WASP project design documents

Andrew McLellan (May 17 2011)

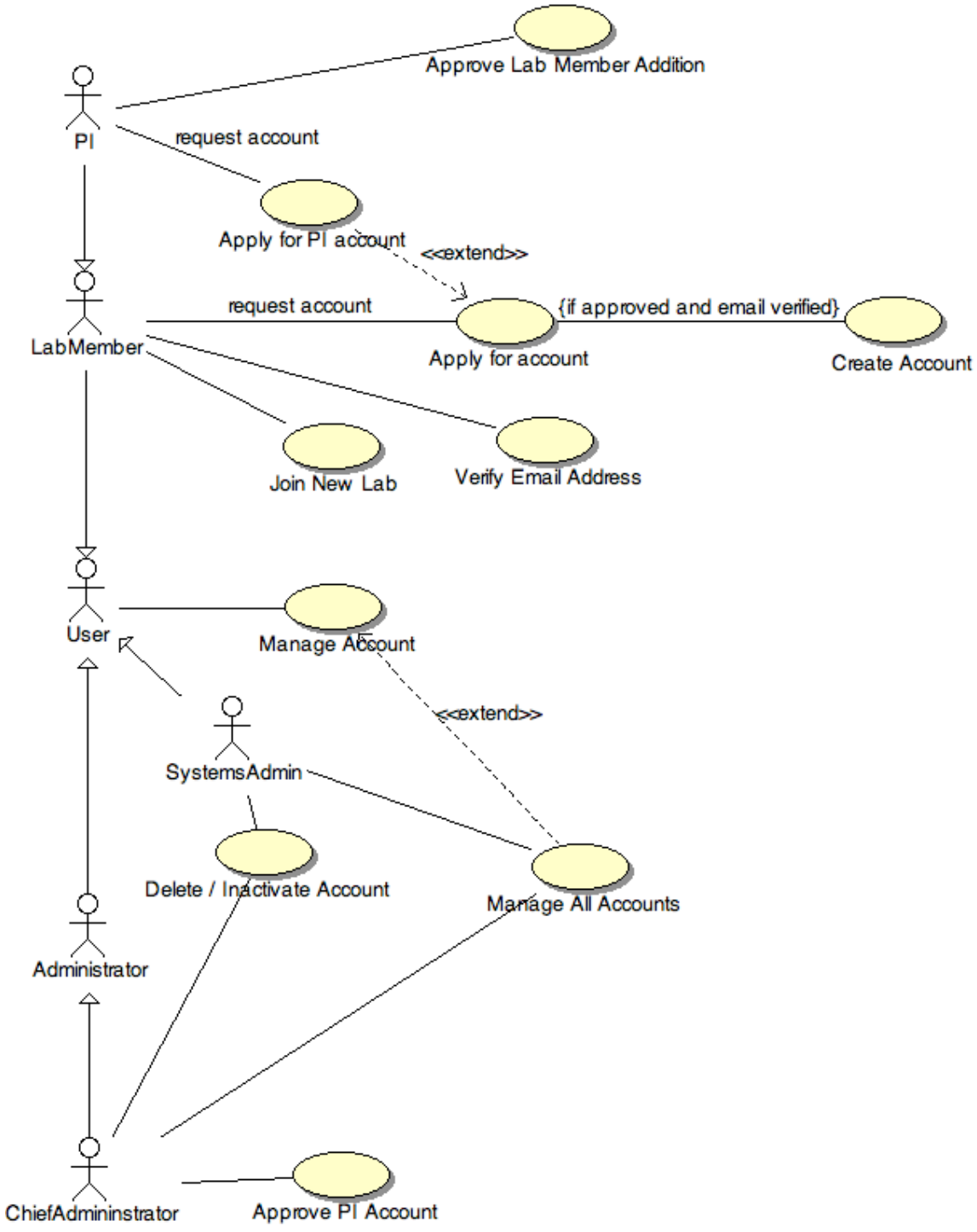
Please note, these are a work in progress, maybe partial / incomplete are subject to change

Actors

Actor	Goal	Brief
Lab Member	Submit jobs	The lab member is the primary user. They fill in sample submission forms, check the progress of submissions then, post sequencing, download data and view analyzed data. They use a wiki to interface with the system and their details are stored in a database. A Lab member can see all the data generated by other members of the same lab.
Principle Investigator (PI)	Validate account applications to their lab and job submissions by members of their lab	The PI is the head of a lab and a lab member themselves. They may be the only lab member but all labs have to have a PI. The PI can regulate who can apply to be a lab member and approve or reject sequencing jobs submitted by lab members.
Administrator	Manage billing and accounts	The Administrator can validate funds available before a job is accepted and may edit / re-send quotes and invoices generated by the system. They may generate accounts statistics.
Chief Administrator	Authorize PI account applications and manage user accounts	The Chief Administrator is able to authorize applications for new PI accounts and CRUD on any accounts in addition to the basic administrator role.
Facility Tech	Update LIMS as samples /	The Facility Tech is a

	libraries are processed	member of core facility staff performing day to day operations. They accept biological material from Lab Users, process libraries, perform QC functions and prepare sequencing runs. They record their activities in the WASP LIMS.
Systems Administrator	Manages configuration of the system and can modify any data in the database	The Systems Administrator is the person responsible for setting up, upgrading and customizing an instance of WASP. They also have database access and can start, stop and restart analysis pipelines.
Facility Manager	Manages the whole WASP system	The Facility manager is at the top of the administrator ladder and inherits the roles of the Chief Administrator and Facility Tech.

