Graduation Report

Crowd-Watson

Crowdsourced Text Annotations

Hui Lin

Information Communication Technology

Fontys University of Applied Science

IBM Netherlands

June, 2013

This page intentionally left blank

Fontys
University of Applied Sciences

Graduation Report

Fontys University of Applied Science
Department of Information Communication Technology
HBO – ICT English Stream

Student	Hui Lin	Student Number	: 2142435
		Study Program	: HBO – ICT English Stream
		Project Period	: 1st February – 29th June 2013
Company	IBM Netherlands	Address	: Johan Huizingalaan 765, 1066 VH Amsterdam
		Department	: CAS (Center for Advanced Studies)
Company Tutor	Manfred Overmeen	Position Title	: IT Specialist
		Email	: manfred.overmeen@nl.ibm.com
		Mobile Phone	: +31 6 51349190
School Tutor	Erik van der Schriek	Email	: e.vanderschriek@fontys.nl
		Phone	: 0885076608

Approved and signed by the company tutor

Date:

Signature:

Preface

I am a fourth year student studying Information Communication Technology at Fontys University of Applied Sciences. This report is about my graduation project conducted both at IBM Netherlands and VU University Amsterdam. It describes why, what and how I have done my graduation project. It is presented by following the Guidelines for Writing a Report (version 13 April 2012) by Fontys.

It's a very unique opportunity for me to gain real working experiences in a world-leading company, IBM and to become familiar with the research university environment. Working on the Crowd-Watson team as my graduation project brought new challenges as well as chances of learning and growth to me. It helped to broaden my horizons, to improve abilities of self-study, problem solving and communication and to keep myself motivated and stay on target.

I would like to thank all the members of Crowd-Watson, my company and university tutors; Manfred Overmeen, Robert-Jan Sips and Erik van der Schriek for all the instructions and help during my graduation period.

Hui Lin

20, May 2013

Amsterdam

Contents

List of Figures

Summary

This report indicates the processes that the intern has completed during the graduation period, at IBM Netherlands and VU University Amsterdam. The contents of the report may be abstract or technical to a certain extent, due to the security and confidentiality policies of IBM. However, it does not prevent the reader from understanding the work conducted by the intern since the report is mostly focused on how the intern performs the graduation project.

This graduation project is about building a crowdsourced text annotations tool with the combination of crowdsourcing and automation modules for the Crowd-Watson team. The purpose of the project is to solve the lack of efficiency and uniformity of current research in crowdsourcing for Watson and to support the further extensive research by enabling data storage and integration.

The conclusions obtained are that the tooling delivered by this graduation project will improve efficiency in setting up large-scale experiments and uniformity in interaction with various crowdsourcing platforms. It helps researchers to assign and manage microtasks through different platforms efficiently, to keep track on history data methodically and to generate statistical output dynamically. Changes during the execution of the project are not to be considered as problems, rather being flexible solutions better to solve issues based on the research output of the Crowd-Watson project. Due to multiple modules, various data and a lot of statistical calculations, the final verification of tooling is essential.

This graduation project brought the intern a new challenge as well as a great chance of practicing programming skills and learning how to cooperate in a research project team.

Glossary

AI	Artificial Intelligence
API	Application Programming Interface
ASD	Agile Software Development
CAS	Center for Advanced Studies
CF	CrowdFlower (a leading Crowdsourcing Web service)
CTR	Computing Tabulating Recording Corporation
FTP	File Transfer Protocol
GBS	Global Business Services
GTS	Global Technology Services
GUI	Graphical User Interface
GWAP	Games with a purpose
IBM	International Business Machines
ITG	Innovation and Technology Group
LAMP	Refer to first letters of Linux, Apache, MySQL and PHP
NLP	Natural Language Processing
STG	Systems and Technology Group
SWG	Software Group
TD	Testing and Debugging
UED	User Experience Design

1. Introduction

This graduation report intends to cover all the information, activities and results associated with the graduation project performed by the intern at IBM Netherlands and VU University Amsterdam. IBM (International Business Machines) is a multinational computer technology and IT consulting corporation headquartered in New York. The company manufactures and sells computer hardware and software, and offers infrastructure services, hosting services, and consulting services in areas ranging from mainframe computers to nanotechnology.

Artificial Intelligence (AI) is defined as anything that makes machines act more intelligently. AI has a long history at IBM, dating back to the 1950s. For this technical field, IBM has been focusing on basic and applied research in machine learning, deep question answering, search and planning, knowledge representation, and cognitive architectures.

IBM Benelux Center for Advanced Studies (CAS Amsterdam) has been established in Amsterdam by IBM Netherlands to stimulate joint innovation between IBM, clients and local scientific communities. The Crowd-Watson project which started in 2012 is collaboration between the VU University Amsterdam and CAS Amsterdam of IBM Netherlands. The intern works in the Crowd-Watson team for the graduation project.

IBM’s question-answering computer, Watson (winner of a recent Jeopardy competition), will be applied to medical fields. The Crowd-Watson team is investigating the use of crowdsourcing (tasks that computers are unable to do yet are split into micro tasks and dispatched to people) in text understanding. The output of the Crowd-Watson project will be the part of NLP (Natural Language Processing) training for Watson, so that it can digest research papers or information and become a diagnostician assistant for physicians.

Currently, the Crowd-Watson team is lacking efficiency to set up large-scale experiments and is in need of uniformity in interactions with various crowdsourcing platforms. In order to solve these two main problems and to support the further extensive research, the intern aims to build a crowdsourced text annotations tool with the combination of crowdsourcing and automation modules as her graduation project. She is responsible to research the framework, create the architecture and make integration of modules and data.

This report starts with an introduction to this graduation project and continues with the explanations about the company where the intern conducts the project and about the graduation assignment details through chapters 1-3. Chapters 4-6 explain how the intern did the research, the methodology and approach used to conduct the project which is an adapted Agile Methodology, how the project is executed and the results obtained.

2. The Company

This chapter not just provides the general information of IBM such as the history, products or services, markets and some figures but also indicates the position of the intern in this organization.

2.1 Introduction – IBM

International Business Machines Corporation, abbreviated IBM, is a multinational computer technology company and consulting corporation. It is one of the oldest information technology companies in the world.

The roots of IBM start in the late years of the 19th century. It was founded in 1896 as the Tabulating Machine Company. Later, in 1911 it was incorporated as Computing Tabulating Recording Corporation (CTR). In 1924 the current known name IBM was adopted.

By the 1950s IBM became the most dominant vendor of computer systems when they introduced the IBM 701 and other models in the 700/7000 mainframe series. As of today, IBM is the largest and most successful IT company in the world. It is also the owner of the largest number of patents in the world.

IBM manufactures and sells computer hardware and software and offers several services like infrastructure, hosting and consulting services in all sorts of different areas of computing.

2.2 Business Core

IBM's business is based on all layers of current technology from mainframes to nanotechnology. Based in Armonk, New York, in the United States, IBM is one of the largest and most portable companies in the US.

IBM has a complex internal structure. Aside of the internal departments such as Finance, Legal and Human Resources, IBM consists of 5 business groups:

<u>Global Business Services (GBS)</u>

The GBS group offers deep technical and strategic expertise to address practically any client need or problem.

Global Technology Services (GTS)

GTS focuses on the technical aspects of services. The GTS delivery model calls for a portfolio of services to be managed and delivered through regional services delivery units around the world.

Software Group (SWG)

IBM created the SWG in 1995 to pull together all of IBM's infrastructure, middleware and operational software from dispersed units of IBM.

Systems and Technology Group (STG)

IBM offers servers of all sizes and storage systems to meet the needs of the largest networks. These systems include the market leading line of servers and storage.

Innovation and Technology Group (ITG)

ITG, started at IBM in 1945, has invented much of the technology underlying IBM products and services and the IT industry at large.

Each of these business units within itself is split amongst the different business areas. Besides these business units, there is an Sales and Distribution department which moves across all of the business groups.

2.3 Facts & Figures

IBM has been well known through most of its recent history as one of the world's largest computer companies and systems integrators.

- With over 433.362 (2012) employees worldwide, IBM is one of the largest and most profitable information technology employers in the world.
- IBM holds the most patents than any other U.S. based technology companies and has 11 research laboratories worldwide.
- The company has scientists, engineers, consultants, and sales professionals in over 170 countries.
- IBM employees have earned 5 Nobel Prizes, 4 Turing Awards, 5 National Medals of Technology, and 5 National Medals of Science. [1]

2.4 IBM Benelux Organizational Structure

The internal structure of IBM would be described as a multidimensional box. The following chart indicates organizational structure of workplace where the intern performed her graduation project.

Benelux Marketing, Communications & Citizenship 2013

(It is a part of IBM GBS)

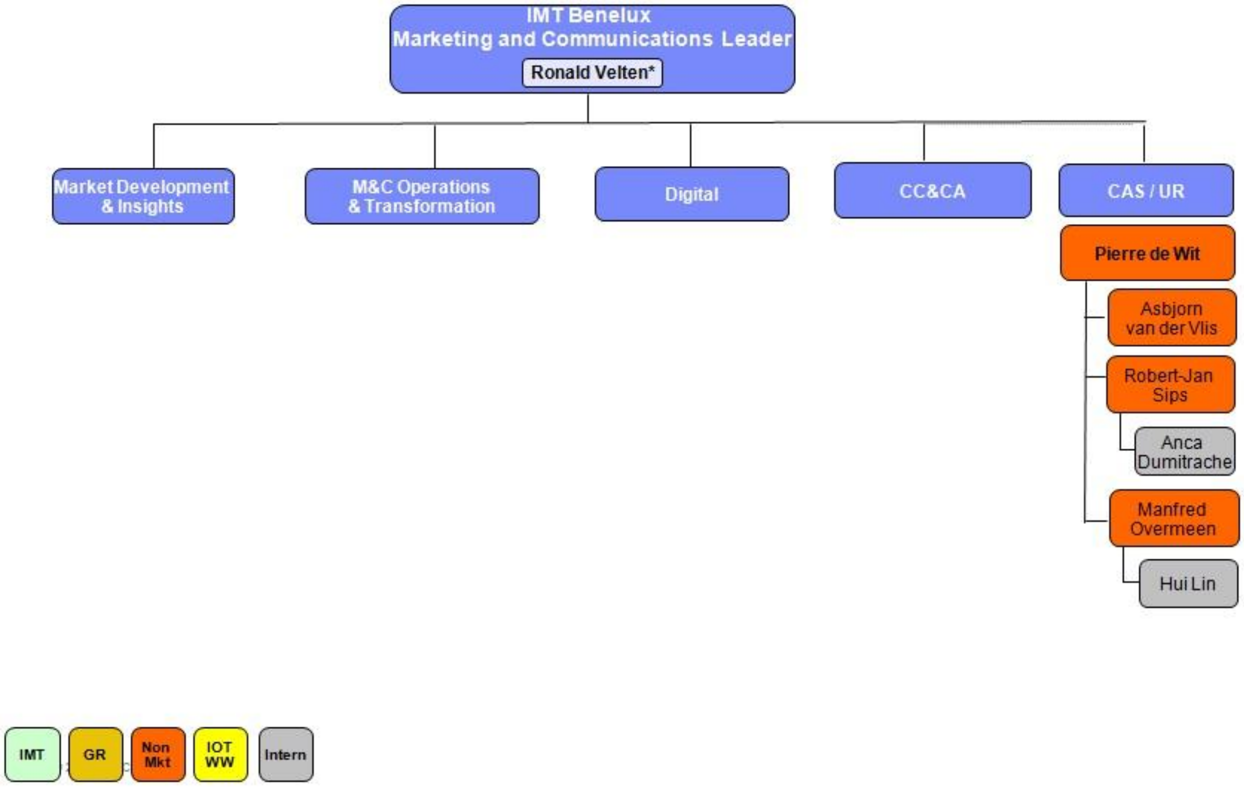


Figure 1: IBM Benelux Organizational Structure

3. The Assignment

This chapter introduces the graduation assignment by describing the project team, current problems of the project team, and objectives of this graduation project and responsibilities of the intern.

3.1 The Crowd-Watson team

IBM's question-answering computer Watson is an example of Artificial Intelligence. Watson is able to understand human natural language. It was named after IBM´s founder Thomas J. Watson and specially developed to answer questions on the quiz show Jeopardy. In 2011, Watson beat former winners Brad Rutter, and Ken Jennings on Jeopardy. However, Watson cannot think like the human brain, but can only learn, synthesize and provide an output. Through various NLP training, Watson is becoming more and more knowledgeable.

Watson would bring a lot of benefits to our daily life and business in terms of applying AI technology. Currently, IBM research team focuses on how to improve Watson to contribute more to the medical fields.

In the past decade, the human-based computation became an important research subject in computer science. Human-based computation is described as a computer science technique in which certain steps of a computational process are performed by humans instead of computers [2]. A reason for outsourcing certain steps of the process to humans is to take advantage of the people knowledge in order to solve some types of problems that are not yet accessible for computers [3]. This approach increases the quality of work and offers more possibilities of solving labor-intensive tasks.

Since using humans for solving difficult problems that computers are not able to tackle is an expensive task, the more recent research in the area tries to create a hybrid approach that combines both humans and computers intelligence. Moreover, it is well known that a group of workers may not work continuously on a specific task. Thus, an important research question is the following: how can we design a system that manages to keep the same workers on for a long time period?

The most important human-based computation techniques are the following: microtask crowdsourcing [4] and the games with a purpose (GWAP) [5], which are closely followed by citizen science [6] and interactive computations [6].

Crowdsourcing is the process of getting work or funding, usually online, from a crowd of people. The term is a combination of the words ‘crowd’ and ‘outsourcing’. The idea is to take work and outsource it to a crowd of workers. By canvassing a large crowd of people for ideas, skills or participation, the quality of content and idea generation will be greater.

Microtasking involves breaking work up into tiny tasks and sending the work to a crowd of people then deciding the payment for each completed task. Work is done faster, cheaper, and usually with less errors.

The Crowd-Watson team is investigating the use of microtask crowdsourcing and the "games with a purpose" (GWAP) in text understanding. The output of the Crowd-Watson project will be the part of NLP training for Watson so that it can easily digest the many annual research papers and blogs and keep up to date with clinical advances in a way few human physicians can do these days. Combining that database with some AI 'rules' can in theory allow Watson to become a good diagnostician and function as a 'helper' or 'Brain extender' for a physician, and perhaps provide a type of 'second opinion'.

Due to the lack of tooling which integrates various modules and data, a tool is necessary to help the further extensive research. The assignment is to build Crowd-Watson Crowdsourced Text Annotations tool.

