

Once the sample is ready (e.g. no further tasks or edits are required) the sample is moved into the 'Validation' state. AutoTasker is then used to generate the FASTA files which are manually emailed to the collaborator.

CTM: All samples which haven't been moved to the deprecated or unresolved states are moved to the 'Validation' state once their closure tasks have been completed. AutoTasker is manually run for them and if no errors were found they are moved to the 'Submission' state and the collaborator is emailed.

JIRA: Samples that require no further closure are moved from the 'Review Assembly' step to the 'Validate' step via 'Annotate' and 'Review Annotation' (for influenza both of the annotation steps don't actually do anything, other than move the issue to the next step).

In the 'Validate' step autotasker is used to run 'flu validator' which gives details of quality

E5 **Enhancement 5 - Increment field**
E6 **Enhancement 6 - Set field based on the contents of a file or the output of a command**
E7 **Enhancement 7- Set GLK field (is_draft, deprecated, batch)**

exceptions (required for Gen-Bank submissions). The issues wait at this step until the output has been reviewed.

Once the 'Validate' step is complete samples are moved to 'Collaborator Review'. All samples in a batch must reach this step before the collaborator should be emailed.[E8]

Emailing the results to the collaborator begins the waiting period, if one is needed. This is the length of time from the collaborator receiving the data until the data is submitted to Gen-Bank.

The Gen-Bank submission is generated by autotasker and then manually sent to Gen-Bank for approval.

CTM: After submitting the samples status is changed to 'Published'

JIRA: Once the collaborator has been emailed and any waiting period has passed the issue is moved to the 'Deliver Data' step. In this step the Gen-Bank submission is generated and sent to NCBI. The final move, to 'Sample Published' is carried out once confirmation of the samples acceptance has been received.

At any time in this process a sample could be classified as 'deprecated' if it is discovered that the sample should be ignored. This could occur of many reasons, examples are; incorrect sample sent, sample was actually a water control or the sample is a duplicate of another sample.

JIRA: Deprecated samples should have a property added to their GLK data, 'deprecated' to indicate to other tools that the sample should be ignored. This property needs to be unset if the issue is no longer deprecated. [E7]

E8 **Enhancement 8 - block transition until all the samples in a batch have reached 'Validate'**

JIRA Issues / Tickets

The system will only create one new Issue Type, Sample.

Associated Data for Sample (JIRA Issue)

Field Types[3]

Immutable fields; are assigned a value when the Issue is created and then cannot be changed.

Text fields; allow values up to 255 characters long.

GLK Fields; Fields display the value of a GLK attribute associated with the sample. These fields are not editable within JIRA. Any changes must be made through the GLK. JIRA caches its data and so it could take up to 1 min from altering the GLK before the change is visible in JIRA. Viewed Issues are only updated when the page is reloaded.

Automatic fields;are created and updated by the system and cannot be changed by users.[E9]

Stored in existing JIRA Fields

Field	Type	Values
Summary[4]	Immutable Text Field Searched by quick search **Required**	**Example:** giv3_nhrca_36020
Priority	Select List **Required**	**Values:** Blocker, (Unused) Critical, Major, Minor, Trivial, (Unused) Defer (Unused)
Due Date	Date	
Assignee	User Select List **Required**	**Value:** Unassigned
Reporter	Multi-User Select List **Required**	**Note:** The current user is automatically provided as the default.
Description	Text (over 255 characters)	

3 All of the field types should be search-able and capable of being used to generate of charts.

E9 **Enhancement 9 - Fields with values from the GLK (including view, search and chart)**

4 *Summary* and *Sample ID* are identical. They are a compound of other fields in the format:
<Database>_<Collection Code>_<BAC ID>
Summary and Sample ID are both immutable, as are their constituent fields.

	Searched by quick search	
CC Users	Multi-User Select List	**Note:** JIRA users can add themselves, or be added, to this list to receive updates when this Issue changes.
Attachment	File Selector	

Added Fields

Field	Type	Values
Database	Immutable Text Field	**Example:** giv
Collection Code	Immutable Text Field	**GLK Field:** Extent.ref_id (but not simple to use)[5] **Example:** nhrca
BAC Id	Immutable Text Field	**Example:** 36020
Sample Id	Immutable Text Field	**Example:** giv3_nhrca_36020
Blinded Number	GLK Text Field	**GLK Field:** Blinded Number **Example:** NIGSP_CEIRS_SJC001_JBC_00300
Computed Subtype	GLK Text Field	**GLK Field:** Subtype **Example:** H9N6
Submission Type	GLK Radio button	**GLK Field:** Is Draft **Values:** None, Complete, Draft (Missing/Incomplete Segments)[6], Unknown
Bin[7]	Radio button	**Values:** None, Complete, Needs Edits, Needs Tasks, Failed Sequencing
Assembly Version[8]	Automatic Number	**Example:** 1
Number of Tasks[9]	Automatic Text Field	**Example:** 10,2,1

5 Select Extent_Type_id from Extent where Extent_id in (Select parent_id from Extent where Extent_id in (Select parent_id from Extent where ref_id = '<BAC ID>'));
Set on import, from the Sample Id.