Use Case Application for WASP Account

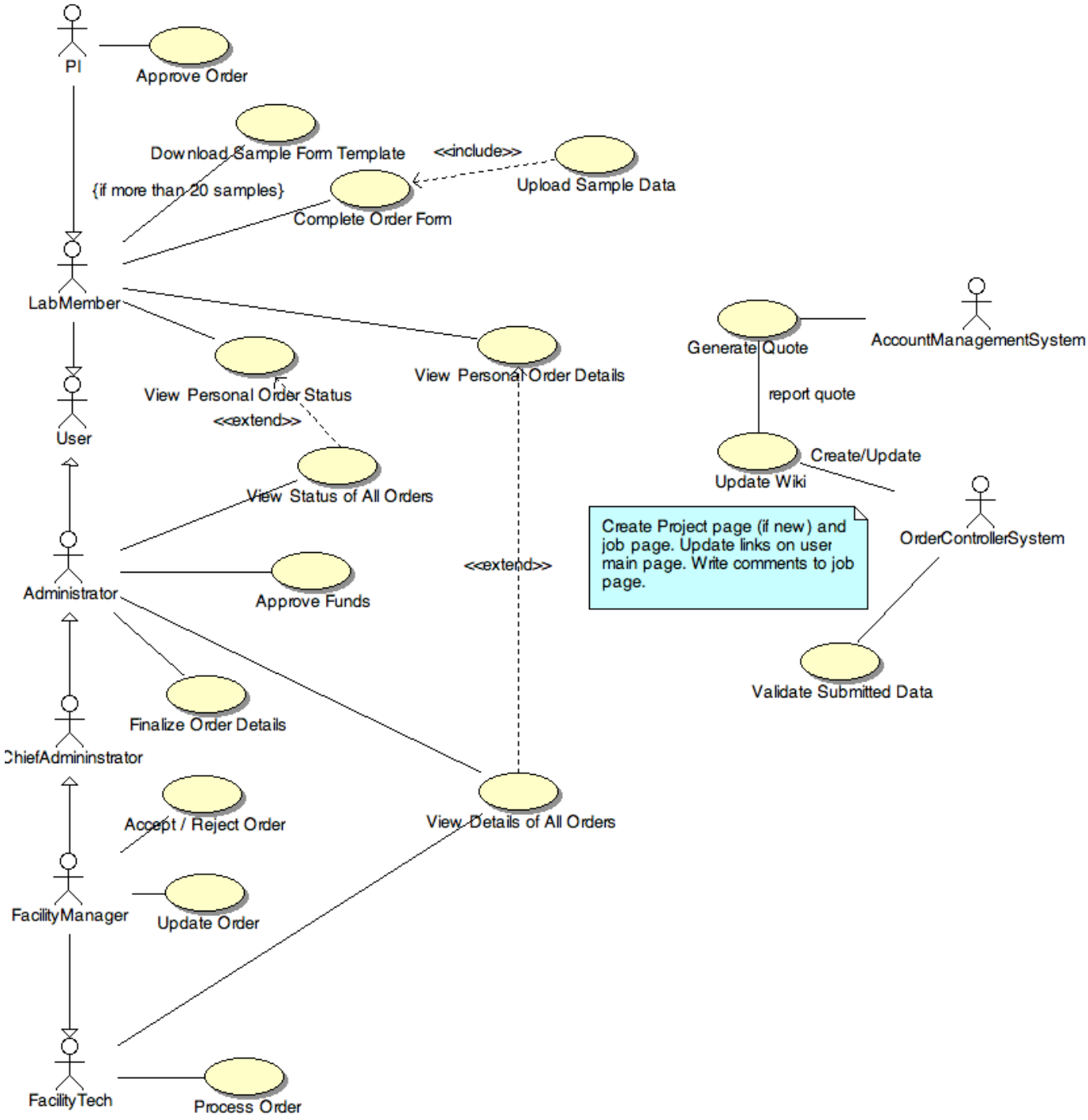


Use Case: Application for a WASP account
Synopsis: A new user wishes to apply for an account. They use a web form embedded into the wiki and visible to anyone. They enter details and specify the email address of their PI which they must know.
Primary Actor: Lab User
Scope: wasp.web
Level: User Goal
Stakeholders and Interests: a Lab Member wishes to submit samples and needs and account. A PI wishes a Lab User to get an account to perform sequencing
Precondition: PI already has an account. Lab member knows PI email address
Minimal Guarantee: Logging will show user attempt to create an account. Validation will ensure correct completion of form.
Success Guarantee: User details entered in database and validation emails sent. Logs updated
Main Success Scenario:

1) User fills in form
2) Form validates
3) User sent an email to confirm address and PI sent an email to validate user
4) User clinks link in email to confirm email address
5) PI clicks link in email to confirm user is a member of their lab
6) User added to the database and provided with a wiki account. User gets a wiki page and details are added to the lab wiki page. User is sent a confirmation email containing login credentials.

Extensions:
1a) User is a PI: They must provide more information e.g. address of lab and (5a) the Chief Administrator validates their application.
2b) If form doesn't validate user is prompted for missing information
5b) If application invalid any user data removed from the database and thee applicant is informed by email of the reason for the denial of an account