3.2 Current Problems of the Crowd-Watson team

At current, the Crowd-Watson project is hampered by **the lack of**:

- Efficiency in setting up large-scale experiments (all work and analysis is conducted manually)
- Uniformity in interaction with various crowdsourcing platforms
- Data storage and data integration

3.3 Research Goals of the Graduation Project

The research goals of the graduation project are to investigate and implement an efficient framework for handling microtask management. Such a framework should be able to:

- Process a –configurable- number of inputs (e.g. 1 sentence, 2 sentences, a text-fragment),
- Process the input(s) by means of a workflow, consisting of an arbitrary number of processing nodes. Processing nodes may be human (e.g. crowdsourced), machine (thus: automated) and others (e.g. gaming). To cope with the "other" requirement, a standardized interface should be available towards these nodes,
- Process the output, in terms of default statistics, e.g. success rate, number of annotations per sentence, number of sentences per worker, etc.

- Finally, give a view of the outcome / results (independent of the type of processing conducted).

For any views, the use of (HTML/XML) templates may be considered.

3.4 The Intern´s Responsibilities in the Crowd-Watson team

1. Find out the suitable framework which can solve problems by analyzing **Jabberwocky** & **CrowdLang** frameworks

2. Design architecture with the needs for configurability, scalability and extendibility

3. Implement the designed architecture
   - Integrate and test CrowdFlower module from Oana Inel, Statistics module from Guillermo Soberon and Games module from Anca Dumitrache
   - Enable data storage and data integration
   - Enable users to track and retrieve history and results data dynamically

3.5 Organization Chart

Crowd-Watson is a project involving multiple group members for different roles and responsibilities during the project execution. To be better able to indicate the role and position of the intern, the organization chart relative to the project Crowd-Watson is shown in this section.

Lora Aroyo and Chris Welty are project leaders who research in Crowdsourcing for Watson, organize regular Crowd-watson-dev meetings, deliver user requirements, keep track of project processes and bring research information. Michiel Hildebrand, Robert-Jan Sips and Manfred Overmeen offer relative research or development support and coordination regarding to the Crowd-Watson project.

Oana Inel, Guillermo Soberon and Anca Dumitrache focus on researching different modules and deliver relevant research output. The intern is responsible for research in the framework, create the architecture and make the final integration.

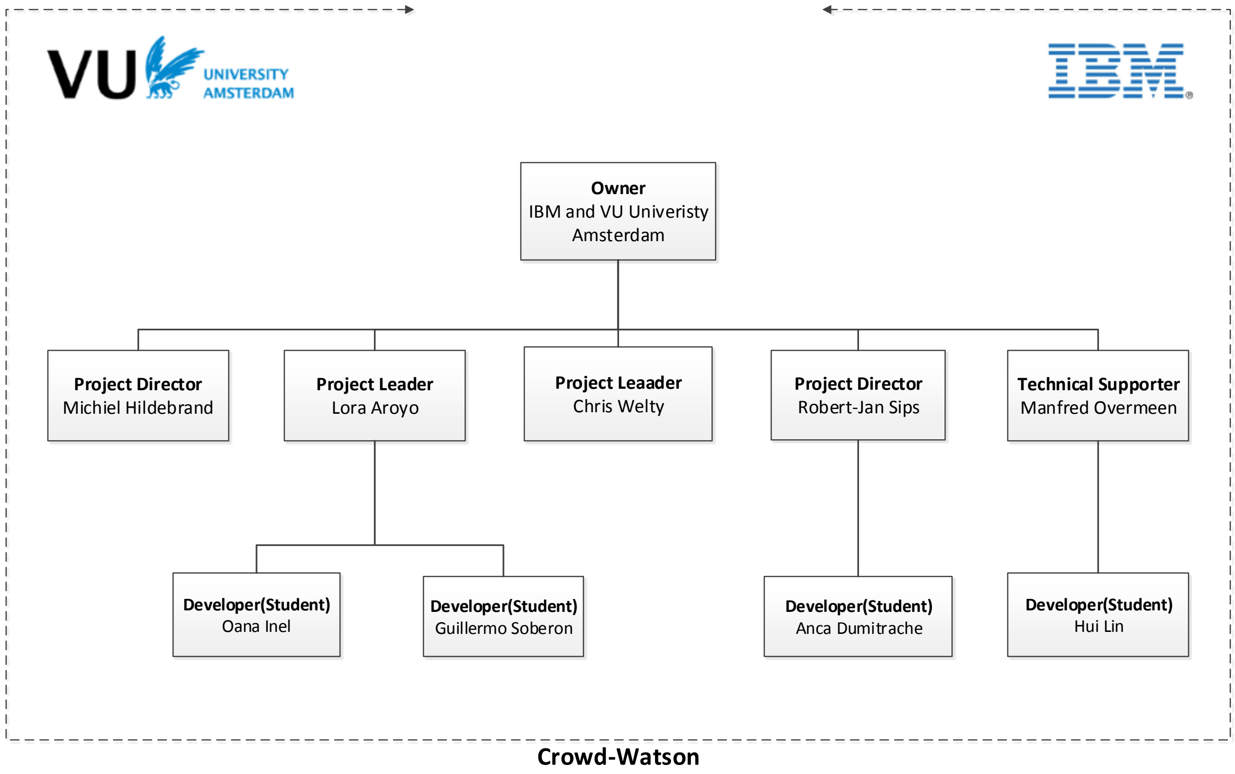


Figure 2: Organization Chart of Crowd-Watson

Team member	Role	Responsibilities
Lora Aroyo	Researcher at VU University	- Start-up plan and initial analysis - Create Project Charter, Research Case, Project Product Description (URS) and Project Schedule - Offer relative guidance and feedback
Chris Welty	Researcher at IBM New York (**One of the developers of Watson**) [7]	
Robert-Jan Sips	University Relations Manager at IBM Amsterdam	
Manfred Overmeen	IT Specialist at IBM Amsterdam	Offer guidance and feedback
Michiel Hildebrand	Research at VU University	
Oana Inel	Computer Science Master student at VU University	Researching in **CrowdFlower** module
Guillermo Soberon Casado	Artificial Intelligence Master student at VU University	Researching in **Statistics** module
Anca Dumitrache	IBM Intern / Artificial Intelligence Master student at VU University	Researching in **Games** module
Hui Lin	IBM Intern / ICT Bachelor student at Fontys	Researching in **framework architecture** and **integration**

Figure 3: Roles and Responsibilities Chart

4. Research

This chapter is divided into three sections:

- Investigation: describes how the intern investigated two framework examples (Jabberwocky and CrowdLang) based on several criterions and concluded the final solutions for the Crowd-Watson team.
- Methodology: indicates which methods the team used to conduct the Crowd-Watson project.
- Implementation: gathers information about what kinds of tooling and programming environments are set up to implement the crowdsourced text annotations tool for the Crowd-Watson team.

4.1 Investigation

<u>Framework Descriptions</u>

Jabberwocky (Stanford University)

It is a programming environment developed for social computing (a general term for an area of computer science that is concerned with the intersection of social behavior and computational systems).

It is composed of three different layers: Dormouse, ManReduce and Dog.

- Dormouse – a human and machine resource management system
- ManReduce – a parallel programming framework for human and machine computation
- Dog – a high-level programming language on top of ManReduce

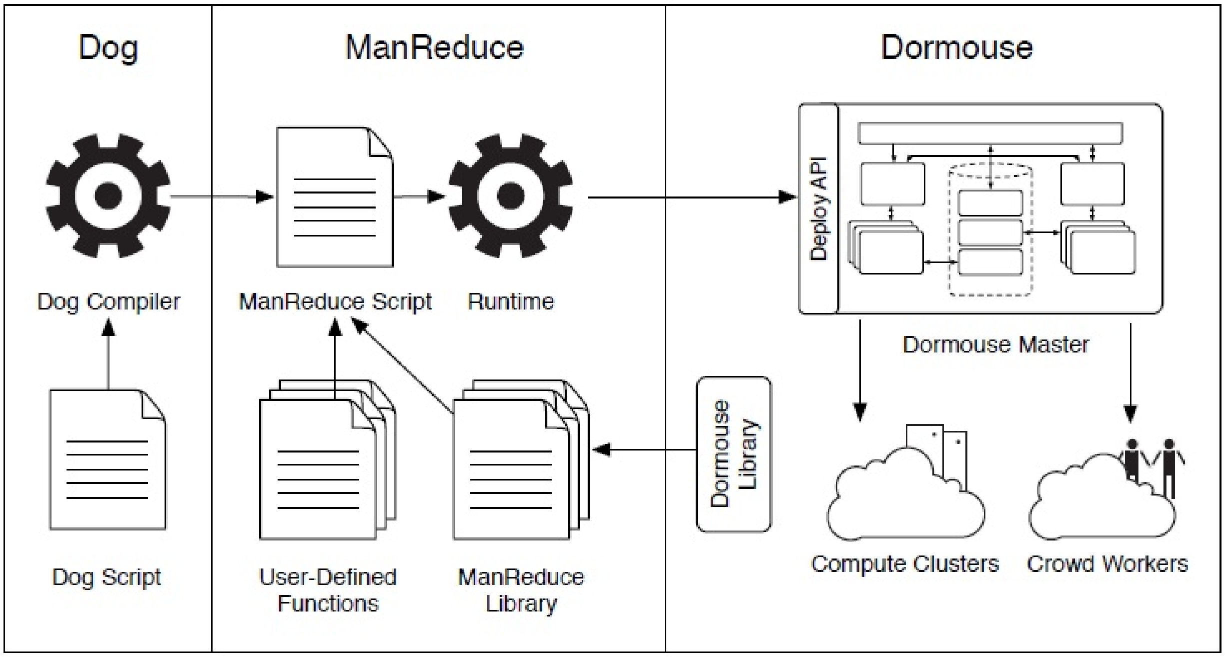


Figure 4: Overview of Jabberwocky

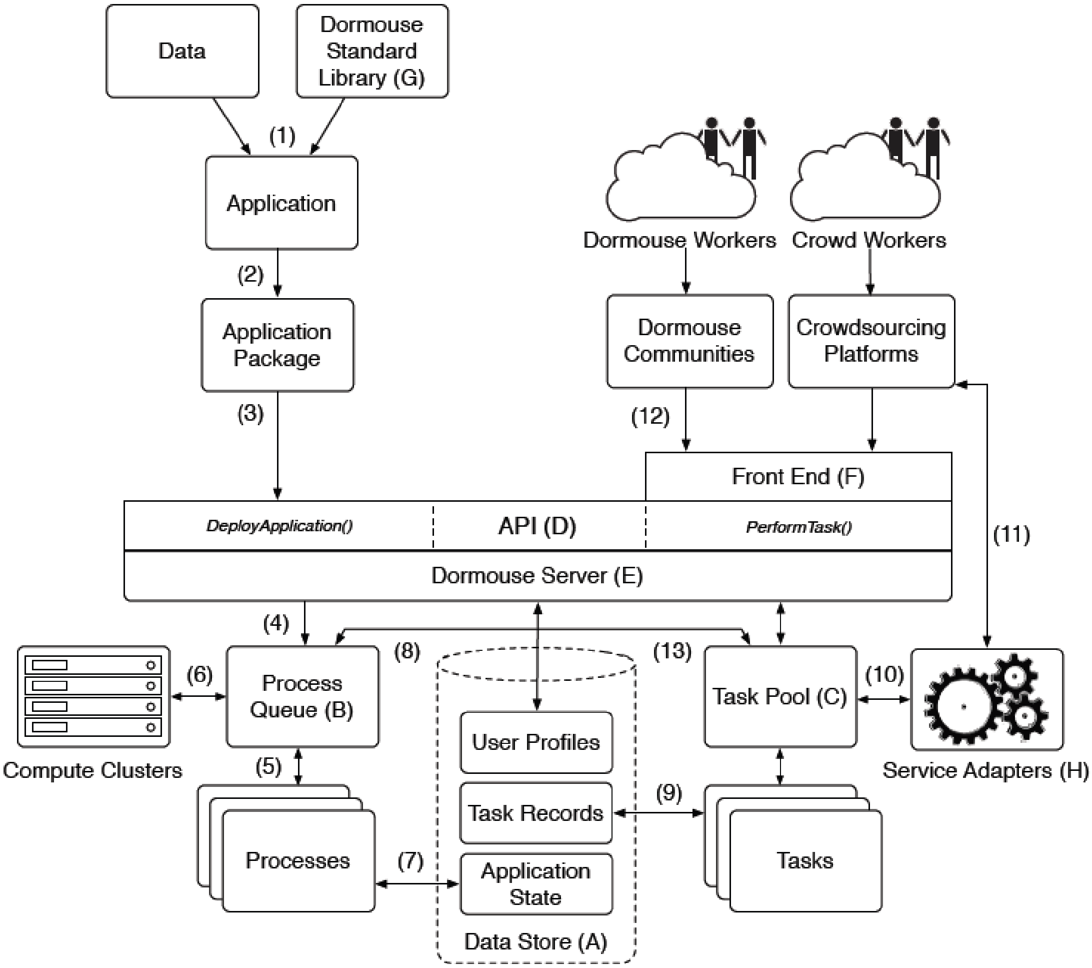


Figure 5: Internal Architecture of Dormouse [8]

CrowdLang (University of Zurich)

It is a general-purpose computation programming language and framework for interweaving human and machine computation within complex problem solving.

The CrowdLang framework architecture consists of three main components: the IDE, the Engine and Integrator.

The CrowdLang IDE – provides a source code editor for the design of human computation algorithms;

The CrowdLang Engine – responsible for managing the execution of CrowdLang algorithms;

The CrowdLang Integrator – builds the interface to different human workforce crowds and machine operators in the cloud.