6 The GLK filed 'is_draft' should be set or removed to match this field. [E7]

7 Actual values to be finalized

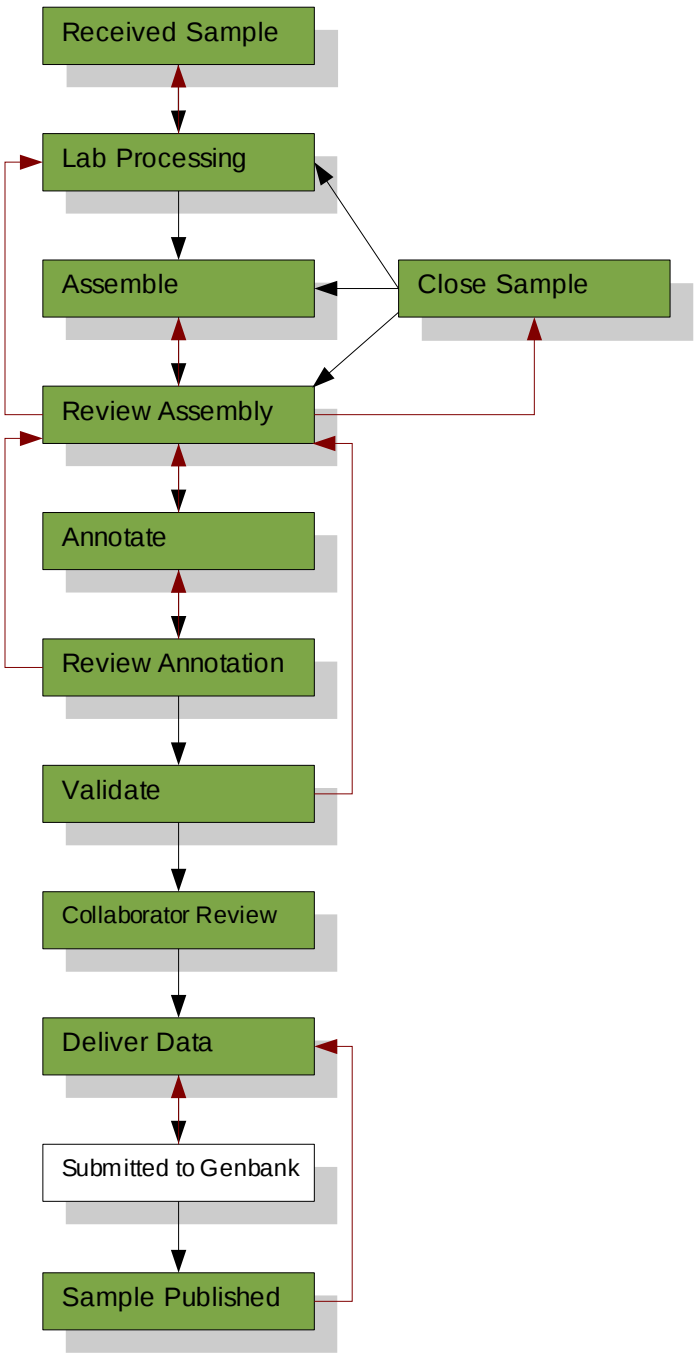
8 Automatically incremented when the sample moves to *assembly review* from *assembly* [E5]

9 A comma separated list of the total jobs remaining each time autotasker is run [E6]

Workflow - Sample (JIRA Issue)

The workflow consists of a number of steps, which are tightly linked to states. These steps are linked by transitions.

Diagram of all the workflow steps and their interconnections



Each step can have multiple transitions entering and leaving it. It is even possible for a state to 'loop back' to itself. Transitions can have actions associated with them, such as setting a field or calling a program.

The steps and their transitions are described in the following format:

Step Name

Summary

Overview of why a sample would be in this state

Actions

- What needs to be done.
- Automatically done actions in bold.

Transitions (in/out):

- **The most common path in bold**

A description of the circumstances under which this transition would be used / would have been used.

- Paths in red are used only if something has gone wrong

Two steps, in the sample tracking workflow, are linked to every other step. These two steps, Sample Deprecated and Unresolved are defined first.

'Sample Deprecated' and 'Unresolved'

GLK Attribute: deprecated (Sample Deprecated)E9

Summary

Sample Deprecated[10]

Samples that are left at this step are ignored when producing statistics. They are moved to this state if there is a problem with the sample that means it should not be used to measure throughput or success. Examples are: duplicate samples, empty samples, externally contaminated samples.

An attribute needs to be added to the GLK record [E9] to indicate to other programs that this issue should be ignored. Inside JIRA there will need to be a filter to help construct charts that do not include deprecated issues.,

10 Will other programs be setting Deprecated?

Unresolved

Samples that cannot be submitted are left at this step. The samples should be suitable for inclusion in statistics, such as throughput and success rate, otherwise they should be moved to the *sample deprecated* step.

Examples are: a sample that contains no complete segments.

Transitions

- To and from all steps

Samples may need to be moved to the *Sample Deprecated* or *Unresolved* steps from anywhere in the workflow. This is because problems that require moving the sample to either of these steps could occur, or be discovered, at any point. Transitions are required from every step to the *Sample Deprecated* and *Unresolved* steps. Return transitions are also needed to return any incorrectly moved sample back to its previous step.

Being connected to every step allows the *Sample Deprecated* step or the *Unresolved* step to act as a bridge between any two steps, avoiding intermediate steps.

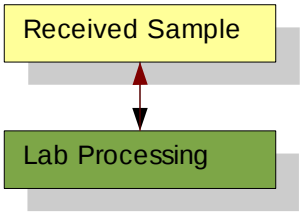
To avoid loosing samples the transitions to these steps will only be available for a limited set of users. A new user group, viral-administrator will be created to manage who can use the transitions. Initially this group will only contain Becky and Nadia.

Actions

Event	Condition	Action
On Entry	*Step = Unresolved or* Step = *Deprecated*	Present a comments field[E10]
		Remove sample from batch (GLK & JIRA) [E7]
		Email Becky [E2]
	Step = *Unresolved*	Set JIRA Resolution to '*Failed*'
	Step = *Deprecated*	Set JIRA Resolution to '*Deprecated*'
		Add GLK field 'Deprecated' to the sample
On Exit	Step = *Unresolved* or Step = *Deprecated*	Set JIRA Resolution to none
		In a later version, return the sample to its batch.
	Step = *Deprecated*	Remove field '*Deprecated*' to sample in GLK

E10 **Workflow transition screen with required comment field**

Received Sample



Summary

This is the first step in the workflow, all samples are initially on this step.

States

State[11]	Action required
The sample is not yet ready for *lab processing* or the lab is not yet ready for the sample.	No action required
The sample and lab are ready	Move the sample to the next step.

Actions

Event	Condition	Action
Before Entering		Flurp or Rurp will normally create the 'issue' when the spreadsheet is uploaded and the GLK populated.

Transitions (in)

- **Issue (Sample) Creation**
  Manual creation can take place either via JIRA or using a bulk issue creation tool that takes a CSV file as input. With the following format for the CSV file:
  `<DB>, <Collection Code>, <BAC ID> [,<Extent ID>]`[12]
- From Lab Processing
  Samples are only anticipated to return from *Lab Processing* to the *Received Sample* step if they have been accidentally moved into *Lab Processing* too soon.
- From Sample Deprecated or
  From Unresolved
  This path provides a means of moving a sample from anywhere in the workflow back to the *Received Sample* step. This would be used if for some reason the sample needed to start again.

Transitions (out):

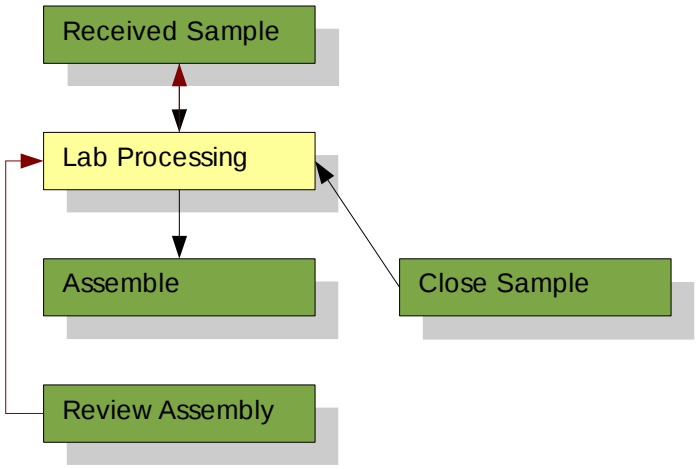
- **To Lab Processing**

11 The state can be thought of as the reason the sample is waiting at this step
12 Optional

When a set/batch of samples are going to be processed they will be moved as a group from the *Received Sample* step to the *Lab Processing* Step where work, in the form of sub-tasks, can be assigned.

