Overview

Entity

Main data model (every Entity has at least these)

_id	entity/[type]/[domain]/[documenttype]/[increment]
domain	medical | news | etc
format	image | video | text | ???
type	factor_span
title or tags (todo)	
parents	array() of the direct parent(s) of an entity. Example: the jobconf on which a new one is based, or all the sentences of a batch
useragent_id	CrowdWatson
activity_id	
created_at	MongoDate
updated_at	MongoDate

Types

All these are described in more detail **below**.

documentType	parents	softwareAgent_id	description
job		(platformname)	1 Job = 1 CF Job or multiple AMT HITs. Refers to JobConf, GoldBatch and Batch.
jobconf	adapted from	jobcreator	Settings: title, payment, etc.
questiontemplate	adapted from	templatebuilder	In JSON format. Also has replace rules
batch	unit id's	batchcreator	Group of units

goldbatch (todo)			Batch of gold questions
unit	raw data	structurer or other preprocessor	‘sentence’ (or image, or…)
annotation	unit	cf / amt / ???	1 annotation by 1 worker on 1 unit

Agents

Main data model

_id	(see below)
created_at	MongoDate
updated_at	MongoDate

Types

type	*_id*	description
user_agent	[username]	User on our platform or ‘CrowdWatson’ (if no user is logged in)
crowd_agent	crowdagent/amt/A374JDHKSHJ	Worker. **See below**
softwareAgent_id	fileuploader | amt | etc	crowdplatform, ‘tab’ on our platform, sentenceSplitter, faceRecognizer, etc

Activity

Main data model

_id	activity/[softwareagent]/[increment]
label	see below

used	entity uri
softwareAgent_id	
user_agent	
crowdAgent_id	
created_at	MongoDate
updated_at	MongoDate

Main types

softwareAgent_id	*used*	*label*
fileuploader		
sentencesplitter		
videosegmenter		
batchcreator		
templatebuilder	questiontemplate	Questiontemplate is saved.
jobcreator		Job is uploaded to crowdsourcing platform(s): [platforms].
[platform]	job	Units are annotated on crowdsourcing platform.
spammeridentifier	annotation	

Entities: more detailed

Job [Entity]

Common fields:

documentType	job

agent_id	[reference]
activity_id	[reference]
jobconf_id	[reference]
batch_id	[reference]
softwareAgent_id	platformname
platformJobId	**For AMT:** for each HIT: array('id'=>$id, 'status'=>'running') **For CF**: 382004
status	**See bottom of document**
startedAt	MongoDate
finishedAt	MongoDate
runningTimeInSeconds	Diff in seconds between startedAt and finishedAt
unitsCount	total number of units
annotationsCount	number of annotations completed
completion	(0.00 - 1.00)
results	array($unitid => [added vector])
projectedCost	Cost of the job if it's completed
template	[soon to be replaced by QuestionTemplate_id]
	(annotations have a job id)

AMT Specific

HITGroupId	2UYNQYN5F3Q3ISB07ZECZYB493Z9D3 (HIT's with the same GroupId show as batches for workers. Based on title, reward, description, etc).
HITTypeId	2FUCOO2GMMEJN0MX1PD6LAB3XHP77I (used for e-mail notifications)
Expiration	MongoDate

CF specific

Updated_at	MongoDate

* Note: A batch has one CF Job, but mostly more AMT HITs (1 HIT per unit [sentence]) is common. This can be misleading, so keep it in mind. Units reference back to the Job and each Job, if created at the same time, refers to the same activity.

HITReviewStatus is left out, because it's an AMT internal thing (about if they think if our HIT is inappropriate).

JobConfiguration [Entity]

documentType	jobconf
hash	md5(serialize(content)) (to see if we have to save a new one or reference an existing JobConfiguration from the Job)
questiontemplate_id	[reference]
content	title description instructions keywords annotationsPerUnit unitsPerTask reward expirationInMinutes notificationEmail requesterAnnotation /* AMT specific */ autoApprovalDelayInMinutes hitLifetimeInMinutes qualificationRequirement assignmentReviewPolicy frameheight /* CF specific */ annotationsPerWorker countries /* for our use */ platform (array('cf', 'amt')) answerfields (the fields of the CSV file that contain the gold answers) (soon to be changed)

QuestionTemplate [Entity]

documentType	questiontemplate
content	- question (json) - replace (array of columns in batch to be replaced - IE array('sentence.noPrefix' => array('cause' => 'causes')).) **[under construction]**

Batch [Entity]

documentType	batch
content	‘title’
parents	[reference to units]

Annotation [Entity]

Common fields

documentType	annotation
crowdagent_id	[reference] (worker)
softwareAgent_id	[reference] (platform)
job_id	[reference]
unit_id	[reference]
acceptTime	MongoDate (started_at in CF terminology)
submitTime	MongoDate (created_at in CF terminology)
platformAnnotationId	**For AMT**: 2UFQY2RCJK66Y7KAQ4RKXIAQB2UU53 **For CF**: 1186540849 (NB this is an Integer!)

content	Array(question => answer)
status	**For AMT**: Submitted | Approved | Rejected **For CF**: We'll probably have to check if 'rejected' and 'reviewed' contain dates.

CF specific

trust	0.75196428571429
external_type	instantrewardz (external platform)
cfChannel	cf_internal or ...

And: tainted, rejected, reviewed, missed, golden. (possibly rejected and reviewed can contain a date).

AMT specific

SubmitTime	MongoDate
ApprovalTime	MongoDate or null
RejectionTime	MongoDate or null
AutoApprovalTime	MongoDate (or null?)

Agents: more detailed

CrowdAgent

platform	[reference to softwareAgent]
PlatformAgentId	**For CF**: 19822336 **For AMT**: A1M46I0H8KNEEX
unitCount	count type (array of types with count) format (array of formats with count) domain (array of domains with count)
annotationsCount	(see above)

jobCount	(see above)

CF specific (we get this from a judgment)

worker_trust	0.73823529411765
country	US
city	Fresno

AMT Specific:

- NumberAssignmentsApproved
- NumberAssignmentsRejected
- PercentAssignmentsApproved
- PercentAssignmentsRejected
- NumberKnownAnswersCorrect
- NumberKnownAnswersIncorrect
- NumberKnownAnswersEvaluated
- PercentKnownAnswersCorrect
- NumberPluralityAnswersCorrect
- NumberPluralityAnswersIncorrect
- NumberPluralityAnswersEvaluated
- PercentPluralityAnswersCorrect

But for every field we want to know we have to execute a separate query to the API.

UserAgent

SoftwareAgent

Extra: Job Status

CF Job state	*AMT HIT status*	*our Job status*
new / unordered	(in sandbox)	**unordered**
running	Assignable/Unassignable*	**running**
paused		**paused**

	Reviewable/Reviewing**	**review**
canceled***		**canceled**
finished job	? / Disposed***	**finished**

* Unassignable when the last assignment is started
** We can set the state to Reviewing from the API
*** We have to decide if and when to dispose jobs.

http://mechanicalturk.typepad.com/blog/2011/04/overview-lifecycle-of-a-hit-.html

Assignable – a HIT can be accepted by a Worker
Unassignable – a HIT has been accepted by a Worker and is being worked on and therefore cannot be accepted by another Worker
Reviewable – a Worker has submitted answers to a HIT and the HIT is available for review
Reviewing (optional state) – the HIT is currently being reviewed by the Requester
Disposed – the HIT has been deleted and can no longer be retrieved.