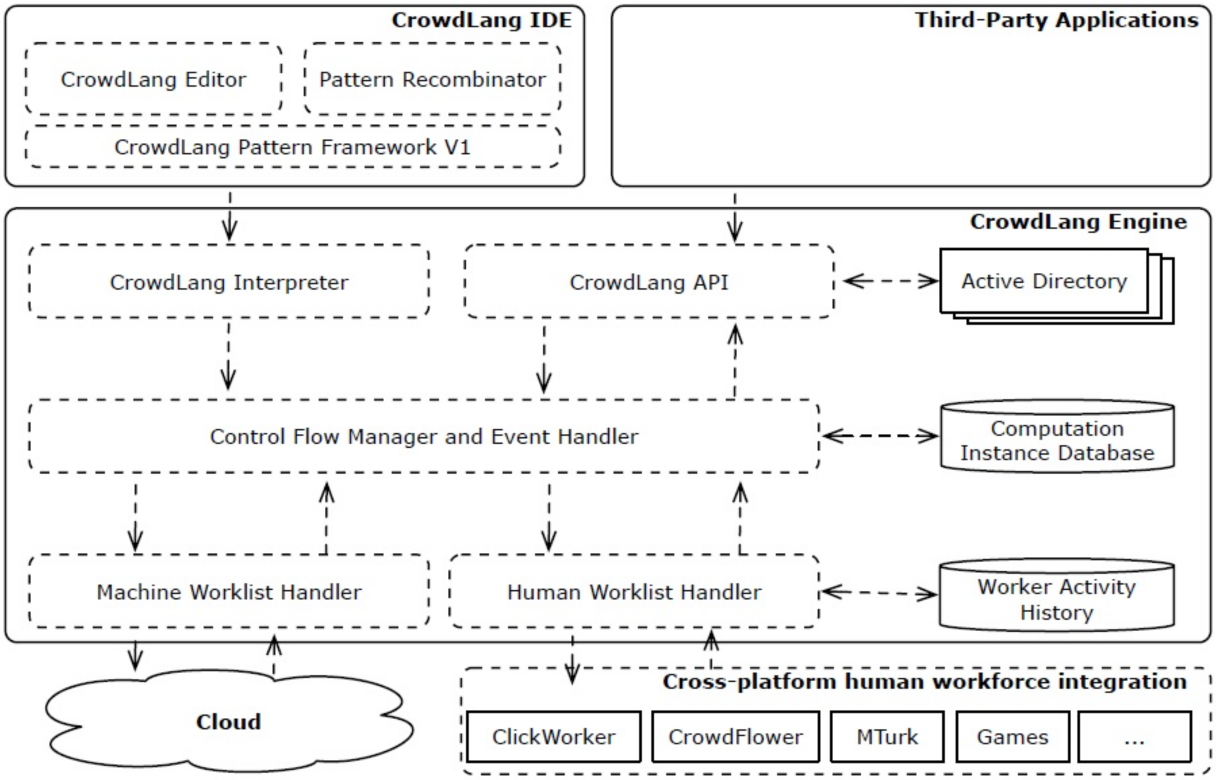


Figure 6: Architecture of CrowdLang [9]

Framework Comparisons

The objectives of this graduation project are to deliver the tool which would solve inefficiency and inconformity problems as well as meet the needs for configurability, scalability and extendibility.

Based on the required objectives, the intern set up multiple criterions with different priorities to compare the two investigated frameworks, in order to determine whether these frameworks could (partially) fulfill demands.

Uniformity and Efficiency are the high priority criterions since they are the bases for the Crowd-Watson team to run current research experiments properly. Configurability, Scalability and Extendibility are the medium priority criterions which are required by the Crowd-Watson team to support further extensive research.

Criterions	Explanations	Jabberwocky	CrowdLang
Uniformity (High Priority)	Interaction with the various crowdsouring platforms;	• Dormouse enables cross-platform programming language; • Easily combine crowdsourcing platforms or create new ones; • The Dormouse Service Adapters(H) manage communication between Dormouse and other crowdsourcing platforms;	• CrowdLang supports cross-platform workforce integration, the management of human computer latency, and incorporates abstractions for group decisions, contests and collaborative interaction patterns; • The CrowdLang Integrator integrates different execution platforms such as micro task markets and games with a purpose;
Efficiency (High Priority)	Setting up large-scale experiments;	• Dormouse has a straightforward mechanism to create new templates for human tasks and to reuse those created by others; • Either people or machines can perform the work necessary for both ManReduce map and reduce steps; • Machine map and reduce steps can call functions from existing Ruby libraries and custom	• Complex mixtures between crowds and machines can be constructed rather than just having them emerge by using CrowdLang; • Easily decompose problem into executable process maps respectively algorithms; • Easily evaluate different research topics such as predicting quality, evaluating the influence of different

		libraries; • A large number of library functions that express common human and machine functions; • Easily test out the control flow and human functions on a small set of people and data before deploying;	process configurations, as well as enhance the use of the crowd for sophisticated computation in business;

Configurability
(Medium Priority)
	Input;
	Workflow;
	Processing nodes;
	Processing;
	Output;
	View-templating;
		• ManReduce provides convenient programming abstractions for both human and machine computation;
		• Enable to route tasks based on personal properties and social structure;
		• Dormouse API contains functions to process and update Dormouse system resources;
		• The Task Pool uses Dormouse Service Adapters to invoke the target service's API to post tasks;
		• The built-in libraries, and the ability to define and share custom steps, provide a large and growing codebase with which to process input in various way;
		• Dog is designed to make it easy to express control flows that involve selecting people and specifying tasks for them to perform;
			• Executable model-based programming language that can be used to compute complex problems by combining humans in the crowd and machines;
			• CrowdLang is based on a small set of operators which can be aggregated to basic coordination patterns and the building block of collective intelligence;
			• Given a problem statement in the form of a task definition and required data items humans and machines do some computation;
			• Multiple flows such as data flow and sequence flow present the transformation from problem statements into solutions;

Scalability (Medium Priority)	Potentially scale up to a large number of tasks/interactions;	• Dog is designed to support sequential transformations on large-scale data, either by parallel human or machine functions; • ManReduce automatically serializes all worker answers, enabling to re-run a script many times, automatically	• For enabling more sophisticated problem-solving, CrowdLang provides a set of routing operators for distributing computation and aggregating results; • The computation of a complex problem can characterized by a sequence of tasks;

		using the available answers to compute new (or corrected) statistics without re-doing human tasks;	
Extendibility (Medium Priority)	The number of processing node types; Input/output types defined currently may be extended at a later stage;	• To hook into Dormouse, a crowdsourcing platform need only provide an API that minimally allows for the posting of tasks; • To introduce arbitrary dependencies between multiple map and reduce steps;	• For enabling more sophisticated problem-solving, CrowdLang provides a set of routing operators for distributing computation and aggregating results; • The computation of a complex problem can characterized by a sequence of tasks;

Conclusions

The intern has contacted both Jabberwocky and CrowdLang teams and was informed that their resources cannot be released during this graduation project. The initial analyses of these two frameworks are actually used to bring the intern inspirations of building efficient framework architecture. The intern put more efforts on the custom design of Crowd-Watson Framework, Architecture and data integration due to the lack of the libraries functions and computational algorithms.

Both frameworks not just have front-end GUI for user interactions and back-end server and database for data storage and integration, but also enable to hook with various crowdsourcing platforms simply through API calls. Based on that, the intern focuses on GUI design for user experiences, the realization of data storage and integration with the help of the server and database, proper connections among GUI, server and database for efficiency and uniformity, scripts to hook GUI with crowdsourcing platforms by API calls and overall architecture design meeting the needs for configurability, scalability and extendibility.

Based on the conclusions above as well as the contents, features and requirements of the Crowd-Watson project, the intern designed the high-level Crowd-Watson Framework and low-level Crowd-Watson Architecture.

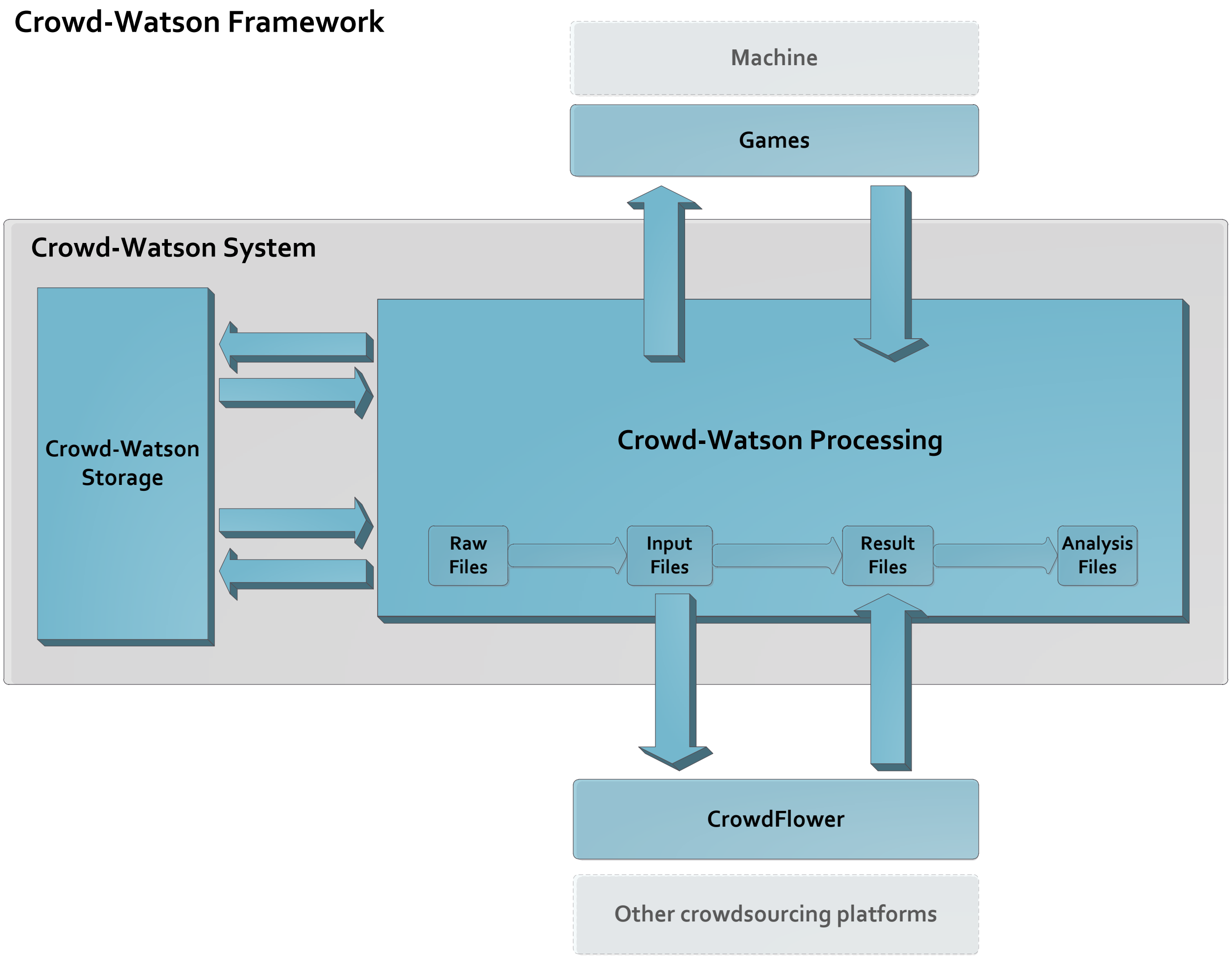


Figure 7: Crowd-Watson Framework

Crowd-Watson Framework (high-level) gives a general overview of how Crowd-Watson System operates internally and interacts with third-party apps externally.

Crowd-Watson Architecture (low-level) indicates the internal architecture of Crowd-Watson System in more details upon Crowd-Watson Framework.

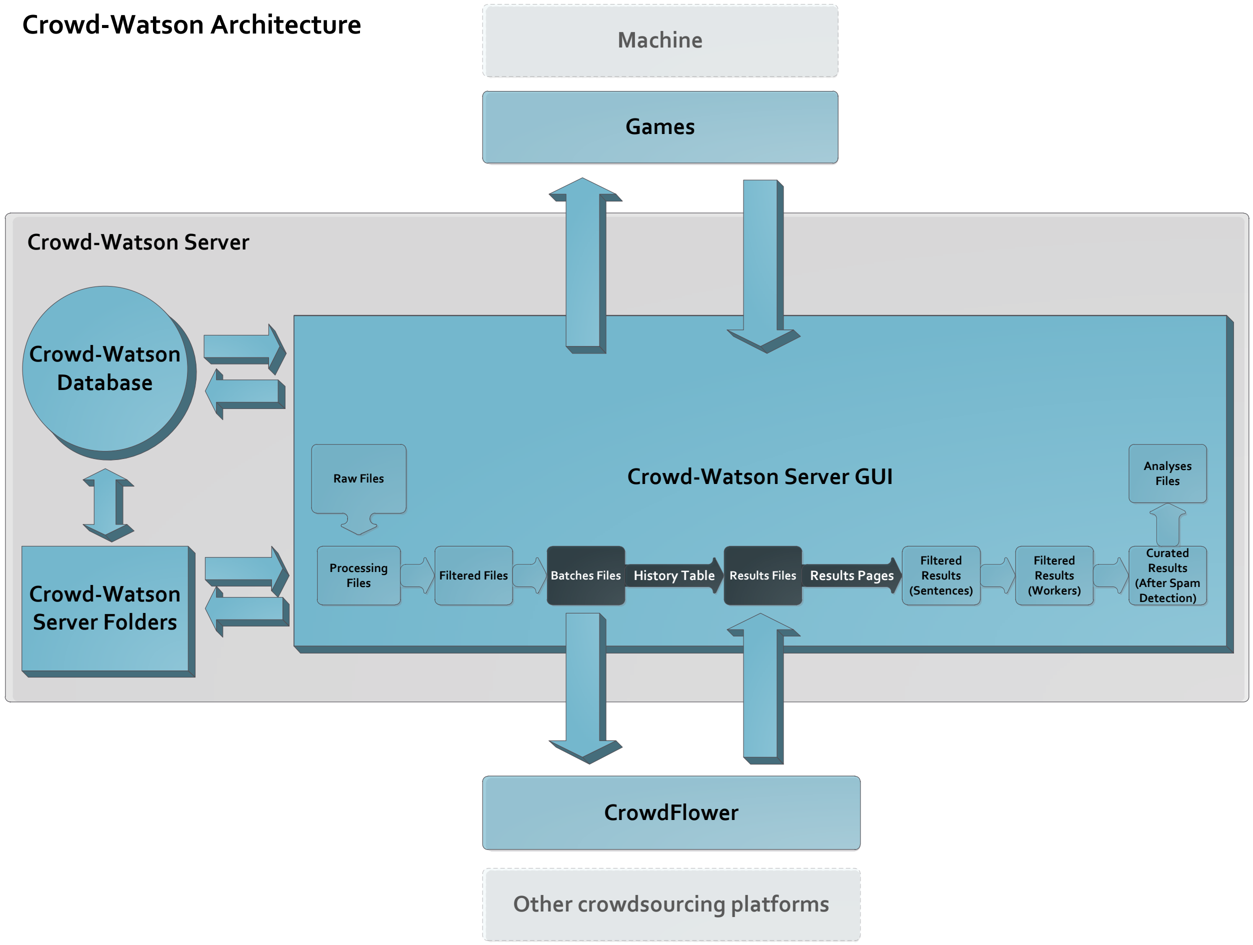


Figure 8: Crowd-Watson Architecture

1. Crowd-Watson Server

- IBM external server which enables LAMP development environment and project members' access
- Contains Crowd-Watson Server GUI, Crowd-Watson Server Folders and Crowd-Watson Database