- To Sample Deprecated
- To Unresolved

Lab Processing



Summary

Samples, during the *Lab Processing* step, are prepared for sequencing and then sequenced. The work carried out is represented and monitored through the creation of sub-tasks. The Sample will only be able to move to *Assemble* once all of its sub-tasks have Completed (either successfully or have failed).

States

State	Action required
Sample has no sub-tasks	Create sub-tasks[13]
Not all of the sample's sub-tasks have completed	None
All of the sub-tasks have completed or failed	Select further sub-tasks or move to the *Assemble* step.[14]
All of the sub-tasks have completed	Proceed to the *Assemble* step.

13 If automated all the sub-tasks need to be created before the first task completes to avoid accidentally moving into assembly.
14 Do we need an email to the sample's assignee if all this happens (probably not)?

Actions

Event	Condition	Action
On Exit		Block if there are sub-tasks that have not yet completed.

Sub-tasks

- SISPA
- Library Construction
- Nextera
- RPL-PCR
- PCR
- RTPCR
- Sequencing

The sub-tasks used depend on several parameters including the project and which sequencing technologies are used; Sanger, 454 and/or Illumina. Initially the creation of the sub-tasks will be manual to make this variation simpler. Any combination of the above sub-tasks can be created, including having multiple instances of the same sub-type. The features of sub-tasks are covered in the section called '35' on page Error: Reference source not found.
Example uses of sub-task in this step:

- For Sanger Sequencing: An *RT_PCR* sub-task and a Sequencing sub-task with the 'Sequencer Type' field set to Sanger
- For 454: A *SISPA* sub-task and a sequencing sub-task with the 'Sequencer Type' field set to 454
- For combined 454 and Illumina: A *SISPA* sub-task, two *Library Construction* sub-tasks and two *Sequencing* sub-tasks will be created. The 'Sequencer Type' for one of the *Sequencing* /*Library Construction* sub-tasks will be set to 454 and the other will be set to Illumina.

Transitions (in)

- **From Received Sample**

  When a sample is ready to begin being processed it will move to this step.

- From Review Assembly and
  From Close Sample

  If a problem with the samples preparation or sequencing is discovered or simply further next-gen sequencing is to be done on the sample it returns to this step.

- From Sample Deprecated or
  From Unresolved

These provide a path to return to this step and carry out further sequencing from any other step.

Transitions (out)

- **To Assemble**

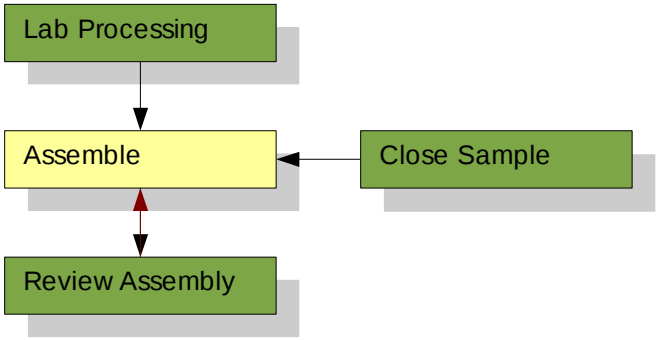
  Restriction: requires all sub-tasks to have finished
  As all sub-tasks have successful completed the sequencing results must be ready for assembly.

- To Received Sample

  If a sample is at the *Lab Processing* step but work should not yet begin on it then the sample should be returned to the *Received Sample* step.

- To Sample Deprecated
- To Unresolved

Assemble



Summary

Samples in this state have completed sequencing and their results are ready and available to be assembled.

Assembly varies depending on the combination of technologies used. If only Sanger Sequencer data is being used then VAPOR is called to carry out the assemblyE3. Once VAPOR completes the sample is moved to the *Review Assembly* step.

The details of assembling other forms and combinations of sequencing data are not stored in the system. Initially all other other assemblers will need to be run manually and the sample's workflow manually moved to *Review Assembly*.

States

State	Action required
Needs assembly starting	Run Assembler
Assembly in progress	None
Assembly complete, needs moving.	Move to the *Review Assembly* step.

Actions

Event	Condition	Action
On Entry	Sanger Sample	Run ? to carryout assembly
Assembler completes	Sanger Sample	Move to the Review Assembly step.
On Exit	to Review Assembly	Increment the assembly version field

Transitions (in)

- **From Lab Processing**

  Once a sample has been sequenced it will move to this step.

- From Close Sample

  When further sequencing data is produced in the *Close Sample* step it will need to be integrated with the assembly.

- From Review Assembly

  If the review of the sample's assembly finds problems with the assembly, which can be fixed, then the sample is returned to this step to carry out a new assembly.

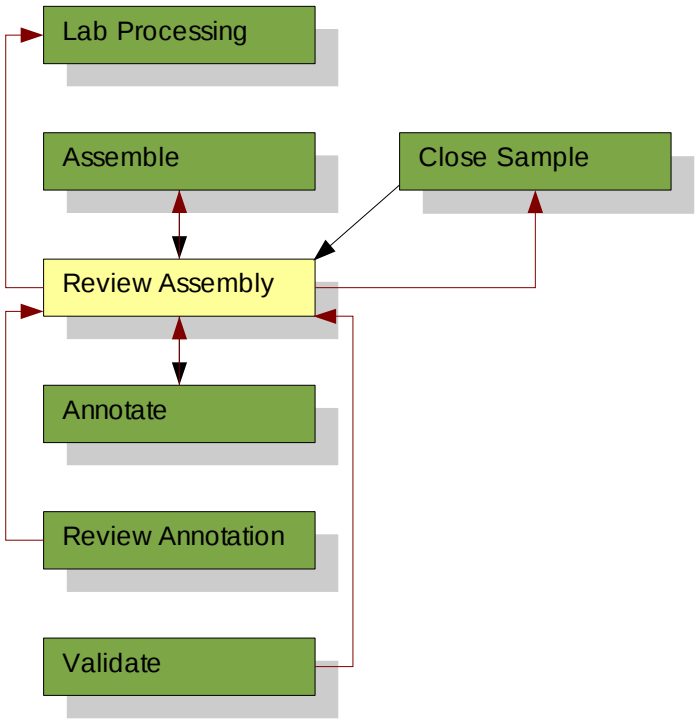
- From Sample Deprecated or
  From Unresolved

  If a fault with the assembly is discovered while in a step that does not have a transition to this step then either the S*ample Deprecated step* or the *Unresolved* step can be used to form a path back to the *Assemble* step.