Use Case Ordering Sequencing

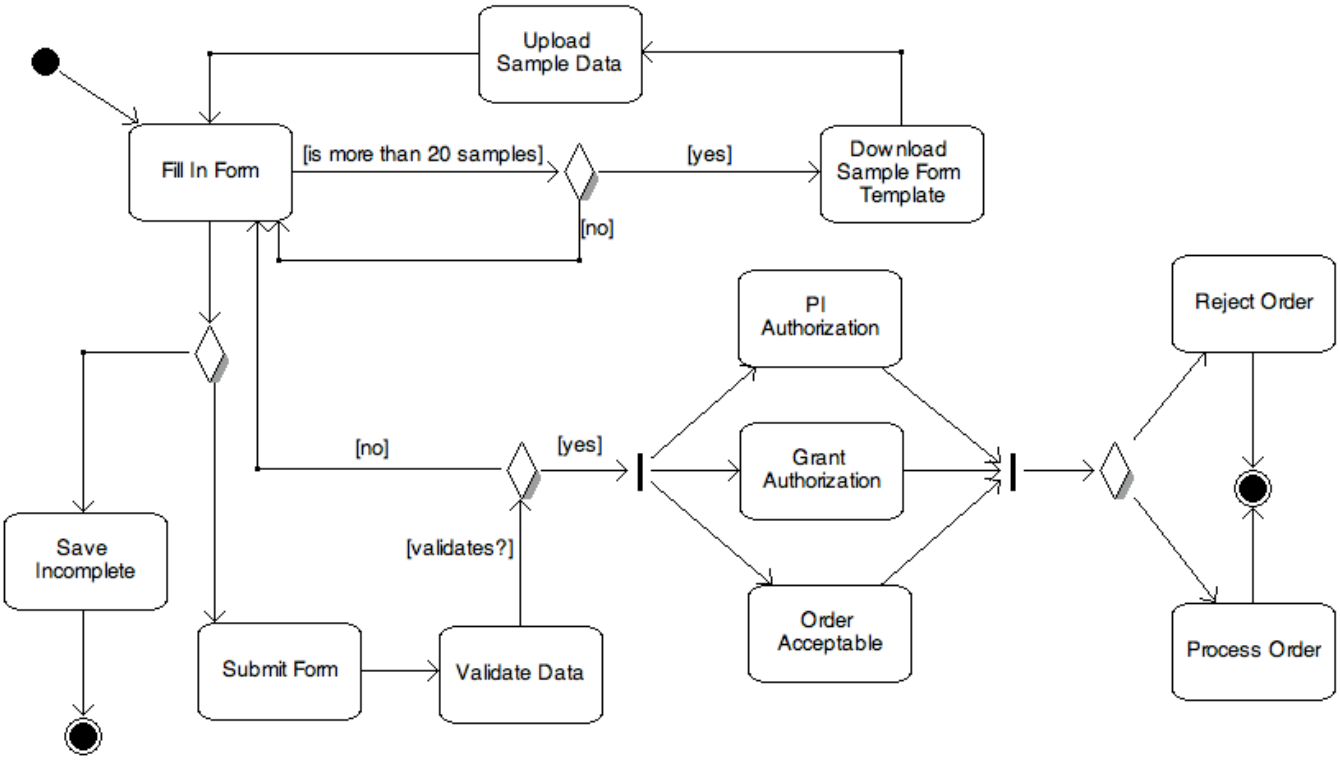


Use Case: Ordering Sequencing
Synopsis: A lab user wishes to send samples to the core facility for sequencing. They fill in a form on the Wiki. A quote for the work is issued to the user and the core facility is notified.
Primary Actor: Lab User
Scope: wasp.web
Level: Lab Member Goal
Stakeholders and Interests: Lab member wishes to order a sequencing service. Core facility wishes to obtain necessary details to perform sequencing (including QC data). WASP system requires certain information for downstream analysis of data. Finance administrators and lab PI are informed of the quote.
Precondition: Lab User has an active account
Minimal Guarantee: Logging of a form filling attempt
Success Guarantee: Database updated. Results pages created and updated with a record (status) of the order. User, PI and Administrator in charge of funds emailed a quote for the service and core facility informed of pending work order.
Main Success Scenario:

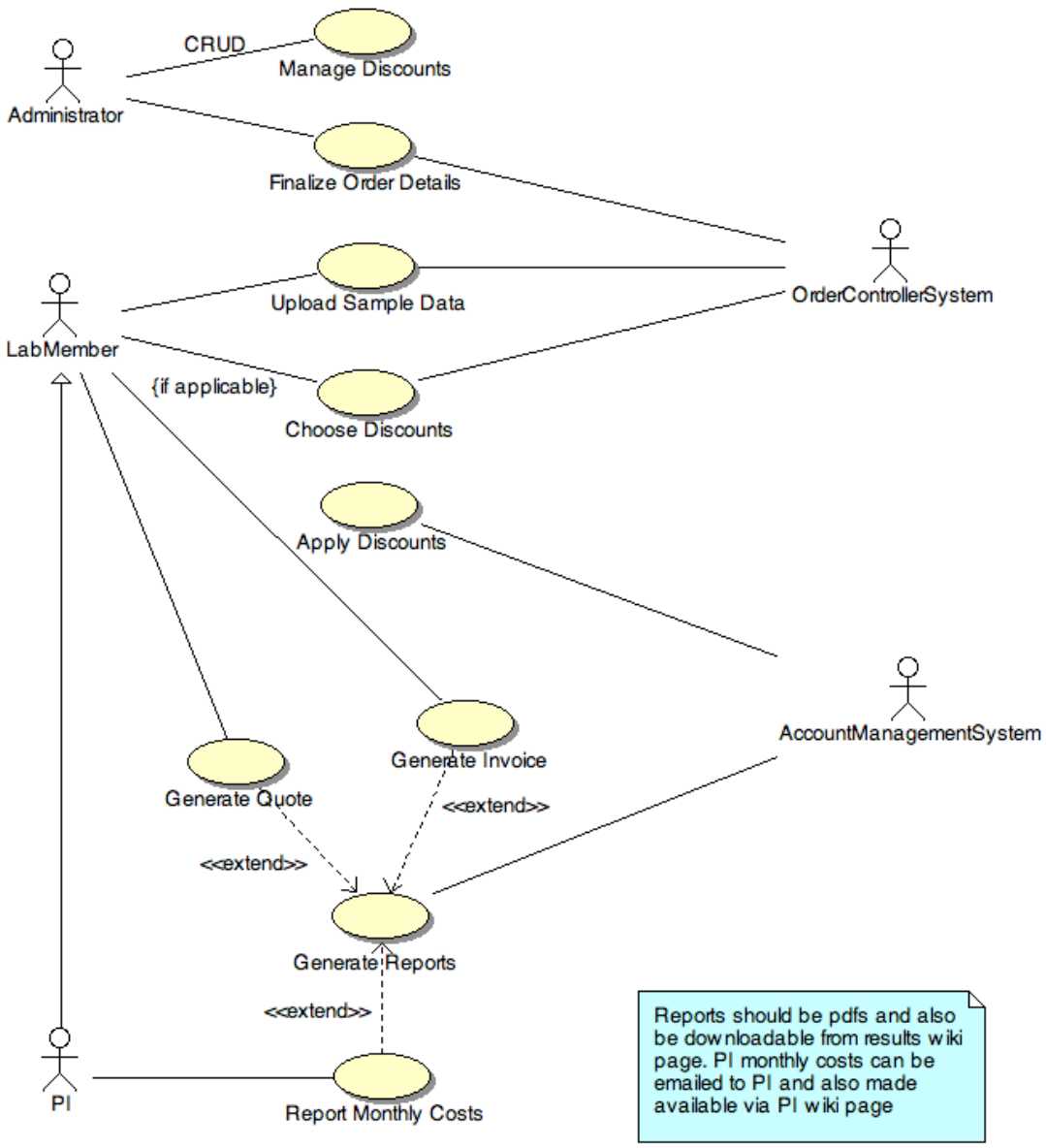
1) Lab User logs into their account on the Wiki
2) Lab User clicks link to make a new order
3) Lab User fills in order form and submits
4) Form validates.
5) Lab User confirms all details are correct
6) System creates a results page for the job and a new project page if necessary (and a link to a new project placed on the user's projects page if necessary). A link to the new job page is placed within the project page. The job page is updated with the date and a status of 'job submitted'.
7) An email is sent to the PI and Finance Administrator asking for approval of the quote (PDF attachment).
8) An email is sent to the User containing the quote (PDF attachment)
9) An email is sent to the Facility Manager containing the quote and summary of the order and asking for approval of the order.
10) PI and Administrator approve the quote and Facility Manager approves order.
11) Database updated. User informed by email that order is accepted and sequencing is pending. Tracking information on the results page is updated. Logs updated.

Extensions:
1a) Lab member login does not authenticate. Prompt for retry
3a) Lab User saves current status to complete later
4a) If a missing or incorrect entry in the form, user is prompted to correct
5a) If any details need changing, lab user can go back and change entries then re-submit.
6a) Currently these are static wiki pages created using wiki-bot. Probably better served up dynamically to wiki via embedded jsp page.
10a) If PI, Administrator or Facility Manager reject order then send email with embedded reason for rejection to Lab User and record in database / logs.