2. Crowd-Watson Server GUI

- Allows users to upload, process, filter, download and analyze files
- Interacts with CrowdFlower and Games in order to create, track, modify and finalize jobs
- Interacts with Crowd-Watson Server Folders and Crowd-Watson Database in order to store and maintain data and files during each transaction

3. Crowd-Watson Server Folders

- Interacts with Crowd-Watson Server GUI and Crowd-Watson Database
- Every file output from Crowd-Watson GUI would be saved in the specific Crowd-Watson Server Folder and registered in Crowd-Watson Database with unique file ID and folder location
- Maintain all kinds of files and enable users to download files

4. Crowd-Watson Database

- Every file which is stored in Crowd-Watson Server Folders would be registered in the database with unique file ID and folder location
- Data of each transaction would be stored in the specific datatable and display the necessary ones in Crowd-Watson Server GUI such as the History Table and Results Pages
- Acts as the bridge between Crowd-Watson Server GUI and Crowd-Watson Server Folders when users want to access specific files

5. Interactions with CrowdFlower and Games

- Batch Files would be used to create new jobs through API (Application Programming Interface) Calls
- History Table not just indicates job parameters and completion through Webhook as well as enables to modify job statuses through API Calls
- Result Files would be retrieved from either CrowdFlower or Games through API Calls

6. Processing Flow

(From Raw Files to Analyses Files)

Orders	Explanations	Tab (GUI)
Pre-process	Raw Files -> Processing Files -> Filtered Files -> Batches Files (All the files are generated at the server and registered in the database)	Input
Create a job	A Batches File are selected from the server -> Fill in certain parameters such as judgments, payments, time and etc. -> Send the selected Batches File to CrowdFlower or other platforms -> The filled-in parameters and creating information would be saved in database and shown in History Table	Jobs

History Table	Acts like a control centre: - Connects input for creating jobs with output retrieved from finished jobs - Indicates integrated data - Indicates job completion - Manages job statuses - Enables files download - Enables to block spammers	History
Retrieve the Results File	Job completion message would be sent from the platform to GUI when the job is finished -> The Results File would be saved in the server and registered in the database -> Results parameters offered by the platform would be shown in History Table -> Results File can be downloaded through History Table	History
Post-process	Filtered Results (Sentences) -> Filtered Results (Workers) -> Curated Results (After Spam Detection) -> Analyses Files (All the files are generated at the server and registered in the database; Results parameters by the analytic analyses would be shown in History Table)	
View the analytic results	Select a Job ID or multiple Job IDs with Finished status from History Table and click on Analyze button -> Results Page will pop up, which contains Sentence Metrics, Worker Metrics, Spammers detected and etc.	History

7. Configurability, Scalability and Extendibility at the architecture level
(Please refer to 3.4 The Intern´s Responsibilities in the Crowd-Watson team)

Criterions	**Explanations**
Configurability	- Pre-process input based on various filters - Create a job with different Batches Files and parameters - Change the job status anytime before it’s finished - Make flexible selections of Job IDs with Finished status and make analyses

Scalability	Database can be scalable by the hardware; Server can be duplicated by using a load balancer; Software is stateless. Together, they enable a large number of tasks and interactions
Extendibility	- The architecture is built with the Web interface and consists of clear modules - Each tab allows users to operate a specific function such as processing input, creating jobs or assignments, tracking history and results and etc. - Within the same tab, each accordion offers the specified interface towards different processing nodes (CrowdFlower, Games and etc.) - New functions or processing nodes can be embedded into the GUI more easily with this design

4.2 Methodology

Agile Software Development (ASD) is a methodology for the creative process that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product. Agile software development focuses on keeping code simple, testing often, and delivering functional bits of the application as soon as they are ready. The goal of ASD is to build upon small client-approved parts as the project progresses, as opposed to delivering one large application at the end of the project.

Many practitioners agree that the heart of Agile Software Development is Agile Manifesto introduced in 2001 which includes:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contact negotiation

Responding to change over following a plan

The most popular agile methodologies include Extreme Programming, Scrum, Crystal, Dynamic Systems Development Method, Lean Development, and Feature-Driven Development. [10]

IBM has been adopting Scrum Agile Software Development practices for its projects. Scrum is simple, which can be understood and implemented in a few days. On a Scrum project, there are four main parts: People, Planning, Execution and Reflection.

People:

People can be divided into Product Owner, Scrum Master, and the Team. A Team consists of a group of software designers, programmers and testers, usually 5-9 members, but may scale much larger. A team is usually directed by a Scrum Master who is responsible for the management of the team and project. The Product owner acts like the client of the project. He defines the requirements, make adjustments, and accept or reject work results.

Planning:

The Scrum Planning occurs at the start of each sprint. Team and project owner negotiate the commitment, and then the team members begin the design process and generate tasks. Crowd-watson-dev regular meeting triggers the start of each sprint. Trello web application is used to manage the Scrum planning.

Trello is a free web-based project management application. Projects are represented by boards, which contain lists (task lists). Lists contain cards (tasks). Cards are supposed to progress from one list to the next (via drag-and-drop). Users can be assigned to cards. Users and boards can be grouped into organizations.

During Crowd-watson-dev regular meeting, new tasks are created in the To Do (Next Sprint) section on Trello board and assigned to different users (the Crowd-Watson members). Crowd-Watson meeting notes (created in Google Drive) are to track the overview of each Crowd-watson-dev regular meeting.

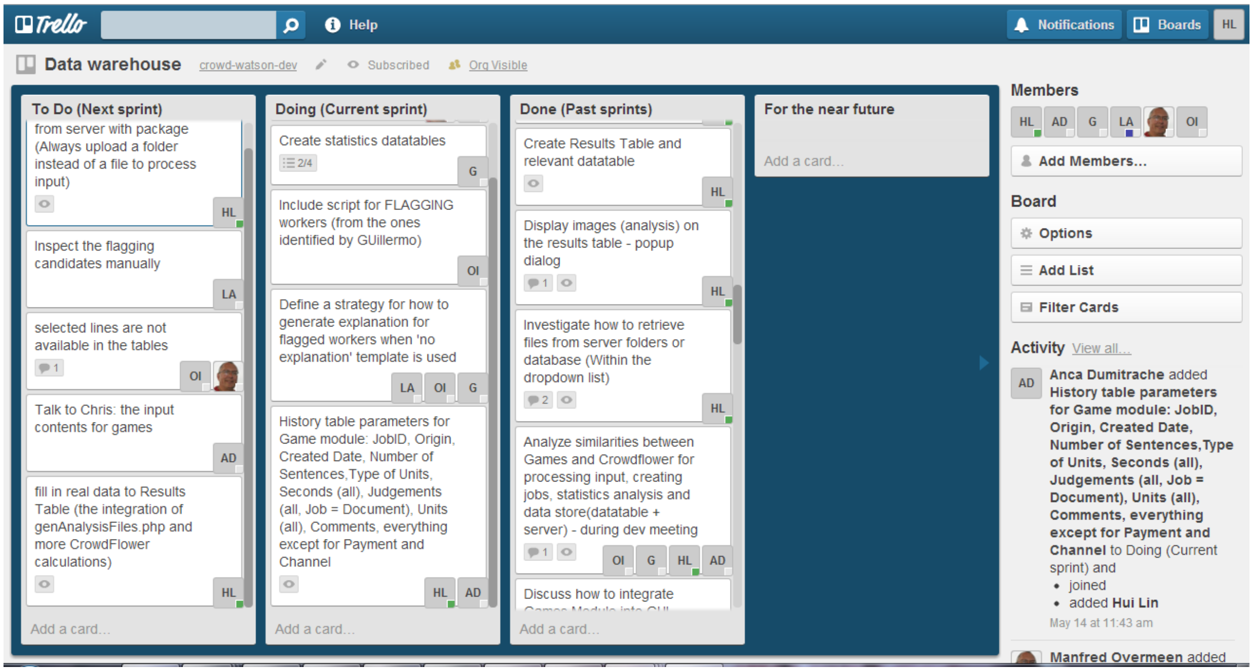
Execution:

A Scrum Project makes progresses in a series of sprints. The product is designed, coded, and tested during the sprint. The Scrum Execution involves tracking Trello, daily Google Chat, weekly Crowd-watson-dev meeting and regular email contacts.

The Crowd-Watson project is a joint program between Computer Science department of VU University Amsterdam and CAS Amsterdam of IBM Netherlands. Various net tools are used to facilitate regular contacts among team members in different places.

Team members use Trello to perform tasks, submit comments, update statuses and receive notifications or messages. Google Handout is used to have quick chats or leave prompt messages since all of members have Gmail accounts and stay available more often. Except for planning and creating new tasks, members discuss research topics or discoveries regarding to the Crowd-Watson project, keep track of project processes, demonstrate the current tool development made and make continuous process improvements. Members receive group emails for necessary notifications, plans, preparations or modifications.

Project developers use Github (a web-based hosting service for software development projects that use the Git revision control system) to fork, pull requests and merge codes. With the help of Putty (a free and open-source terminal emulator, serial console and network file transfer application), developers can access Crowd-Watson Server remotely to pull latest version of codes from Github to the server, to test on the server and to change permissions if it’s necessary. FileZilla which is free and cross-platform FTP (File Transfer Protocol) software is also often used to have quick tests on the server due to the graphical interface.



Data warehouse crowd-watson-dev Subscribed Org Visible

To Do (Next sprint)

- from server with package (Always upload a folder instead of a file to process input) — HL
- Inspect the flagging candidates manually — LA
- selected lines are not available in the tables — 1 — OI
- Talk to Chris: the input contents for games — AD
- fill in real data to Results Table (the integration of genAnalysisFiles.php and more CrowdFlower calculations) — HL

Add a card...

Doing (Current sprint)

- Create statistics datatables — 2/4 — G
- Include script for FLAGGING workers (from the ones identified by GUillermo) — OI
- Define a strategy for how to generate explanation for flagged workers when 'no explanation' template is used — LA OI G
- History table parameters for Game module: JobID, Origin, Created Date, Number of Sentences,Type of Units, Seconds (all), Judgements (all, Job = Document), Units (all), Comments, everything except for Payment and Channel — HL AD

Add a card...

Done (Past sprints)

- Create Results Table and relevant datatable — HL
- Display images (analysis) on the results table - popup dialog — 1 — HL
- Investigate how to retrieve files from server folders or database (Within the dropdown list) — 2 — HL
- Analyze similarities between Games and Crowdflower for processing input, creating jobs, statistics analysis and data store(datatable + server) - during dev meeting — 1 — OI G HL AD
- Discuss how to integrate

Add a card...

For the near future

Add a card...

Members

HL AD G LA OI

Add Members...

Board

Options

Add List

Filter Cards

Activity View all...

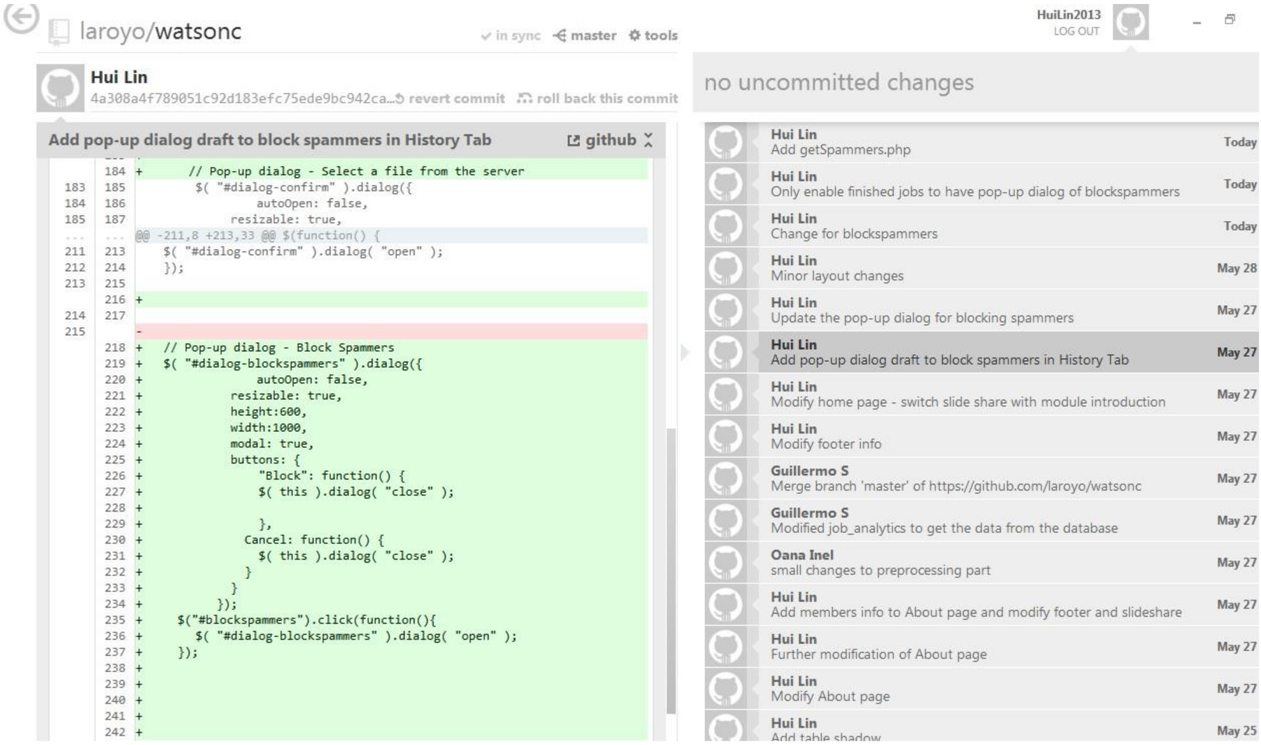
AD **Anca Dumitrache** added **History table parameters for Game module: JobID, Origin, Created Date, Number of Sentences,Type of Units, Seconds (all), Judgements (all, Job = Document), Units (all), Comments, everything except for Payment and Channel** to Doing (Current sprint) and

- joined
- added **Hui Lin**

May 14 at 11:43 am

Manfred Overmeen added

Figure 9: Use Trello to help performance tasks of team members



Hui Lin
4a308a4f789051c92d183efc75ede9bc942ca... revert commit roll back this commit

Add pop-up dialog draft to block spammers in History Tab github

```
+       // Pop-up dialog - Select a file from the server
         $( "#dialog-confirm" ).dialog({
                autoOpen: false,
            resizable: true,
@@ -211,8 +213,33 @@ $(function() {
     $( "#dialog-confirm" ).dialog( "open" );
     });

+

-
+    // Pop-up dialog - Block Spammers
+    $( "#dialog-blockspammers" ).dialog({
+                autoOpen: false,
+            resizable: true,
+            height:600,
+            width:1000,
+            modal: true,
+            buttons: {
+                "Block": function() {
+                $( this ).dialog( "close" );
+
+                },
+              Cancel: function() {
+                $( this ).dialog( "close" );
+              }
+            }
+        });
+    $("#blockspammers").click(function(){
+       $( "#dialog-blockspammers" ).dialog( "open" );
+    });
+
+
+
+
```

no uncommitted changes

Hui Lin	Add getSpammers.php	Today
Hui Lin	Only enable finished jobs to have pop-up dialog of blockspammers	Today
Hui Lin	Change for blockspammers	Today
Hui Lin	Minor layout changes	May 28
Hui Lin	Update the pop-up dialog for blocking spammers	May 27
Hui Lin	Add pop-up dialog draft to block spammers in History Tab	May 27
Hui Lin	Modify home page - switch slide share with module introduction	May 27
Hui Lin	Modify footer info	May 27
Guillermo S	Merge branch 'master' of https://github.com/laroyo/watsonc	May 27
Guillermo S	Modified job_analytics to get the data from the database	May 27
Oana Inel	small changes to preprocessing part	May 27
Hui Lin	Add members info to About page and modify footer and slideshare	May 27
Hui Lin	Further modification of About page	May 27
Hui Lin	Modify About page	May 27
Hui Lin	Add table shadow	May 25

Figure 10: Use Github to fork, pull requests and merge codes

4.3 Implementation

To implement the new Crowd-Watson crowdsourced text annotations tool, certain tooling and programming environments are set up.