Transitions (out):

- **To Review Assembly**
  If there is no error then a newly created assembly should be reviewed.
- To Sample Deprecated
- To Unresolved

Review Assembly



Summary

Samples waiting at this step require human review.

The review of an assembly occurs in two phases. The first phase is carried out as soon as the sample reaches this step. Autotasker 2 is run, to check for various problems and to generate a report of its findings. The testing includes running Flu-Validator. The full report is stored in a file while the number of remaining tasks (reactions[15]) is appended to the *number of tasks* field which contains a comma separated list of the values from each time the sample has entered *Review Assembly*. The number of remaining tasks should not include tasks for segments that will not be submitted.

The second phase is selecting the next step. Initially this will be a totally manual process. In a future enhancement autotasker will be used to set the bin field on the sample. The exact criteria by which the value will be selected still needs to be resolved. The specific 'bins' may also change.

States (Initially)

State	Action required

15 'number of tasks' does not include edits. It is only the number of closure reactions required.

Requires human review	Review autotasker output, select next step. (See Summary)

States (Auto-binning)

The bins are currently defined as:

- 'Complete':
  - The sample requires 0 tasks AND
  - The sample requires 0 edits
- 'Needs Edits':
  - The sample requires 0 tasks AND
  - The sample requires 1 or more edits
- 'Needs Tasks':
  - The sample requires between 1 and N tasks
- 'Requires Review':[16] One of the following is true.
  - The sample would require excessive work to close (Tasks > N) OR
  - A Processing or Sequencing error is known to have occurred OR
  - Sequencing returned significantly less data than expected.
- 'Segments Missing':
  - The sample has at least one complete segment.

Currently the behaviors for the various values are expected to be:

Bin	Reviewer determined type	Sample's next step	Sample assigned to
Complete	-	*Annotate*	-
Needs Edits	-	*Close Sample*	Nadia
Needs Tasks	-		
Segments Missing	Sequencing error	*Lab Processing*	
	Draft	*Annotate*	
	Draft or Important	*Close Sample*	
Requires Review	Normal	*Unresolved*	Becky
	Important	*Close Sample*	

16 Merged with 'Failed Sequencing' as assigning this value for bin based on the number of tasks is much simpler than deciding if there was a sequencing problem. The human reviewing the sample can decide what has caused the errors.

	Failed Sequencing	*Lab Processing*	
	Draft	*Annotate*	

Actions

Event	Condition	Action
On Entry		Run autotasker and append the number of tasks required to the comma separated list, *Number of tasks*.
On Entry	Auto-binning	Run autotasker and use the 'bin' selected to automatically move samples to their next state, where possible. Segments Missing and Requires Review cannot be automatically forwarded as there are multiple steps that they could proceed to.

Transitions (in)

- **From Assemble**

  Once assembled the samples information needs to be reviewed.

- From Close Sample

  If closure only involved editing the assembly and no new sequencing data was produced then the sample can be re-evaluated without passing through the *Assemble* step.

- From Annotate

  If generating annotation revealed a problem the sample is returned to this step to evaluate what the next step should be.

- From Review Annotation
  If reviewing the annotation shows that more work is needed.

- From Validate

  If the validation process finds further work is needed then the sample is returned to this step to be re-evaluated.

- From Sample Deprecated or

  From Unresolved

  If a sample's existing assembly needs to be reviewed, from a step other than those listed above it can reach this step via the *Sample Deprecated* or *Unresolved* steps.

Transitions (out)

- **To Annotate**
  - Bins: 'Complete', 'Segments Missing - Draft' and 'Requires Review - Draft'

If the sample is complete, or draft with no closure work remaining, it should be moved to *Annotate*.

All samples, including influenza samples, pass through annotation. Those where annotation is not done in house pass through annotation automatically without any work being done. In these cases the *Annotate* and *Review Annotation* steps will automatically forward the sample to the *Validate* step.

- To Close Sample
  - Bins: 'Needs Edits' and 'Needs Tasks'

  If the sample needs work but can be closed without excessive effort it should proceed to the C*lose Sample* step.
  - Bin: 'Requires Review'

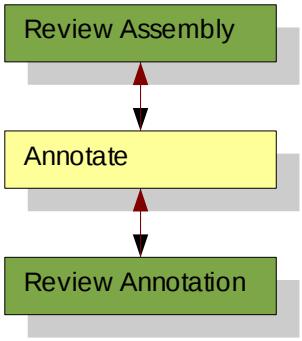
  If the sample requires more work than normal but is still going to be closed then the sample should be moved to the *Close Sample* step.
- To Assemble
  - Bin: No matching bin

  If the sample is to be re-assembled then the next step should be the *assembly* step. Re-assembly could be needed if the original assembly was made before all of the sequencing data had been returned or if an alternate assembly protocol is to be tried.
- To Lab Processing
  - Bin: 'Requires Review'

  If a problem occurred during the preparation of the sample or its sequencing then it should be moved back to the *Lab Processing* step.
- To Sample Deprecated
- To Unresolved
  - Bin: 'Requires Review'

  If the sample requires too much work to close then it should be moved to the Unresolved step.

Annotate



Summary

Initially this step will be empty and all samples will simply be forwarded to the *Review Annotation* step which in turn will forward the samples to the *Validate* step.

Actions

Event	Condition	Action
On Entry	Influenza Sample	Forward to the Review Annotation , no annotation is generated.

Transitions (in)

- **From Review Assembly**

  Once the sequence's lab work is complete and it has been assembled.

- From Review Annotation

  If the review reveals errors that require regenerating the annotation to resolve.

- From Sample Deprecated or
  From Unresolved

  If the sample was incorrectly moved to the Sample Deprecated step or the Unresolved step.

Transitions (out)

- **To Review Annotation**

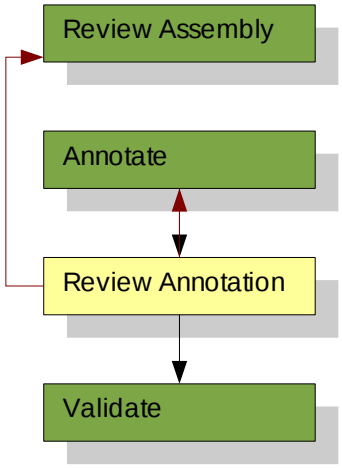
  Once the annotation has been generated it should be reviewed. This transition may be automated at a later date, along with running the annotation pipeline.