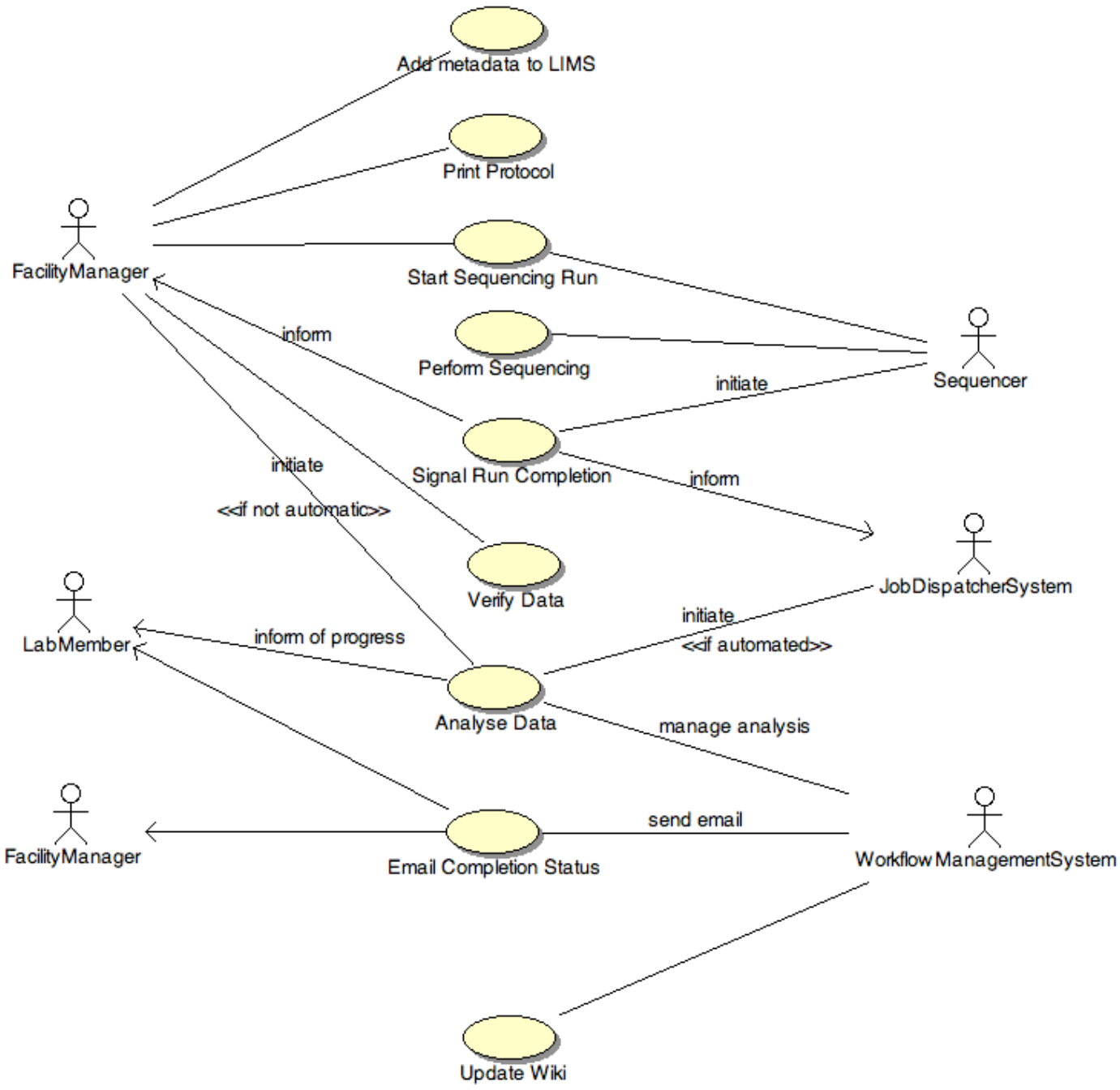
Make Order Activity Diagram



Use Case Accounts Processing

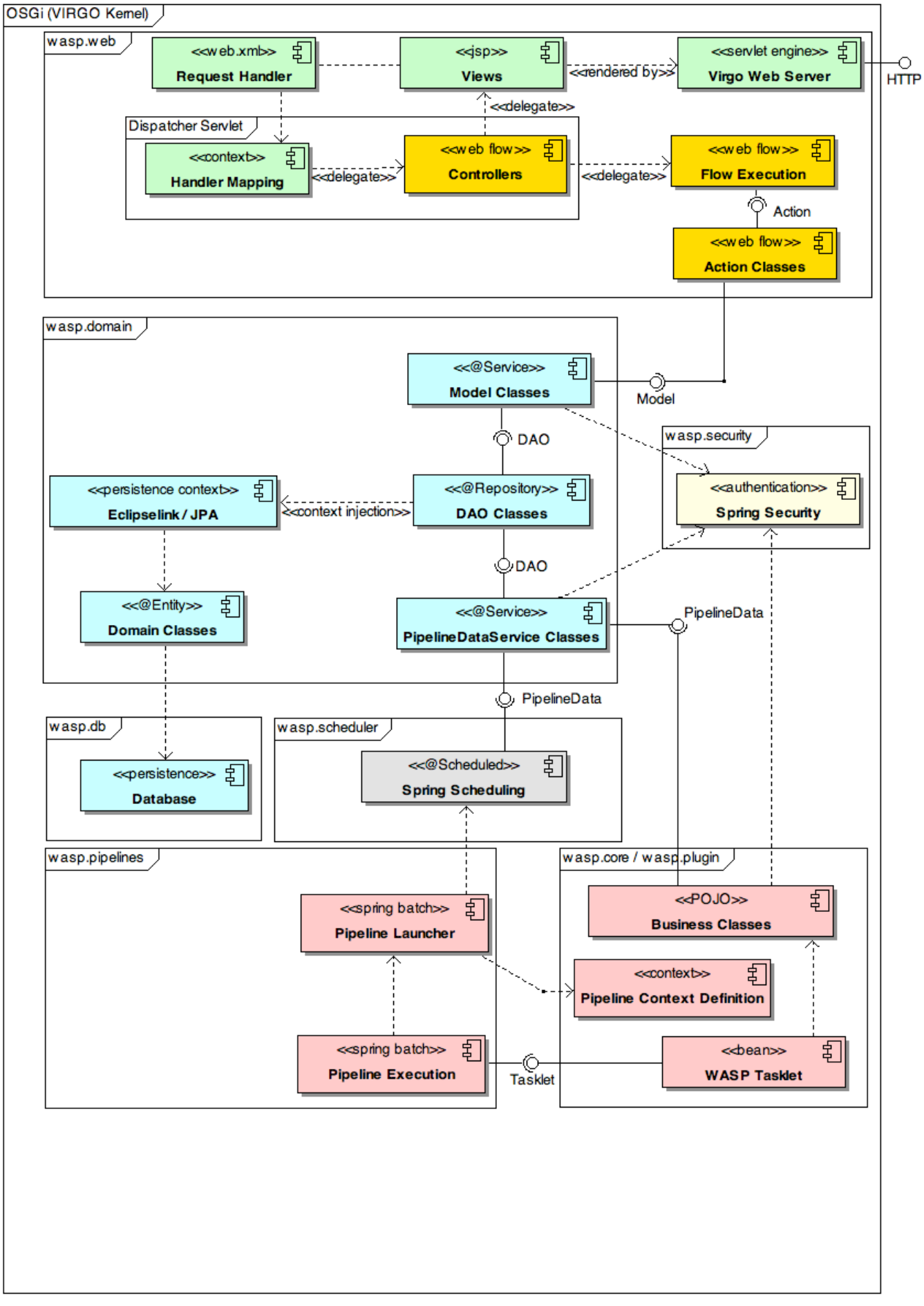


Use Case Process Data



Components Diagram (proposed rough OSGi model)

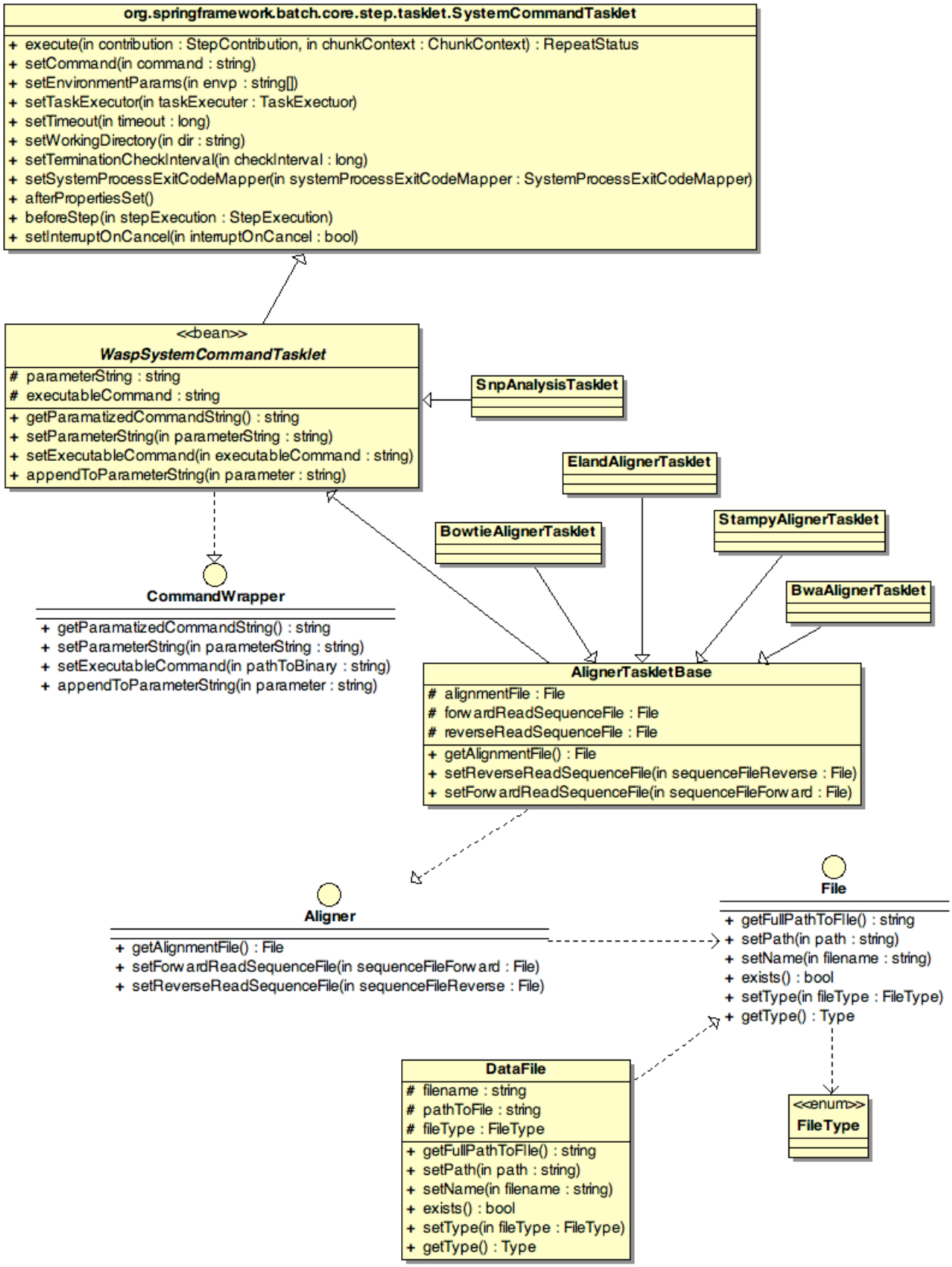
Disclaimer: This is probably really badly designed as some components are not really components (as normally defined) and it is based on a naive knowledge of OSGi and the Spring / JPA framework. Also we haven't resolved exactly how to handle plug-in persistence or properly defined a plug-in. This is one area where consultant expertise will be invaluable. There will probably be two classes of plug-in. One could be defined simply as a swappable tasklet (e.g. a wrapper around a custom or third party program) however, most will involve handling a new assay-type, i.e. will have derive specialized forms, special persistence entities, analysis workflows and results templates for display on the web. This is where good design is particularly essential and is how most third party developers will extend WASP functionality. Also there will be plug-ins which handle parsing data (including metadata) from various current (and future) sequencing machines. This part remains to be modeled.