Programming Languages	PHP, jQuery, jQuery UI, MySQL, HTML, CSS, JavaScript, Ajax and Java
Development Tools	IBM Cloud Server, Eclipse, FileZilla, Putty, Github, and IBM Lotus Symphony
Conducting Environments	IBM HQ Amsterdam – software development; VU University Amsterdam (Web Sciences department) – regular meetings and discussions;

5. Project Execution

The Project Execution chapter describes how the intern's graduation project (the build-up of Crowd-Watson Crowdsourced Text Annotations tool) is executed during various sprints by specifying the conduction period, user requirements, result demonstrations and modifications or exceptions.

The Crowd-Watson project is a long term project involving with multiple members and various module durations. The build-up of Crowd-Watson Crowdsourced Text Annotations tool itself would be split into five sprints from 1st February to 29th June with months as the units due to the fact that the Crowd-Watson project is a university research project and does not specify sprints clearly like normal company projects. Each sprint holds several weekly meetings to review previous conducted tasks, discuss issues, request modifications, show user requirements for next sprint and assign new tasks.

5.1 Sprint 1

(1st Feb – 28th Feb)

Sprint 1 is conducted from 1st February to 28th February. During this period, the focus is to get familiar with the project environment, to study existing components from the Crowd-Watson project and to investigate on research papers.

Investigate Jabberwocky and CrowdLang

- Read and understand several research papers from Stanford University and University of Zurich

- Find out and study CrowdLand student assignment from University of Zurich and try to contact the professor for more details and available resources offering

- Analyze the added values of two frameworks and make a framework comparison report

-Try to implement the analyzed frameworks theory to the Crowd-Watson project by making some analysis diagrams

Study existing components from CrowdFlower (CF) and Statistics modules

- Read module description documents

- Try to understand existing scripts

- Analyze data storage and transfer between these two modules and make Data Flow Diagram and Data Flow Control description document

- Test existing components with IBM software development environment

Self-study programming language

To understand existing scripts of CrowdFlower and Statistics modules, the intern self-studied some programming language information especially R programming. R is a free software programming language for statistical computing and graphics. Furthermore, programming examples of javaScript, jQuery, CSS, AJAX and PHP were reviewed.

5.2 Sprint 2

(1st March – 31st March)

Sprint 2 is conducted from 1st March to 31st March. During this period, the intern started to implement initial GUI based on designed Crowd-Watson Framework and Architecture. However, it took some time to arrange back-end server and LAMP development environment to enable further development of GUI.

Design Crowd-Watson Framework and Architecture

Both Jabberwocky and CrowdLang resources cannot be released during this graduation project. The initial analyses of these two frameworks are actually used to bring the intern inspirations of building efficient framework architecture (Refer to Chapter 4 Research for more details). Therefore, the intern put more efforts on the custom design of Crowd-Watson Framework and Architecture and data integration.

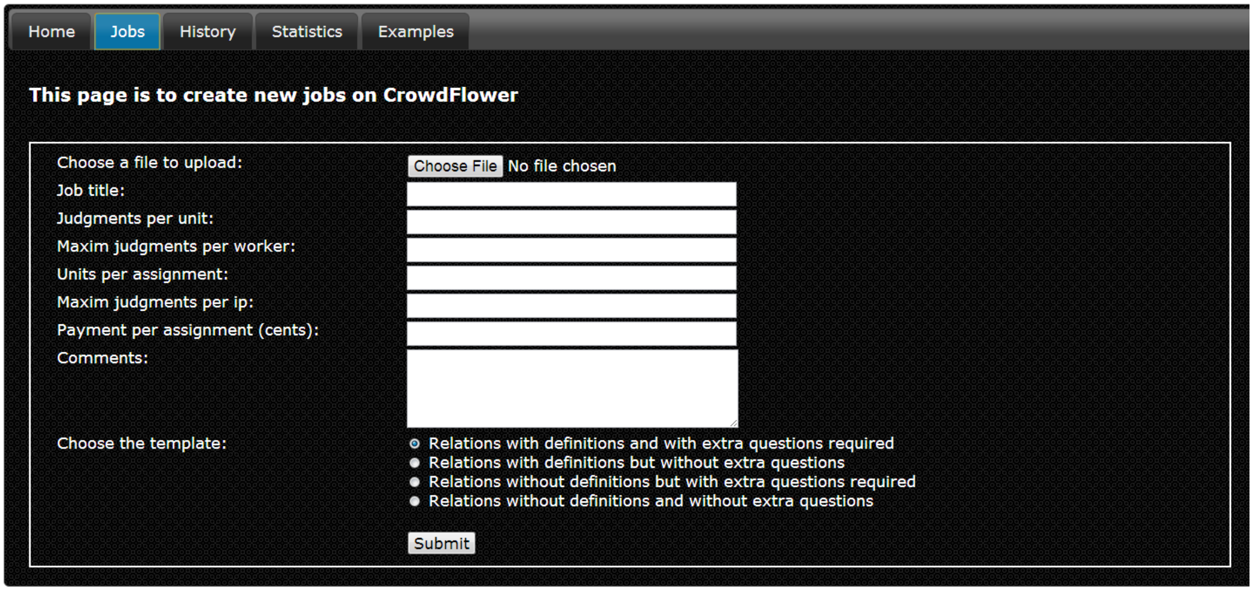
Create initial GUI with CrowdFlower module integrated

The Crowd-Watson project is used to research the Crowdsourcing for Watson. Crowdsourcing is an online, distributed problem-solving and production model. Based on this speciality, the intern designed and implemented the Web GUI with PHP, jQuery, MySQL, HTML, CSS, javaScript, Ajax and Java. The initial GUI had CrowdFlower module integrated, which users can upload a file from local to create a new job for CrowdFlower through API call by using this GUI. Relevant parameters of the newly-created job will be stored in the back-end database and shown in History tab of GUI.

Arrange Crowd-Watson Server and configure LAMP development environment

When initial GUI was created, it could not function properly due to the lack of a proper back-end server. The Crowd-Watson team tried out several servers such as IBM stift 7, eculture.cs.vu.nl and eculture2.cs.vu.nl but none of them was suitable to support the development and run of GUI. Only IBM employees are entitled to access IBM stift 7 and IBM internal firework blocks Web call-back from CrowdFlower. Both VU servers lack of LAMP development environment (MySQL and phpMyAdmin are disabled).

It took some time to arrange a proper server which could solve previous problems due to IBM policy. After Crowd-Watson Server was set up properly, LAMP environment was configured as well.



This page is to create new jobs on CrowdFlower

Choose a file to upload: No file chosen
Job title:
Judgments per unit:
Maxim judgments per worker:
Units per assignment:
Maxim judgments per ip:
Payment per assignment (cents):
Comments:

Choose the template:
◉ Relations with definitions and with extra questions required
○ Relations with definitions but without extra questions
○ Relations without definitions but with extra questions required
○ Relations without definitions and without extra questions

Figure 11: Initial version of GUI

5.3 Sprint 3

(1st April – 30th April)

Sprint 3 is conducted from 1st April to 30th April. During this period, the intern mainly focused on the further development of GUI (updating History table, integrating Pre-processing and Statistics modules and creating Results table). The intern also gained experience by dealing with relatively frequent requests of modifications during this sprint (Please refer to 5.6 Issues and Requirement Changes). For example, one of the cases is newly created Results Table was requested to be removed.

Integrate Pre-processing module

Process Input tab was created to integrate Pre-processing module. Only files with certain formats are suitable to be used as input files for jobs creations of CrowdFlower. Pre-processing module is aimed to configure raw files with defined format requirements. Raw files are processed by back-end .jar scripts previously created by Oana through GUI

Update History table

- Enable Webhook from CrowdFlower so that users can check job completion and change statuses which can be synchronized among front-end GUI and back-end server and database
- Improve user experiences:

  User abbreviations for headers to reduce the table size;

  Enable columns to be resized;

  Enable columns to be re-ordered;

  Make the header fixed so that users can always view it when scrolling the web page;

  Users can hide and show columns;

  Format rows' size;
- Enable users to download all files generated in GUI through the hyper links in History table

History table is still being requested to modify due to the newly-discovered user needs.

Integrate Statistics module with Results table created

After a job is completed at CrowdFlower, its results file can be extracted from CrowdFlower automatically and Statistics scripts can be run afterwards to generate Statistical results (parameters, files and images) in Results table (in Results tab of GUI). Results table is closely connected to History table by the job id.

After the Results table was created, the team decided to move some parameters from Results table to History table, to remove the Results table and to generate statistical results in a dynamical way. This modification would be conducted during Sprint 4.

Discuss with team members for the integration of Games module

The intern arranged several student group meetings to discuss together about the integration of Games module, especially to find out similarities between Games and CrowdFlower modules for processing input, creating jobs, Statistics analysis and data storage (the database and server). It is expected that the intern can start to integrate parts of Games module during the next sprint.

Create initial version of Graduation Report

Due to the graduation deadline of Fontys, the intern started to create the initial version of the final report for feedback.

5.4 Sprint 4

(1st May – 31st May)

Sprint 4 is conducted from 1st March to 31st March. During this period, main functions of GUI required by this graduation project were somewhat completed. The layout and style of GUI were further modified to have more official interface. The general information was added to Home and About pages and Website footer since Crowd-Watson Crowdsourced Text Annotations tool will be released on Github as open resource software.

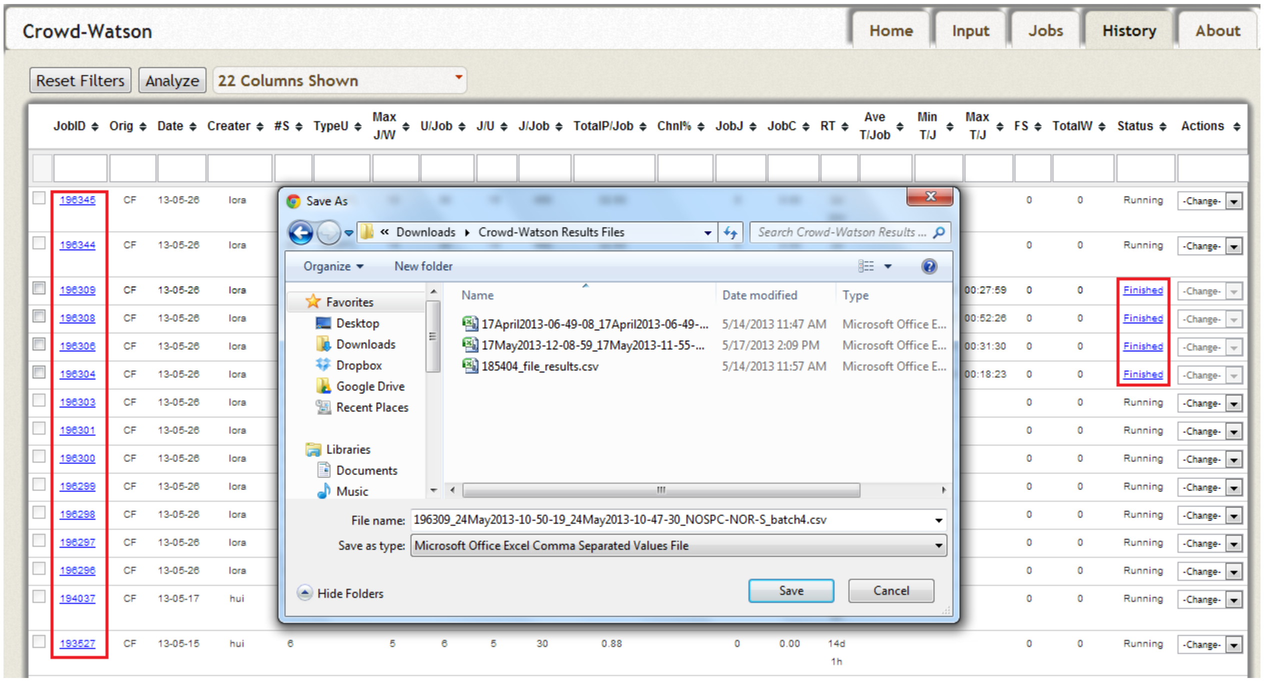
Further development on required main functions of GUI

Due to the requirement of the last sprint, Results table would be removed and the parameters from it need to be moved to History table. Users can make flexible selections of a job id or several job ids to generate or view statistical results dynamically in the pop-up Results pages, after clicking on the Analyze button.

Users can download both batch files (through Job ID link) and results files (jobs with Finished Status) from History Table.

When hiding columns, History Table size can be reduced (Initially, the table size stayed the same and only columns' sizes were getting bigger to fill in the extra space). Check all and uncheck all options were integrated into Hide/Show columns drop down list so that users can hide or show columns more conveniently.

Back to top button was added to each page as an alternative for users instead of scrolling the Web page to the top. It's especially useful when users view through a large amount of data in History Table.