- To Review Assembly

  If a problem is found during annotation the sample should be returned to the *Review Assembly* step. From the *Review Assembly* step it can be evaluated and moved to *Close Sample*, *Lab Processing*, *Assemble* or C*ollaborator Review* as needed.

- To Sample Deprecated
- To Unresolved

Review Annotation



Summary

Samples waiting at this step have been annotated and require reviewing. Initially this step will be empty and all samples will simply be forwarded to the *Validate* step.

Action

Event	Condition	Action
On Entry	Influenza Sample	Forward to the *Validate* step

Transitions (From)

- **From Annotate**

  Once the annotation has been generated the sample moves to this step to review it.

- From Sample Deprecated or Unresolved

  If the sample was incorrectly moved to the Sample Deprecated step or the Unresolved step.

Transitions (out)

- **To Validate**

  If the annotation generated is acceptable, or no annotation was required, then the sample moves to the *Validate* step for the submission files to be generated.

- To Annotate

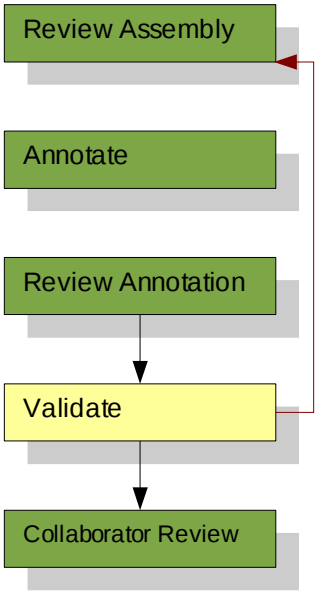
  If the annotation needs to be regenerated.

- To Review Assembly

  If problems with the sequence or assembly are discovered

- To Sample Deprecated
- To Unresolved

Validate



Summary

Samples at this step have been assembled and reviewed. They will belong to one of the following groups:

- Complete and Annotated
- Draft and Annotated
- Complete and Unannotated
- Draft and Unannotated

All samples must be validated, including those marked as draft. This is to generate the list of quality exceptions required for their submission to Gen-Bank. For Sanger sequenced samples the list is generated by FLAVOR. Next-gen sequenced samples do not currently have an equivalent to FLAVOR and should be moved to *Collaborator Review* without carrying out any further tests.

States

Samples waiting at this step require Validate before being moved to the *Collaborator Review* step.

State	Action required
Requires validation	Run FLAVOR, or other validation tool
Next Gen	Move to *Collaborator Review*

Transitions (in)

- **From Review Annotation**

  If the sample is suitable for submission it passes through annotation (although not all samples will have had any annotation generated while passing though the annotation steps) to the Validate step.

- From Sample Deprecated or

  From Unresolved

  If a sample is found to be complete, but it's current step isn't one of those above then it can be moved via the *Sample Deprecated* or *Unresolved* steps to this step to be validated.

Transitions (out)

- **To Collaborator Review**

  Next-Gen samples; are moved to *Collaborator Review* without carrying out any other actions. Sanger samples; are moved to Collaborator Review after FLAVOR has been run, unless they **unexpectedly**[17] fail validation.

- To Review Assembly

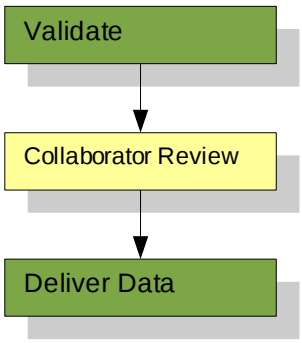
  If the sample unexpectedly fails validation then it should be moved to the *Review Assembly* step.

- To Sample Deprecated

- To Unresolved

17 Draft submissions will 'fail' validation but should still be moved on to *Collaborator Review*

Collaborator Review



Summary

Samples at this step are ready to be sent to the collaborator. They have been validated (if required) and annotated (if required)

States

State		Action required
Collaborator has not been emailed	Part of a batch[18] and not all of the batch has reached this step.	None
	Not part of a batch, or all of batch is ready	Email collaborator
Collaborator has been emailed	'Days to wait' period has not finished	None
	No 'days to wait' period, or the 'days to wait' period has finished	Move to the *Deliver Data* step

Transitions (in)

- **From Validate**

  Once all of the data for the sample has been generated and checked the sample moves to this step.

- From Sample Deprecated or
  From Unresolved

Transitions (out)

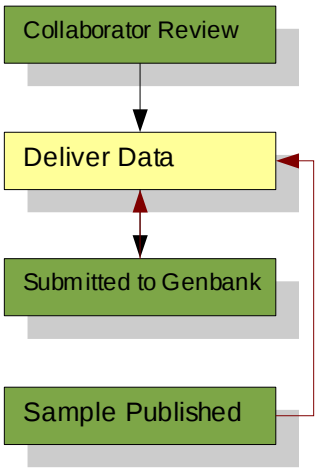
- **To Deliver Data**

  Once the Collaborator has been emailed and any wait period has passed.

- To Sample Deprecated
- To Unresolved

18 Samples that are part of a batch will have a GLK attribute, *batch*, whose value indicates which batch they belong to.

Deliver Data



Summary

Before reaching this step the sample has been submitted to the collaborator and any wait period has passed.

State

Samples waiting at this step require the Gen-Bank file's generation and submission before being moved to the *Submitted to Genbank* step.

Transitions (in)

- **From Collaborator Review**

  Once the collaborator has been emailed and any wait period has passed.

- From Sample Published
  From Submitted to Genbank

  If publishing, to Gen-Bank, needs repeating.

- From Sample Deprecated or
  From Unresolved

  If the sample was incorrectly moved to the *Sample Deprecated* step or the *Unresolved* step.

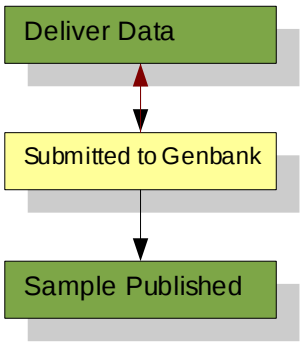
Transitions (out)

- **To Submitted to Genbank**

  When the sample has been submitted to Gen-Bank

- To Sample Deprecated
- To Unresolved

Submitted to Genbank



Summary

Before reaching this step the sample has been submitted to Gen-Bank.

State

Samples at this step have been submitted to Gen-Bank and are waiting for confirmation of their acceptance before moving to *Sample Published*.

Transitions (in)

- **From Collaborator Review**

  Once the collaborator has been emailed and any wait period has passed.