OSGi (VIRGO Kernel)
wasp.web
<<web.xml>>
Request Handler
<<jsp>>
Views
<<rendered by>>
<<servlet engine>>
Virgo Web Server
HTTP
<<delegate>>
Dispatcher Servlet
<<context>>
Handler Mapping
<<delegate>>
<<web flow>>
Controllers
<<delegate>>
<<web flow>>
Flow Execution
Action
<<web flow>>
Action Classes
wasp.domain
<<@Service>>
Model Classes
Model
DAO
wasp.security
<<persistence context>>
Eclipselink / JPA
<<context injection>>
<<@Repository>>
DAO Classes
<<authentication>>
Spring Security
DAO
<<@Entity>>
Domain Classes
<<@Service>>
PipelineDataService Classes
PipelineData
PipelineData
wasp.db
<<persistence>>
Database
wasp.scheduler
<<@Scheduled>>
Spring Scheduling
wasp.pipelines
wasp.core / wasp.plugin
<<spring batch>>
Pipeline Launcher
<<POJO>>
Business Classes
<<context>>
Pipeline Context Definition
<<spring batch>>
Pipeline Execution
Tasklet
<<bean>>
WASP Tasklet

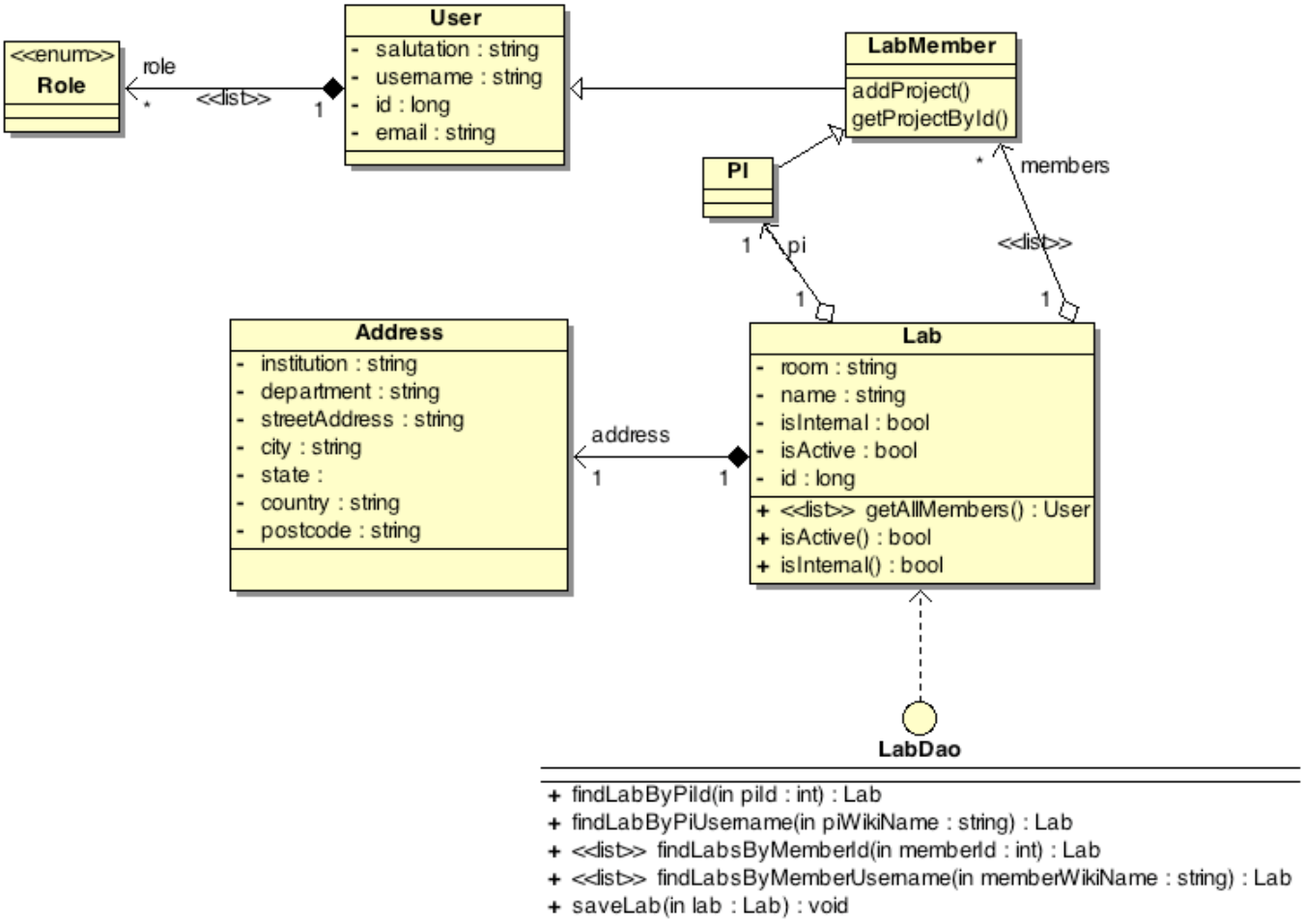
Class Diagrams

Class diagrams are not yet present for the entire system, just certain parts. Some parameters and methods have been added to help demonstrate core functionality.

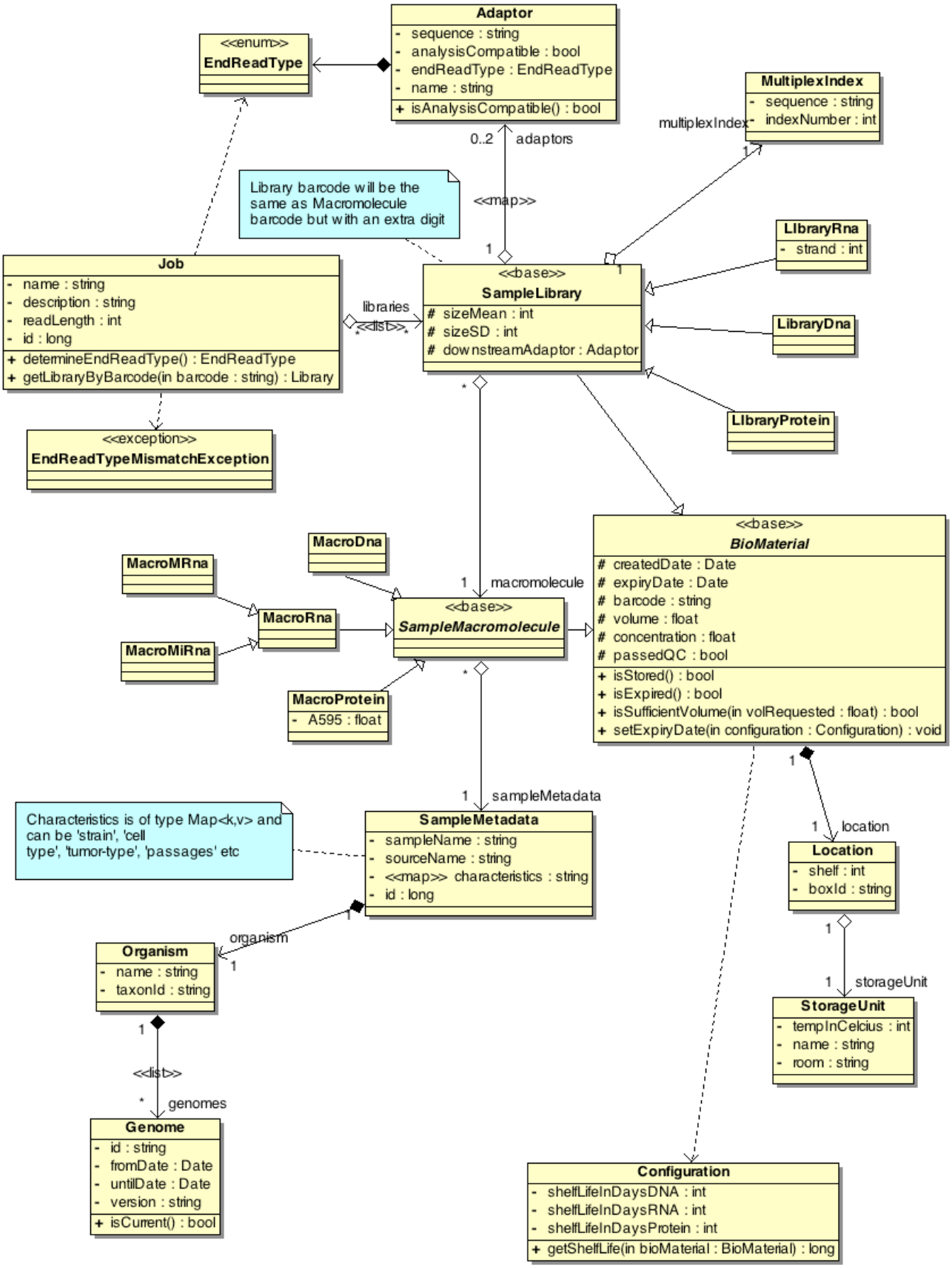
Spring Batch Tasklet Extension – one potential solution