	JobID	Orig	Date	Creater	#S	TypeU	Max J/W	U/Job	J/U	J/Job	TotalP/Job	Chnl%	JobJ	JobC	RT	Ave T/Job	Min T/J	Max T/J	FS	TotalW	Status	Actions
□	196345	CF	13-05-26	lora															0	0	Running	-Change-
□	196344	CF	13-05-26	lora															0	0	Running	-Change-
□	196309	CF	13-05-26	lora														00:27:59	0	0	Finished	-Change-
□	196308	CF	13-05-26	lora														00:52:26	0	0	Finished	-Change-
□	196306	CF	13-05-26	lora														00:31:30	0	0	Finished	-Change-
□	196304	CF	13-05-26	lora														00:18:23	0	0	Finished	-Change-
□	196303	CF	13-05-26	lora															0	0	Running	-Change-
□	196301	CF	13-05-26	lora															0	0	Running	-Change-
□	196300	CF	13-05-26	lora															0	0	Running	-Change-
□	196299	CF	13-05-26	lora															0	0	Running	-Change-
□	196298	CF	13-05-26	lora															0	0	Running	-Change-
□	196297	CF	13-05-26	lora															0	0	Running	-Change-
□	196296	CF	13-05-26	lora															0	0	Running	-Change-
□	194037	CF	13-05-17	hui															0	0	Running	-Change-
□	193527	CF	13-05-15	hui	6		5	6	5	30	0.88		0	0.00	14d 1h				0	0	Running	-Change-

Name	Date modified	Type
17April2013-06-49-08_17April2013-06-49-...	5/14/2013 11:47 AM	Microsoft Office E...
17May2013-12-08-59_17May2013-11-55-...	5/17/2013 2:09 PM	Microsoft Office E...
185404_file_results.csv	5/14/2013 11:57 AM	Microsoft Office E...

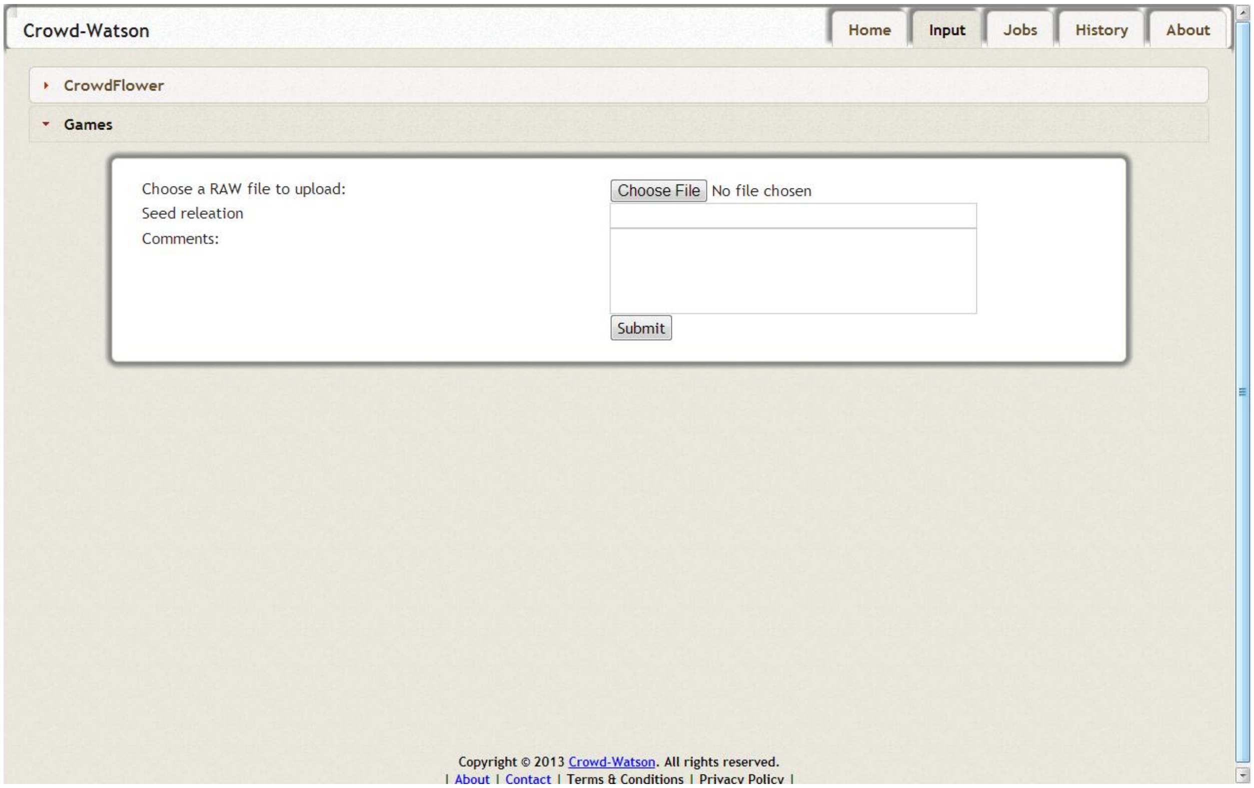
File name: 196309_24May2013-10-50-19_24May2013-10-47-30_NOSPC-NOR-S_batch4.csv

Save as type: Microsoft Office Excel Comma Separated Values File

Figure 12: Download batch files (by JobID link) and results files (for Finished jobs)

Integrate the input section for Games module

Due to the group discussion, the input section for Games module was integrated to GUI. By default, the raw file for Games module will be registered in the database and stored in the server after uploading. Further processing of the raw files will depend upon the new research results of Games module.



Crowd-Watson

Choose a RAW file to upload: Choose File No file chosen

Seed releation

Comments:

Copyright © 2013 Crowd-Watson. All rights reserved.
| About | Contact | Terms & Conditions | Privacy Policy |

Figure 13: Integrate the input section for Games module

Modify the layout and style of GUI for more official interface

- Synchronize colors for Web site theme

- Add box-shadow effect for headers, containers and tables

- Centralize texts and images

- Add the title to Web header and adjust tabs float to the right

- Make consistent spaces among the Web components

Add general information to GUI

Crowd-Watson Crowdsourced Text Annotations tool will be released on Github as open resource software in the end. Some general information is necessary to be added to GUI for the release.

- Add a slideshow of Harnessing Disagreement in Crowdsourcing presentation (by Lora Aroyo) to indicate general details of the Crowd-Watson project

- Introduce the Crowd-Watson project members in About page
- Create the sticky footer with copyright and contact information for each page

5.5 Sprint 5 and Closing

(1st June – 29th June)

At the time of the submission of this report, Sprint 5 has just started. Sprint 5 and Closing will be conducted until the end of the graduation internship. The following activities are planned for this period.

Implement the function to block spammers

After a job is completed, workers who performed the job would be analyzed by the Statistics module in order to filter out spammers. The data will be shown in FW column of History Table. When users click on FW data relative to a particular job id from History Table, a dialogue would pop up to indicate found spammer ids and enable users to click on the checkboxes to block spammers. The blocked spammers would be registered in the database and prevented from creation of new jobs.

Create descriptions for Web pages

The intern will create descriptions of four main components of Crowd-Watson GUI and place them on the home page. The four components are input data, jobs to run, history of jobs and analytics of jobs.

Final testing and clean-up

- Debug and test all the components of GUI
- Organize database and codes (delete redundant sections, make necessary comments, make codes tidy and more readable and etc.)
- Create Readme file (with explanations about what has been implemented and clear divisions of different components)

5.6 Issues and Requirement Changes

This section lists some examples of how to track the modifications or exceptions, generated problems, possible solutions, potential impacts, agreements, actions and recorded date.

Date	Modifications or Exceptions	Generated Problems	Possible Solutions	Potential Impacts	Client Agreements	Actions
03-04-2013	Jabberwocky and CrowdLang resources are not available;	Time spent for the investigations about the actual use of the frameworks resources; Machine module is especially effected;	Find out other solutions for machine module ;	Reinvestigation time requested; Less time left for other implementations; May affect the high-priority implementation and testing;	Wait for the resources until they are available; Not necessary to use those resources during this stage;	The intern would not use resources of Jabberwocky and CrowdLang for architecture implementations;
18-04-2013	Games module is still during the researching stage; The real implementation is expected to start in the early May of 2013 and will last until August;	Not possible to have the whole integration of Games module;	Integrate parts of Games module;	Games module of Crowd-Watson Crowdsourced Text Annotations tool is not complete;	Anca cannot start the real implementation immediately;	Only parts of Games module can be integrated into GUI;
25-04-2013	GUI theme was requested changed from Dark Hive to Pepper Grinder;	Extra time required to modify, test, debug and adjust;	The request can be conducted since it is not that time consuming;	Extra time required;	Prefer GUI theme to be changed to Pepper Grinder;	GUI theme would be changed to Pepper Grinder;

25-04-2013	Move some parameters from Results table to History table; Remove the Results Table; Generate statistical results in a dynamical way;	Time consuming to change; Need to make assumptions about what the client really needs;	Discuss with Guillermo for the possible ways;	Quite some extra required; May disable existing functions; Testing and debugging are necessary;	Place these modifications as current high priority;	Focus on these modifications ; Work during the weekend and holidays to catch up;
08-05-2-13	Official Machine module requirements have not been set up by the client;	Limited time left to set up detailed requirement by the client and to investigate implement solutions by the intern;	Crowd-Watson Architecture leaves the space for the extension of processing-nodes (Accordion mode);	Move the implementation of Machine module to W-would have section of MoSCoW List;	Research in Machine module was moved to next period;	Machine module would not be implemented during this graduation project due to the client decisions;
28-05-2013	Add the new requirement of downloading files in package based on flexible selections of categories (such as download all batch files, results files, files from today or etc.);	The intern needs to focus on the writing of final report and the preparations of final presentation and defense; Normally, the last period of the graduation project is used for final testing, arranging codes and documentations;	Discuss with Manfred the impact and possible solutions;	Not enough time left for the implementation; May delay graduation preparation;	Implement it during next development stage;	Focus on the graduation preparation;

28-05-2013	Since multiple developers work on the Crowd-Watson project, individual changes may disable existing functions;	Extra time required to find out issues and to make adjustments;	Communications with each other; Always keep the backup;	Extra time required; May affect normal implementation;	Better communications; Inform each other of the changes on time;	Communicate with each other in advance; Be more careful; Extra time spent if it really occurs;

6. Results

This chapter shows the results obtained by the intern during the graduation period. The assignment of the graduation project is to build a crowdsourced text annotations tool with the combination of crowdsourcing and automation modules, researching on the framework, creating the architecture and making the final integration. The delivery of the tool would solve inefficiency, inconformity and lack of data storage and integration problems encountered by the Crowd-Watson team. It would support further more extensive research.

As mentioned in Chapter 3, Crowd-Watson is a project involving multiple group members for different roles and responsibilities during the project execution. Oana Inel, Guillermo Soberon and Anca Dumitrache focus on researching different modules and deliver relevant research output. The intern is responsible for research in the framework, create the architecture and make the final integration. Therefore, the intern will specify clearly what her final deliveries are regarding to the build-up of the crowdsourced text annotations tool in the following table.

The intern will also use the criterions which are the objectives of this graduation project to match the results delivered by the intern in the table below.

Criterions:

Efficiency – Ef; Uniformity – U; Configurability – C; Extendibility – Ex; Scalability – S;

(At the GUI level)

	Hui Lin		**Oana Inel**	**Guillermo Soberon**	**Anca Dumitrache**
Research	Investigate and compare Jabberwocky and CrowdLang frameworks;		Research in CF module;	Research in Statistics module;	Research in Games module;
	Design Crowd-Watson Framework and Architecture; *(Fig. 7 and Fig. 8)*				
Database	Build the initial version of database;			Extend the database regarding to Statistics module;	
Implementation	Implement the designed architecture (with PHP,	Integrate preprocessing scripts of CF module into GUI *(Fig. 14)*: - Connect this section to database and server for data storage and	Create preprocessing scripts of CF module		

	JavaScript, jQuery, jQuery UI, HTML, CSS, Ajax and MySQL);	integration; (U, S and Ef) - Generated files would be registered in database and saved in server as input files for job creation of CF module; (Ef, C and S)	(with Java, PHP and HTML);		
		Integrate job creation scripts of CF module into GUI *(Fig. 16)*: - Connect this section to database and server for data storage and integration; (U, S and Ef) - Enable users to select files from the server to create jobs in the pop-up dialogue (with columns-based sorting and filtering functions) *(Fig. 15)*; (Ef and S)	Create job creation scripts of CF module (with PHP and HTML);		
		Integrate Statistics scripts into GUI: - Enable to make flexible selections of a job id or several job ids to generate or view statistical results dynamically in the pop-up Results pages, after clicking on Analyze button in History page *(Fig. 25)*; (C, S, Ex, C, U and Ef)		Create Statistics scripts (with R, PHP and MySQL): - Connect this section to database and server for data storage and integration;	
		Create the input section of Games module: - Raw files for Games module can be registered in the database and stored in the server *(Fig. 13)*; (Ef, C and S)			Specify the requirements of input section of Games module;