- From Sample Published

  From Submitted to Genbank

  If publishing, to Gen-Bank, needs repeating.

- From Sample Deprecated or

  From Unresolved

  If the sample was incorrectly moved to the *Sample Deprecated* step or the *Unresolved* step.

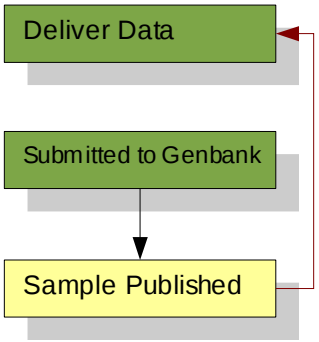
Transitions (out)

- **To Sample Published**

  When the sample has been accepted by Gen-Bank

- To Sample Deprecated
- To Unresolved

Sample Published



Summary

The final step for Samples.

Samples waiting at this step have been successfully processed.

State

This is the final step, the sample has been published and nothing else needs to be done.

Actions

Event	Condition	Action
On Exit		If a sample is being restarted its 'submitted date' needs to be blanked.

Transitions (in)

- **From Submitted to Genbank**

  Once the sample has been accepted by Gen-Bank

- From Sample Deprecated or

  From Unresolved

  If the sample was incorrectly moved to the *Sample Deprecated* step or the *Unresolved* step.

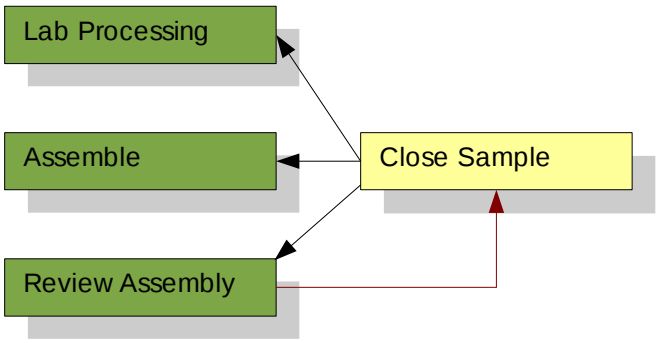
Transitions (out)

- To Deliver Data

  If publishing, to Gen-Bank, needs repeating.

- To Sample Deprecated
- To Unresolved

Close Sample



Summary

While a sample is being worked on, in an attempt to close it, it waits at this step. sub-tasks are used, in the same way as in 'Lab Processing', to represent the work taking place. As with 'Lab Processing' samples may be waiting here for a number of reasons, such as:

- they need to be assigned sub-tasks
- their sub-tasks aren't complete
- they have failed sub-tasks

(Nadia's Notes) Once reactions are finished and scheduled, something should:

- change '*In-house closure*' task status to '*Complete*' E4
- change state from '*Close Sample*' to '*Assemble*' E4
- assign sample task to Nadia[19]

States

State	Action required
Sample has no sub-tasks	Create sub-tasks
Not all of the sample's sub-tasks have completed	None
All of the sample's sub-tasks have completed, but some failed	Select further sub-tasks or Move to the next step
All of the sample's sub-tasks have successfully completed	Move to the next step

See: Transitions (out) for a list of the possible next steps and their uses.

Sub-tasks

- **Closure Editing Task**

  Used when the sample has small errors that can be remedied using the existing data

- **In-House Closure Task**

  Used when the sample has gaps / low coverage regions that can be closed using the existing

19 Do we really want the sample assigning to Nadia when it enters *Assemble* (from closure)?

primers

- **Custom Closure Task**

  Used when the sample has gaps / low coverage regions that require custom primers to close

Transitions (in)

- **From Review Assembly**

  If the output of autotasker indicates that a limited amount of work is needed to close the sample, it is moved to the C*lose Sample* step while that work is done.

- From Sample Deprecated or

  From Unresolved

  If the sample was incorrectly moved to the Sample Deprecated step or the Unresolved step.

Transitions (out)

- **To Assemble**

  All sub-tasks must be in a final state. If at least one of sub-tasks was sequencing then the sample continues to the *Assemble* step.

- **To Review Assembly**

  All sub-tasks must be are complete. If the work has only consisted of edits to the existing assembly then the sample moves directly to the R*eview Assembly* step.

- To Lab Processing

  All sub-tasks must be are complete. If the sample requires further next gen sequencing then the sample moves directly to the *Lab Processing* step.

- To Sample Deprecated
- To Unresolved

sub-tasks

Sub-tasks are being used to represent the physical work done with a sample. Their advantages over workflow steps are:

- multiple can be in progress simultaneously
- multiple instances of a sub-task can co-exist, each with it's own values and state
- the order in which they are created and used does not need defining before they are used
- sub-tasks can be added and removed more readily than workflow steps

The ability to have multiple instances of a sub-type allows for re-use of the sub-task definitions, e.g. only one sequencing sub-task is needed despite there being 3 types of sequencing currently in use. It also allows a sub-task to be repeated without loosing the information from previous uses. For example; a sequencing sub-task can be created on each iteration of closure without over writing the original sequencing sub-tasks information.

All sub-tasks share these common fields (that are built into JIRA):

Field	Type	Values
Summary	Immutable Text Field	Used to identify the instance of the sub-task
Assignee	User Select List	The user responsible for the sub-task.

The Assignee field is used to show who is currently responsible for the sub-task. When the sub-tasks are initially created the Assignee field is set to no-one. This makes it simple to search for sub-tasks that have not been allocated. Sub-tasks can only have one user assigned to them, which avoids work getting repeated.

sub-tasks all have a workflow associated with them, details of the workflows are given after the sub-tasks. Independent of the workflow allocated when a sub-task finishes (reaches the completed or failed step) the owner of the containing issue will be notified.

Lab Processing sub-tasks

All sub-tasks created during this step will have a name (summary field) in the format:

```
   <collection code>_<sample number>_<sub-task Type>
e.g.
BHTEST_00001_SISPA
BHTEST_00002_RPL-PCR
BHTEST_00003_PCR
```

Library Construction

Information stored - Added Fields

Field	Type	Values
Lib Sequencer Type	Immutable Radio Button	None, 454, Illumina
~~454~~Library ID (From 454SEQ or SOLEXASEQ JIRA Ticket)	Text Field	
~~Illumina Library ID~~[20]	~~Text Field~~	

Workflow

Laboratory Workflow

Notes

A separate task will be created for each NG technology.