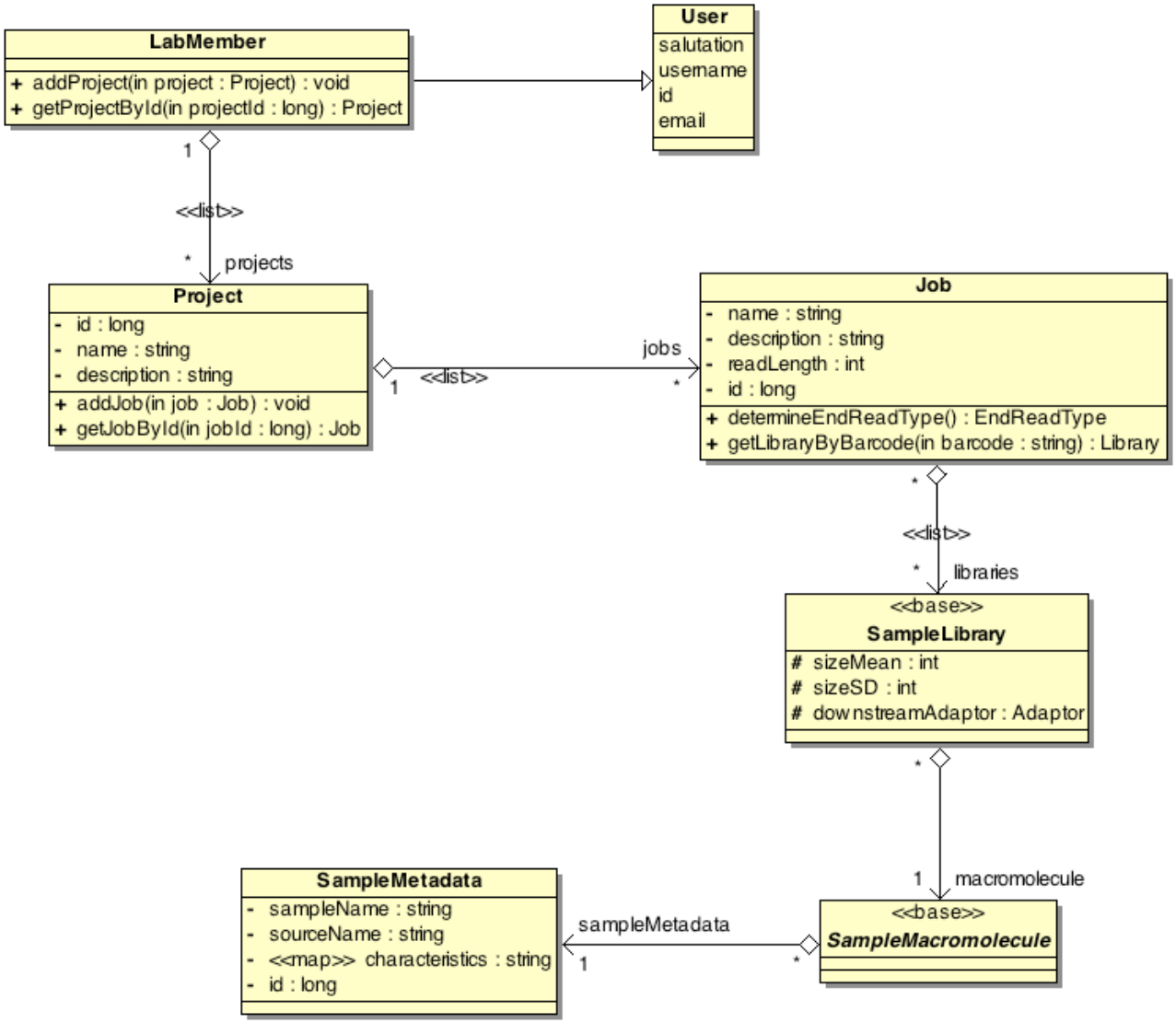
Users

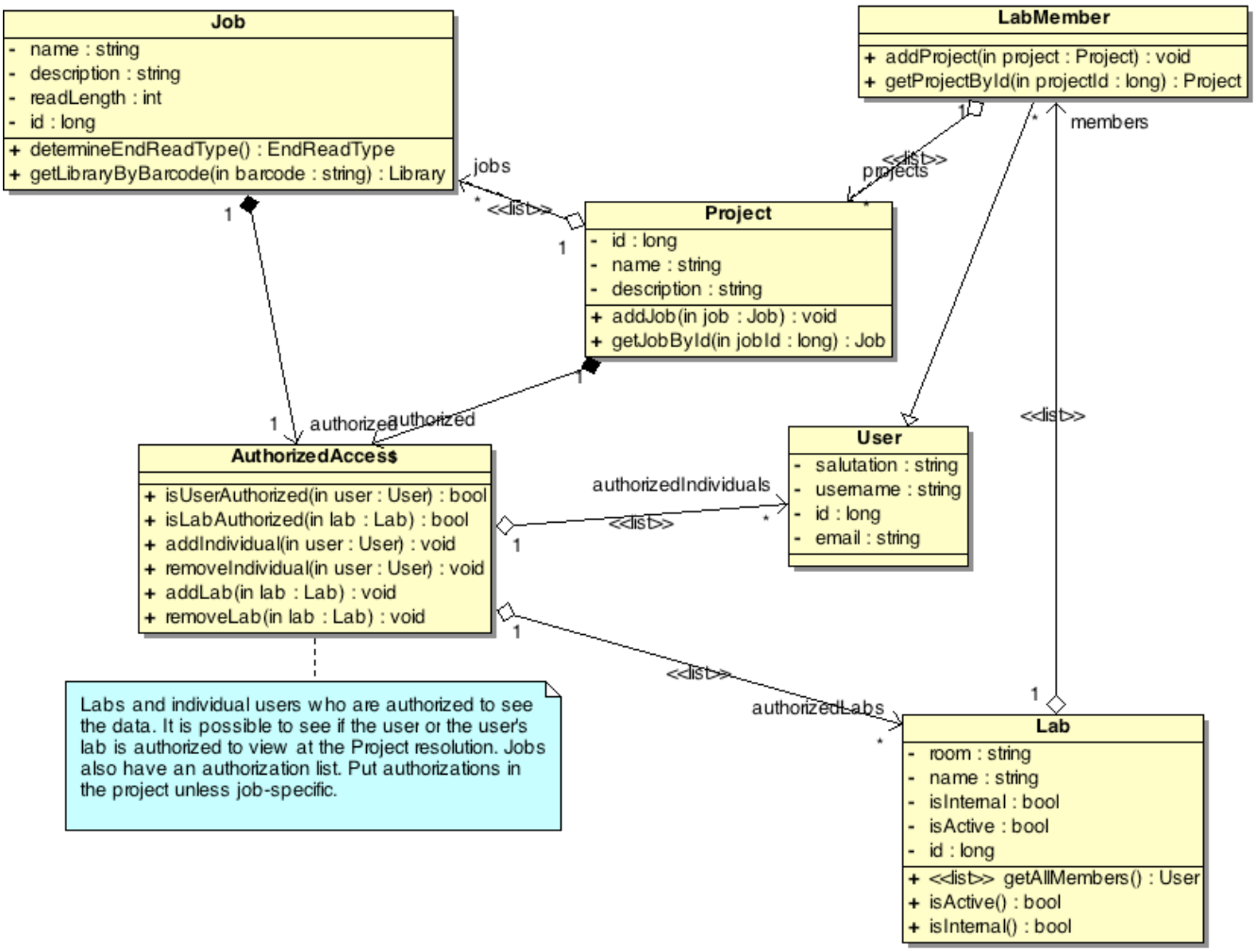

<<enum>>
Role
role
*
<<list>>
1
User
- salutation : string
- username : string
- id : long
- email : string
LabMember
addProject()
getProjectById()
PI
1
pi
1
*
members
<<list>>
1
Address
- institution : string
- department : string
- streetAddress : string
- city : string
- state :
- country : string
- postcode : string
address
1
1
Lab
- room : string
- name : string
- isInternal : bool
- isActive : bool
- id : long
+ <<list>> getAllMembers() : User
+ isActive() : bool
+ isInternal() : bool
LabDao
+ findLabByPiId(in piId : int) : Lab
+ findLabByPiUsername(in piWikiName : string) : Lab
+ <<list>> findLabsByMemberId(in memberId : int) : Lab
+ <<list>> findLabsByMemberUsername(in memberWikiName : string) : Lab
+ saveLab(in lab : Lab) : void

Samples


Adaptor
- sequence : string
- analysisCompatible : bool
- endReadType : EndReadType
- name : string
+ isAnalysisCompatible() : bool
<<enum>>
EndReadType
MultiplexIndex
- sequence : string
- indexNumber : int
multiplexIndex
0..2
adaptors
1
Library barcode will be the same as Macromolecule barcode but with an extra digit
<<map>>
LIbraryRna
- strand : int
Job
- name : string
- description : string
- readLength : int
- id : long
+ determineEndReadType() : EndReadType
+ getLibraryByBarcode(in barcode : string) : Library
libraries
<<list>>
<<base>>
SampleLibrary
sizeMean : int
sizeSD : int
downstreamAdaptor : Adaptor
LibraryDna
LIbraryProtein
<<exception>>
EndReadTypeMismatchException
<<base>>
BioMaterial
createdDate : Date
expiryDate : Date
barcode : string
volume : float
concentration : float
passedQC : bool
+ isStored() : bool
+ isExpired() : bool
+ isSufficientVolume(in volRequested : float) : bool
+ setExpiryDate(in configuration : Configuration) : void
MacroDna
MacroMRna
MacroRna
MacroMiRna
MacroProtein
- A595 : float
macromolecule
<<base>>
SampleMacromolecule