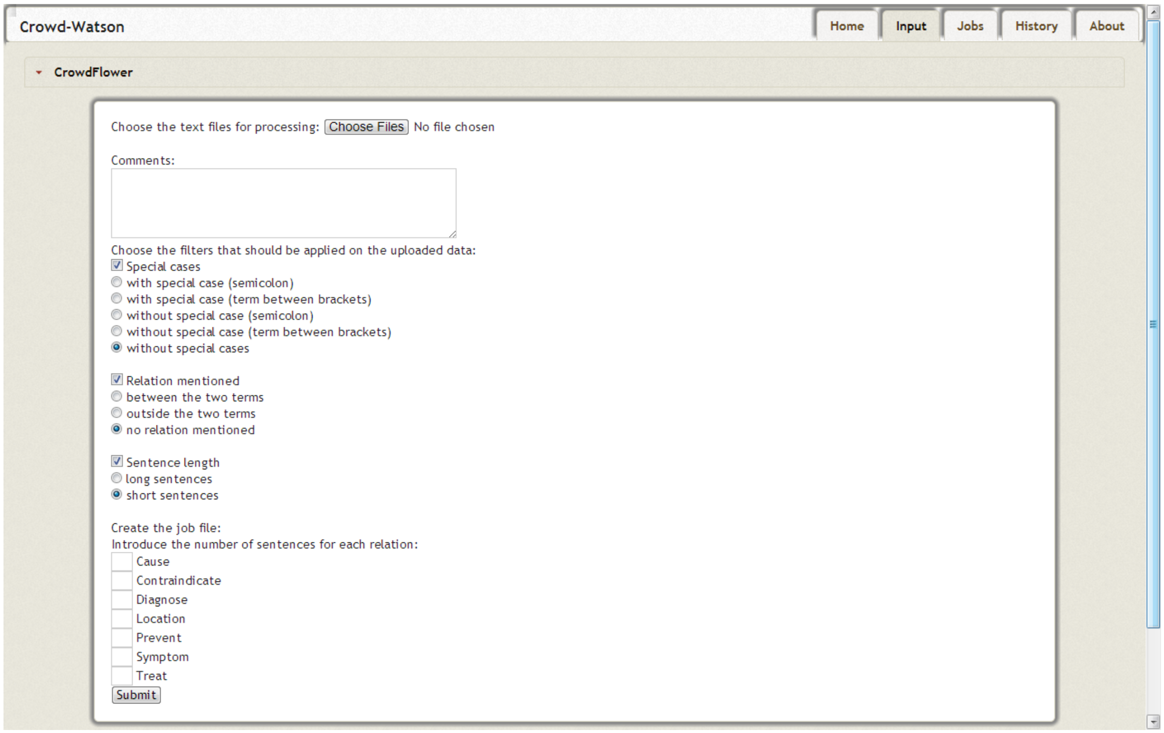
		Create History Table: - Indicate integrated data; - Track parameters, statuses and results of jobs *(Fig. 22)*; (Ef, U, Ex and S) - Enable to change status of jobs and synchronize the changes in the database and origin *(Fig. 19, Fig. 20 and Fig. 21)*; (Ef, U and S) - Indicate the completion of each job *(Fig. 22)*; (Ef) - Enable to download both batch files (through Job ID link) and results files (jobs with Finished Status) *(Fig. 12)*; (Ef, U and S) - Enable to block spammers after clicking on Filtered Workers data of a particular finished job *(Fig. 23)*; (Ef, C and U) - Enable to make flexible selections of a job id or several job ids to generate or view statistical results dynamically in the pop-up Results pages, after clicking on Analyze button *(Fig. 25)*; (C, S, Ex, C, U and Ef) - Enable to sort and filter many types of data including linked data in a cell without page refreshes, based on different columns *(Fig. 22)*; (Ef and S)			

		User Experience Design (UED): - Each tab allows users to operate a specific function; Within the same tab, each accordion offers the specified interface towards different processing nodes *(Fig. 13)*; (U, C, Ex and S) - Back to top button was added to each page to facilitate users to scroll the Web page to the top immediately. It’s especially useful when users view through a large amount of data in History Table *(Fig. 22)*; (Ef) - Use abbreviations for headers to reduce the table size *(Fig. 22)*; - Enable table columns to be resized; (Ef and Ex) - Enable table columns to be re-ordered; (Ef and Ex) - Make the table header sticky on the top so that users can always view it when scrolling the web page *(Fig. 22)*; (Ef, Ex and S) - Create the Web footer sticky on the bottom whatever the height of Web page is; - Users can hide and show columns of History Table *(Fig. 24)*; (Ef) - Synchronize colors for Web site theme;			

		- Add box-shadow effect for headers, containers and tables; - Centralize texts and images; - Add the title to Web header and adjust tabs float to the right; - Make consistent spaces among the Web components;			
		Add general information: - Add a slideshow of Harnessing Disagreement in Crowdsourcing presentation (by Lora Aroyo) to indicate general details of the Crowd-Watson project *(Fig. 26)*; - Introduce the Crowd-Watson project members in About page *(Fig. 27)*; - Add copyright and contact information in the sticky footer for each page;			

Testing and Debugging (TD)	Integration TD;
	Final TD;
			CF Module TD;	Statistics Module TD;	

Documentations	GUI descriptions;
	Readme file;
			CF Module descriptions and Readme file;	Statistics Module descriptions and Readme file;	Games Module descriptions;



Choose the text files for processing: Choose Files No file chosen

Comments:

Choose the filters that should be applied on the uploaded data:

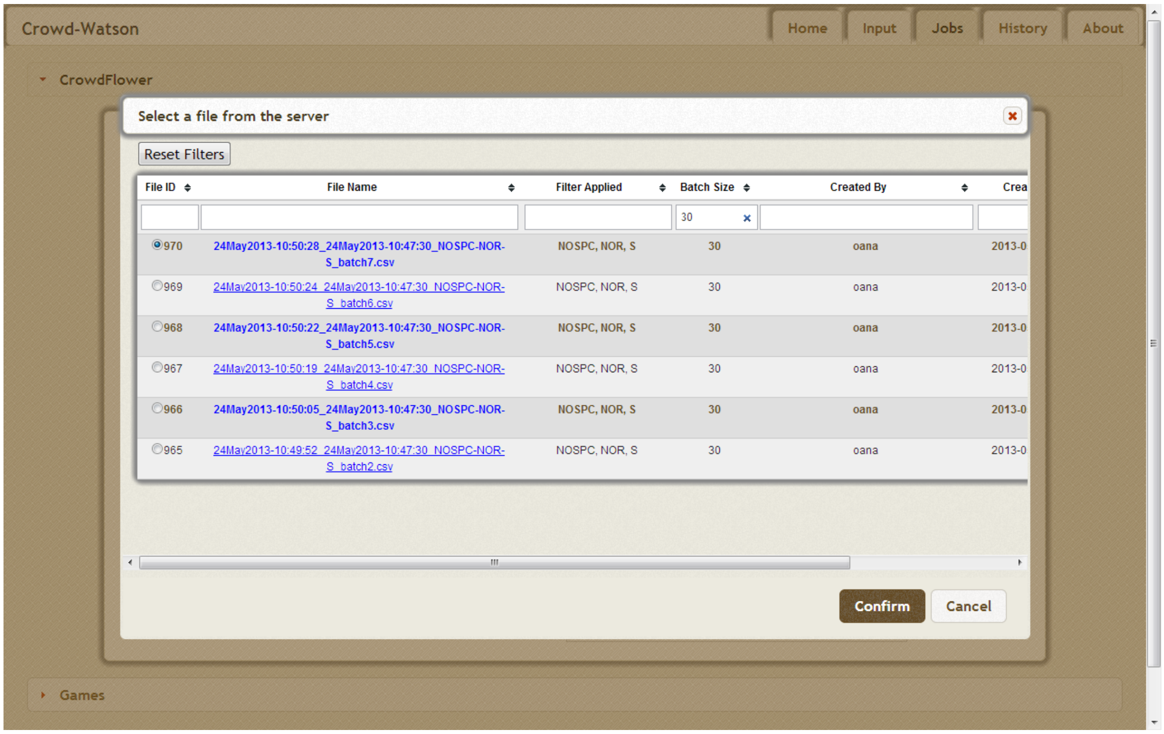
☑ Special cases
○ with special case (semicolon)
○ with special case (term between brackets)
○ without special case (semicolon)
○ without special case (term between brackets)
● without special cases

☑ Relation mentioned
○ between the two terms
○ outside the two terms
● no relation mentioned

☑ Sentence length
○ long sentences
● short sentences

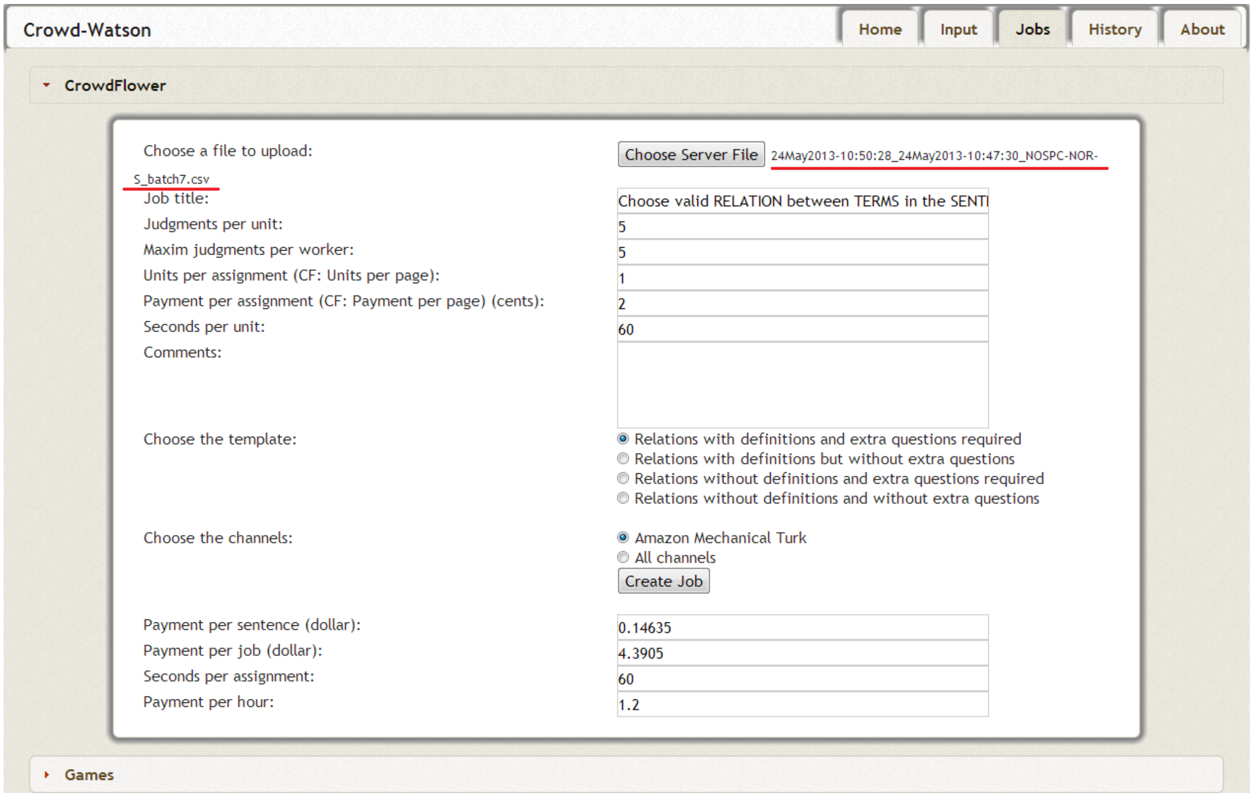
Create the job file:
Introduce the number of sentences for each relation:
□ Cause
□ Contraindicate
□ Diagnose
□ Location
□ Prevent
□ Symptom
□ Treat

Figure 14: Integrate pre-processing scripts of CF module into GUI



File ID	File Name	Filter Applied	Batch Size	Created By	Crea
			30		
● 970	24May2013-10:50:28_24May2013-10:47:30_NOSPC-NOR-S_batch7.csv	NOSPC, NOR, S	30	oana	2013-0
○ 969	24May2013-10:50:24_24May2013-10:47:30_NOSPC-NOR-S_batch6.csv	NOSPC, NOR, S	30	oana	2013-0
○ 968	24May2013-10:50:22_24May2013-10:47:30_NOSPC-NOR-S_batch5.csv	NOSPC, NOR, S	30	oana	2013-0
○ 967	24May2013-10:50:19_24May2013-10:47:30_NOSPC-NOR-S_batch4.csv	NOSPC, NOR, S	30	oana	2013-0
○ 966	24May2013-10:50:05_24May2013-10:47:30_NOSPC-NOR-S_batch3.csv	NOSPC, NOR, S	30	oana	2013-0
○ 965	24May2013-10:49:52_24May2013-10:47:30_NOSPC-NOR-S_batch2.csv	NOSPC, NOR, S	30	oana	2013-0

Figure 15: Enable to select files from the server to create jobs



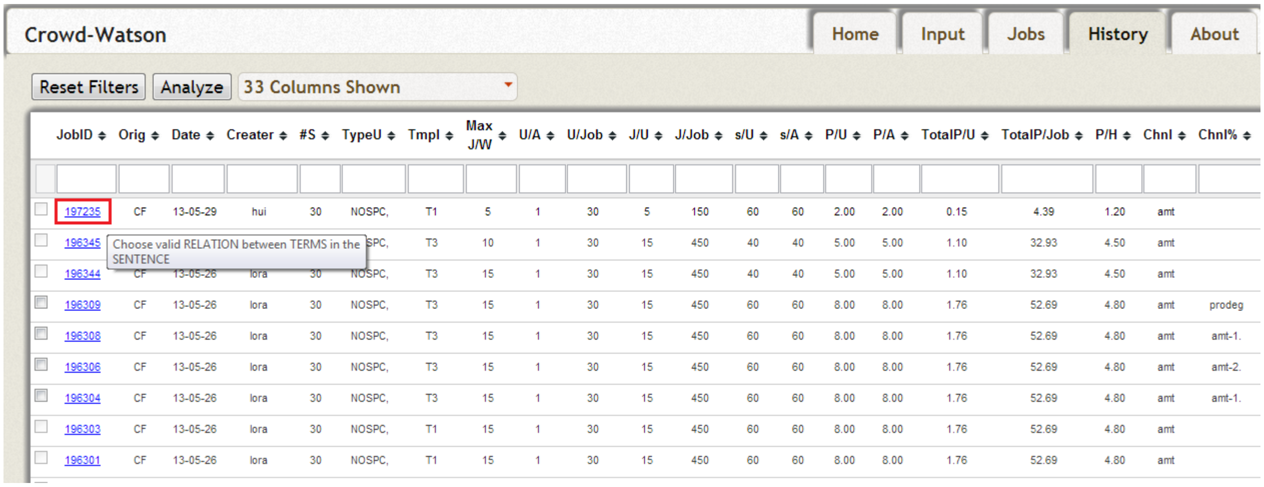
Choose a file to upload:	24May2013-10:50:28_24May2013-10:47:30_NOSPC-NOR-S_batch7.csv
Job title:	Choose valid RELATION between TERMS in the SENTI
Judgments per unit:	5
Maxim judgments per worker:	5
Units per assignment (CF: Units per page):	1
Payment per assignment (CF: Payment per page) (cents):	2
Seconds per unit:	60
Comments:	

Choose the template:
- ● Relations with definitions and extra questions required
- ○ Relations with definitions but without extra questions
- ○ Relations without definitions and extra questions required
- ○ Relations without definitions and without extra questions

Choose the channels:
- ● Amazon Mechanical Turk
- ○ All channels

Payment per sentence (dollar):	0.14635
Payment per job (dollar):	4.3905
Seconds per assignment:	60
Payment per hour:	1.2