Nextera

Information stored - Added Fields

Field	Type	Values
Nextera Type	Immutable Radio Button	None, 454, Illumina
Barcode Name [21]	Text Field	Example: BC004CG
Barcode Sequence [21]	Text Field	Example:CGTAGTACACTCTAGAGCACTA
Has Sanger Data [21]	Select List	None,

20 The field only needs to be called Library ID, we can tell if it's 454 / Solexa by the sequencer type.
21 From Nextra Spreadsheet

		Undefined, Yes, No
Pool Name[22] (Manually entered)	Text Field	Examples*:* 20110707, 20110707-2

Workflow

Laboratory Workflow

Notes

A new method of Library Construction for Next-Generation Sequencers.

PCR, RPL-PCR and RT-PCR

Information stored - Added Fields

Field	Type	Values
Primer Plate Name (Initially Manual) (Later From JLims)	Text	Example: T48

Workflow

Laboratory Workflow

Sequencing Task

Custom Fields

Field	Type	Values
Sequencer Type (Set on Creation)	Immutable Radio Button	~~None,~~ Sanger, 454, Illumina

Workflow

Laboratory Workflow

22 Not available until the sample has been SISPAed

SISPA Task

Information stored - Added Fields

Field	Type	Values
Sample Type	Select List	None, Undefined, DNA, RNA
Barcode Name [23]	Text Field	Example: BC004CG
Barcode Sequence [23]	Text Field	Example:CGTAGTACACTCTAGAGCACTA
Has Sanger Data [23]	Select List	None, Undefined, Yes, No
Pool Name[24] (Manually entered)	Text Field	Examples: 20110707, 20110707-2

Workflow

Laboratory Workflow

23 From SISPA Spreadsheet

24 Not available until the sample has been SISPAed

Closure sub-tasks

The sub-tasks will be named in the format:

<project name>_<SISPA pool number>_<sub-task type><sequencing type>

Project name	-	The D*atabase* from the B*linded Number*.
SISPA pool number	-	Generated during the initial sequencing, more than one may apply. If the sample was run over multiple lanes then the lanes are differentiated by -N being added to the end, where N is [0-9]
sub-task type	-	The name of the sub-task
sequencing type	-	Sanger, Illumina, 454, what about mixed?

Closure Editing Task

Information stored - Added Fields

No additional data stored

Workflow

Closure Workflow

Notes

Only one Editing Task is create per pass through Review Assembly. It will contain the information necessary to carry out all of the edits.

Custom Closure Task

Information stored - Added Fields

Field	Type	Values
Amplicon Name [25]	Text	

Workflow

Custom Closure Workflow

Notes

Amplicon Name will not be known when the task is created, it will need to be filled in later.

There could be more than one amplicon name.[EX]

25 The script that creates the closure manifest produces a template name that could possibly be used to identify the amplicon nane.

EX Create a 'list' custom field type

In House Closure Task

Theoretically only one In House Closure Task will be run per-sample. If more are run all that is required is appending a date or number to the sub-task's name to keep it unique.

Information stored - Added Fields

Field	Type	Values
Amplicon Name	Text	

Workflow

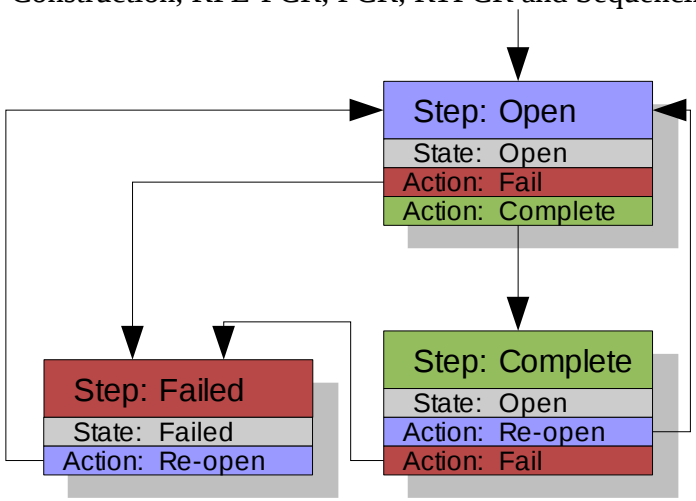
In House Closure Workflow

sub-task Workflows

The workflows for the sub-tasks are nearly all fully linked, each step can be reached from any other step. The descriptions will therefor concentrate on the steps and not the actual transitions.

Laboratory Workflow

Used by: SISPA, Library Construction, RPL-PCR, PCR, RTPCR and Sequencing



Open

This is the initial Step. The sub-task remains at this step until the work has concluded.

Complete

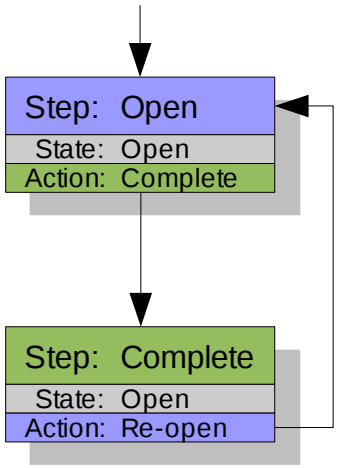
The sub-task moves to this step once all work has concluded successfully

Failed

The sub-task moved to this step once no further work is to be carried out and the process failed.

Closure Editing Workflow

Used by: Closure Editing Task



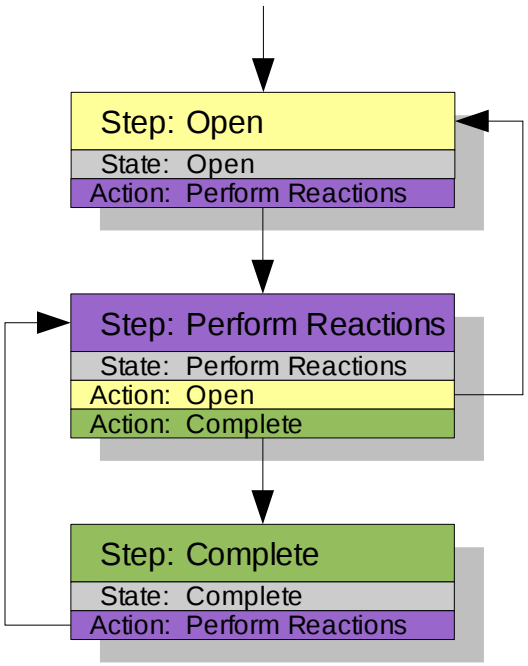
Open

This is the initial Step. The sub-task remains at this step until the work has concluded.

Complete

The sub-task moves to this step once all work has concluded.

In-House Closure Workflow



Used by: In-House Closure Workflow

Open

This is the initial Step. The sub-task remains at this step until the work **begins**, unlike the previous workflows.