Characteristics is of type Map<k,v> and can be 'strain', 'cell type', 'tumor-type', 'passages' etc
SampleMetadata
- sampleName : string
- sourceName : string
- <<map>> characteristics : string
- id : long
sampleMetadata
location
Location
- shelf : int
- boxId : string
storageUnit
StorageUnit
- tempInCelcius : int
- name : string
- room : string
organism
Organism
- name : string
- taxonId : string
<<list>>
genomes
Genome
- id : string
- fromDate : Date
- untilDate : Date
- version : string
+ isCurrent() : bool
Configuration
- shelfLifeInDaysDNA : int
- shelfLifeInDaysRNA : int
- shelfLifeInDaysProtein : int
+ getShelfLife(in bioMaterial : BioMaterial) : long

User-sample-job-project relationships




Job
- name : string
- description : string
- readLength : int
- id : long
+ determineEndReadType() : EndReadType
+ getLibraryByBarcode(in barcode : string) : Library
LabMember
+ addProject(in project : Project) : void
+ getProjectById(in projectId : long) : Project
jobs
projects
members
<<list>>
Project
- id : long
- name : string
- description : string
+ addJob(in job : Job) : void
+ getJobById(in jobId : long) : Job
authorized
AuthorizedAccess
+ isUserAuthorized(in user : User) : bool
+ isLabAuthorized(in lab : Lab) : bool
+ addIndividual(in user : User) : void
+ removeIndividual(in user : User) : void
+ addLab(in lab : Lab) : void
+ removeLab(in lab : Lab) : void
authorizedIndividuals
User
- salutation : string
- username : string
- id : long
- email : string
authorizedLabs
Lab
- room : string
- name : string
- isInternal : bool
- isActive : bool
- id : long
+ <<list>> getAllMembers() : User
+ isActive() : bool
+ isInternal() : bool
Labs and individual users who are authorized to see the data. It is possible to see if the user or the user's lab is authorized to view at the Project resolution. Jobs also have an authorization list. Put authorizations in the project unless job-specific.