Perform Reactions

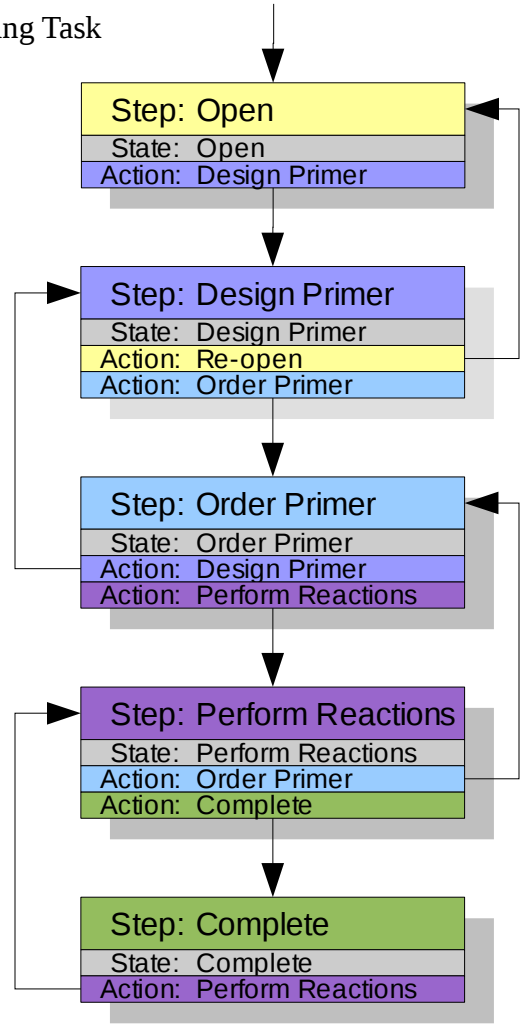
The sub-task enters this step when the lab work begins and remains at this step until the lab work is complete.

Completed

The sub-task moves to this step once all work has concluded

Custom Closure Workflow

Used by: Custom Closure Editing Task



Open

This is the initial Step. The sub-task remains at this step until the work **begins**, unlike the previous workflows.

Design Primer

The sub-task remains at this step until the primer design is complete

Order Primer

When this step is reached the primers are ordered. The sub-task then remains at this step until the primers have been delivered.

Perform Reactions

The sub-task enters this step when the primers arrive. It remains at this step until all of the lab work is done.

Completed

The sub-task moves to this step once all work has concluded

Changes to the Workflow

The workflow has changed in a number of ways from the initial draft. This section exists to document both the changes and the reasons for them.

Removal of path from Assemble to Close Sample

Samples can still move from the *Close Sample* step to the *Assemble* step when there closure tasks were only edits and no new sequencing was carried out.

The A*ssemble* step is intended to be used by samples awaiting assembly, or that are in the process of assembly. No decisions, about the samples path through the workflow, need to be made there.

The *Assemble* step contains samples that are waiting to be assembled.

- Once assembled the sample should be moved to the *Review Assembly* step.
- If assembly fails for a sample then it should still proceed to *Review Assembly.*
- If it is decided to skip assembly, because of some known problem, the sample should still proceed to *Review Assembly* to indicate that manual intervention is needed to decide what the samples next step should be.

Once assembly has finished (completed, failed or aborted) the sample should be moved to the R*eview Assembly* step. If closure tasks are required the sample can be moved to the close Sample step from the R*eview Assembly* step.

Passing through the *Review Assembly* step and before moving to the *Close sample* step makes sense as samples should be reviewed, either manually or automatically, before they enter closure. Closure is also only one of the possible steps a sample could require. The assembler, human or script, can indicate that there is a problem with the sequence without having to decide what should be done about it.

Accessing Close Sample only via Review Assembly

As indicated in the 'Removal of path from Assemble to Close Sample' section it is simpler for a step to determine something is wrong than to decide on the best cause of action. By having the *Close Sample* step only accessible via the *Review Assembly* step only samples that have been reviewed and it has been decided to close will enter closure. Users or programs acting in other steps do not need to decide what should be done if a problem is found with a sample. Equally they cannot assign work to the sample without it passing back to the *Review Assembly* step.

The *Review Assembly* step has paths to handle a wide range of problems with samples:

- If the sample is complete it is automatically moved to the *Annotate* step
- If the sample is going to be submitted as draft it proceeds to the *Annotate s*tep
- If the sample should be re-sequenced it is moved back to the *Lab Processing* step
- If the sample should be re-assembled it is moved back to the *Assemble* step

- If the sample requires closure work it is moved to the ***Close Sample*** step
- If the sample is not going to be submitted but should still be used for statistics then it is moved to the Un*resolved* step
- If the sample should be ignored then it is moved to the D*eprecated* step

Removing the, optional, bypass of Validate

All Sanger sequenced samples should pass through the *Validate* step. At the moment Next-Gen samples do not have an equivalent step. A path from the *Review Assembly / Review Annotation* steps to the *Collaborator Review* step was previously added to allow Next-Gen samples to skip the *Validate* step.

The alternative is instead of bypassing *Validate*, and adding an extra path, to instead pass all samples into the *Validate Step* but only run 'FLAVOR' on the Sanger sequenced ones. This has the advantage that if in the future Next-Gen samples can be validated, possibly by a separate program, no change to the workflow is needed.

Initially it will be a human who decides which samples to run FLAVOR on and which to pass-through. Latter this logic could be automated based on the sub-tasks that have been run on the issue.

From a user's perspective they will only need to use one path for samples that are ready to submit, instead of two (one for Sanger and one for Next-Gen).

Adding the Annotation Steps for all Samples

Originally there was a path, for influenza, that went from the *Review Assembly* step directly to the *Validate* step without passing through annotation. There was also a path from the *Review Assembly* step to the *Collaborator Review* step for influenza draft samples (this was removed along with the bypass of Validate). Rather than have these separate paths for influenza it would be simpler if all viruses used the same path. Instead of a path around the annotation steps the steps can be configured to pass influenza samples through to the *Validate* step without carrying out any actual work.

This reduces the number of paths and hence complexity of the workflow and also simplifies adding and removing annotation for different pipelines.

Redefining Collaborator Review

Previously the *Collaborator Review* step was before the *Validate* step and used for:

- samples that had too many tasks required to close them.
- samples waiting between being sent to the collaborator and being sent to gen bank.

The *Collaborator Review* step has now been moved to after the *Validate* step. It will only contain samples that are ready for submission and are:

- waiting to be sent to the collaborator (waiting for their batch to all be ready)
- waiting between being sent to the collaborator and being sent to gen bank

Samples that have too many tasks required to close them will now be kept at the *Review Assembly* step.

This effectively creates two 'holding areas'.

The *Review Assembly* step which is used for all the samples that have problems; acting as a gateway to closure.

The *Collaborator Review* step which is used for samples that are ready for submission; acting as the gateway to being published.