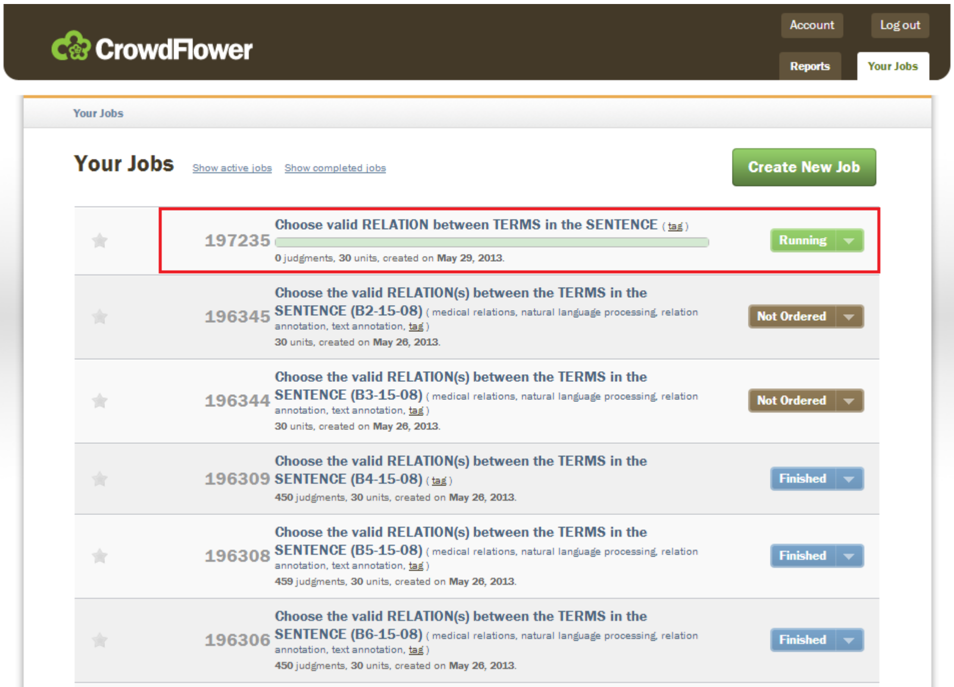
Figure 16: Fill in parameters after selecting a file from the server to create a new job



	JobID	Orig	Date	Creater	#S	TypeU	Tmpl	Max J/W	U/A	U/Job	J/U	J/Job	s/U	s/A	P/U	P/A	TotalP/U	TotalP/Job	P/H	Chnl	Chnl%
☐	197235	CF	13-05-29	hui	30	NOSPC,	T1	5	1	30	5	150	60	60	2.00	2.00	0.15	4.39	1.20	amt	
☐	196345	[illegible]	[illegible]	[illegible]	[illegible]	SPC,	T3	10	1	30	15	450	40	40	5.00	5.00	1.10	32.93	4.50	amt	
☐	196344	CF	13-05-26	lora	30	NOSPC,	T3	15	1	30	15	450	40	40	5.00	5.00	1.10	32.93	4.50	amt	
☐	196309	CF	13-05-26	lora	30	NOSPC,	T3	15	1	30	15	450	60	60	8.00	8.00	1.76	52.69	4.80	amt	prodeg
☐	196308	CF	13-05-26	lora	30	NOSPC,	T3	15	1	30	15	450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-1.
☐	196306	CF	13-05-26	lora	30	NOSPC,	T3	15	1	30	15	450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-2.
☐	196304	CF	13-05-26	lora	30	NOSPC,	T3	15	1	30	15	450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-1.
☐	196303	CF	13-05-26	lora	30	NOSPC,	T1	15	1	30	15	450	60	60	8.00	8.00	1.76	52.69	4.80	amt	
☐	196301	CF	13-05-26	lora	30	NOSPC,	T1	15	1	30	15	450	60	60	8.00	8.00	1.76	52.69	4.80	amt	

Choose valid RELATION between TERMS in the SENTENCE

Figure 17: Job ID 197235 was created in History Table



Your Jobs

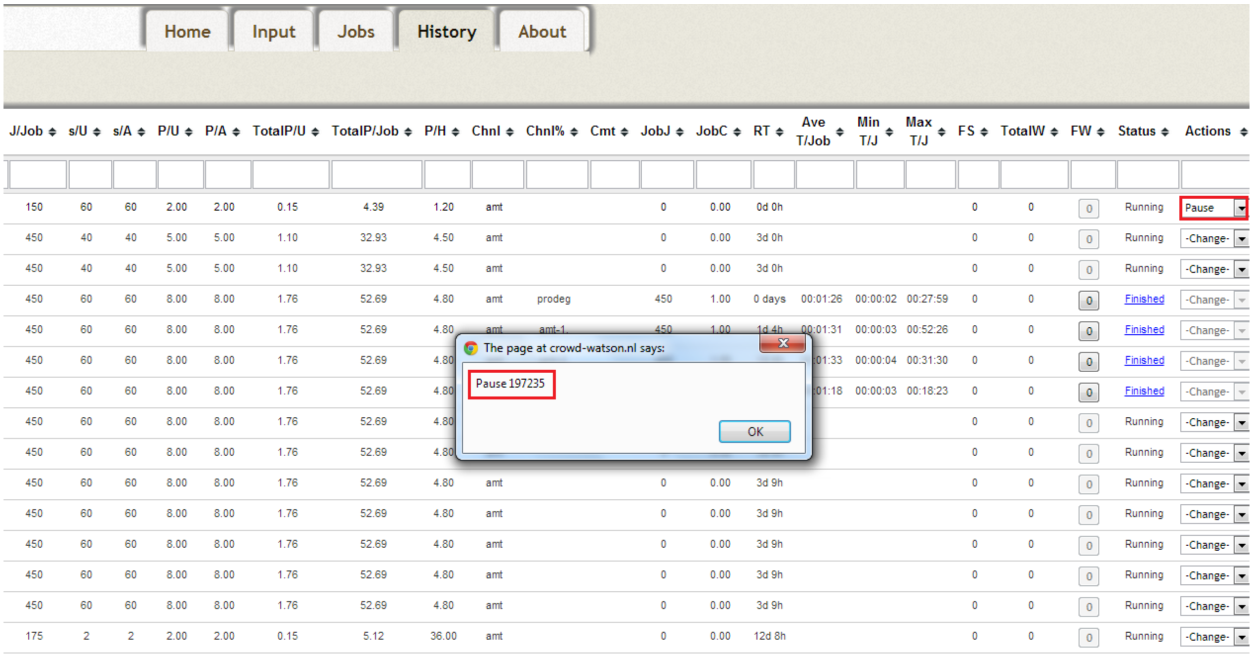
Your Jobs

Show active jobs Show completed jobs

Create New Job

★	197235	Choose valid RELATION between TERMS in the SENTENCE (tag) 0 judgments, 30 units, created on May 29, 2013.	Running
★	196345	Choose the valid RELATION(s) between the TERMS in the SENTENCE (B2-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag) 30 units, created on May 26, 2013.	Not Ordered
★	196344	Choose the valid RELATION(s) between the TERMS in the SENTENCE (B3-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag) 30 units, created on May 26, 2013.	Not Ordered
★	196309	Choose the valid RELATION(s) between the TERMS in the SENTENCE (B4-15-08) (tag) 450 judgments, 30 units, created on May 26, 2013.	Finished
★	196308	Choose the valid RELATION(s) between the TERMS in the SENTENCE (B5-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag) 459 judgments, 30 units, created on May 26, 2013.	Finished
★	196306	Choose the valid RELATION(s) between the TERMS in the SENTENCE (B6-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag) 450 judgments, 30 units, created on May 26, 2013.	Finished

Figure 18: CF indicates the newly-created Job ID 197235



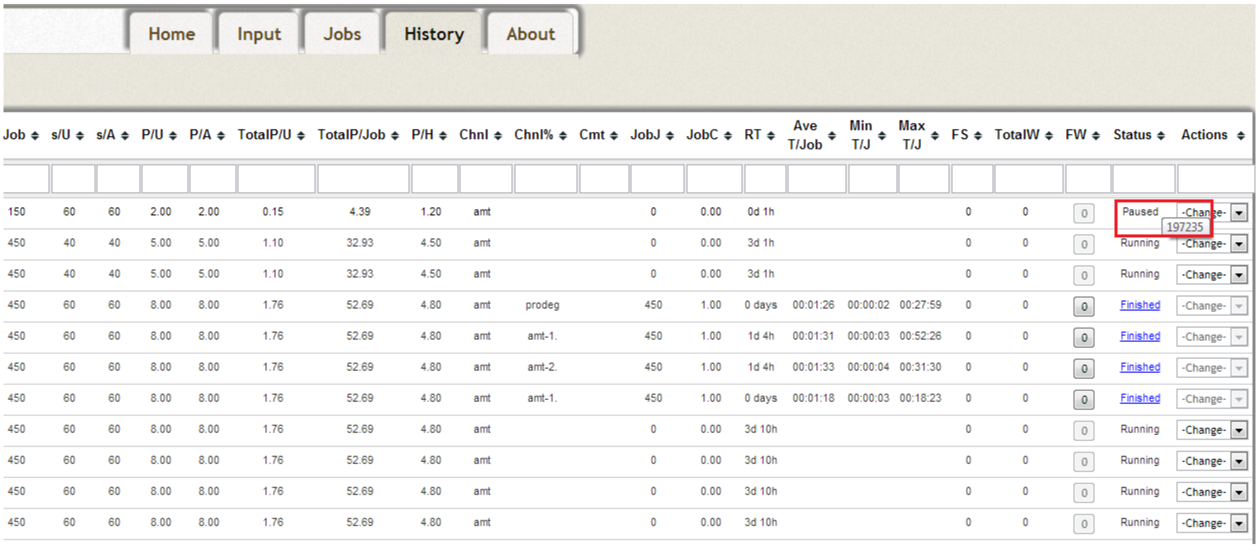
J/Job	s/U	s/A	P/U	P/A	TotalP/U	TotalP/Job	P/H	Chnl	Chnl%	Cmt	JobJ	JobC	RT	Ave T/Job	Min T/J	Max T/J	FS	TotalW	FW	Status	Actions
150	60	60	2.00	2.00	0.15	4.39	1.20	amt			0	0.00	0d 0h				0	0	0	Running	Pause
450	40	40	5.00	5.00	1.10	32.93	4.50	amt			0	0.00	3d 0h				0	0	0	Running	-Change-
450	40	40	5.00	5.00	1.10	32.93	4.50	amt			0	0.00	3d 0h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt	prodeg		450	1.00	0 days	00:01:26	00:00:02	00:27:59	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-1		450	1.00	1d 4h	00:01:31	00:00:03	00:52:26	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80							01:33	00:00:04	00:31:30	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80							01:18	00:00:03	00:18:23	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80										0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80										0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 9h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 9h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 9h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 9h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 9h				0	0	0	Running	-Change-
175	2	2	2.00	2.00	0.15	5.12	36.00	amt			0	0.00	12d 8h				0	0	0	Running	-Change-

The page at crowd-watson.nl says:

Pause 197235

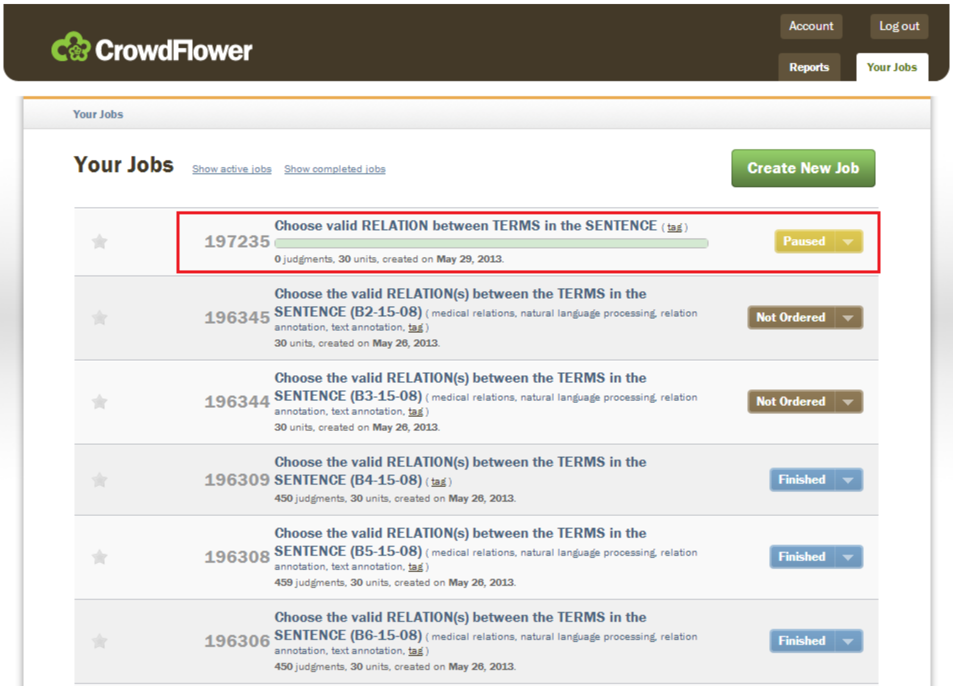
OK

Figure 19: Pause 197235 from History Table



Job	s/U	s/A	P/U	P/A	TotalP/U	TotalP/Job	P/H	Chnl	Chnl%	Cmt	JobJ	JobC	RT	Ave T/Job	Min T/J	Max T/J	FS	TotalW	FW	Status	Actions
150	60	60	2.00	2.00	0.15	4.39	1.20	amt			0	0.00	0d 1h				0	0	0	Paused 197235	-Change-
450	40	40	5.00	5.00	1.10	32.93	4.50	amt			0	0.00	3d 1h				0	0	0	Running	-Change-
450	40	40	5.00	5.00	1.10	32.93	4.50	amt			0	0.00	3d 1h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt	prodeg		450	1.00	0 days	00:01:26	00:00:02	00:27:59	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-1.		450	1.00	1d 4h	00:01:31	00:00:03	00:52:26	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-2.		450	1.00	1d 4h	00:01:33	00:00:04	00:31:30	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt	amt-1.		450	1.00	0 days	00:01:18	00:00:03	00:18:23	0	0	0	Finished	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 10h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 10h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 10h				0	0	0	Running	-Change-
450	60	60	8.00	8.00	1.76	52.69	4.80	amt			0	0.00	3d 10h				0	0	0	Running	-Change-

Figure 20: 197235 was updated to Paused status in History Table after Pause action



Your Jobs

197235 Choose valid RELATION between TERMS in the SENTENCE (tag)
0 judgments, 30 units, created on May 29, 2013.
Paused

196345 Choose the valid RELATION(s) between the TERMS in the SENTENCE (B2-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag)
30 units, created on May 26, 2013.
Not Ordered

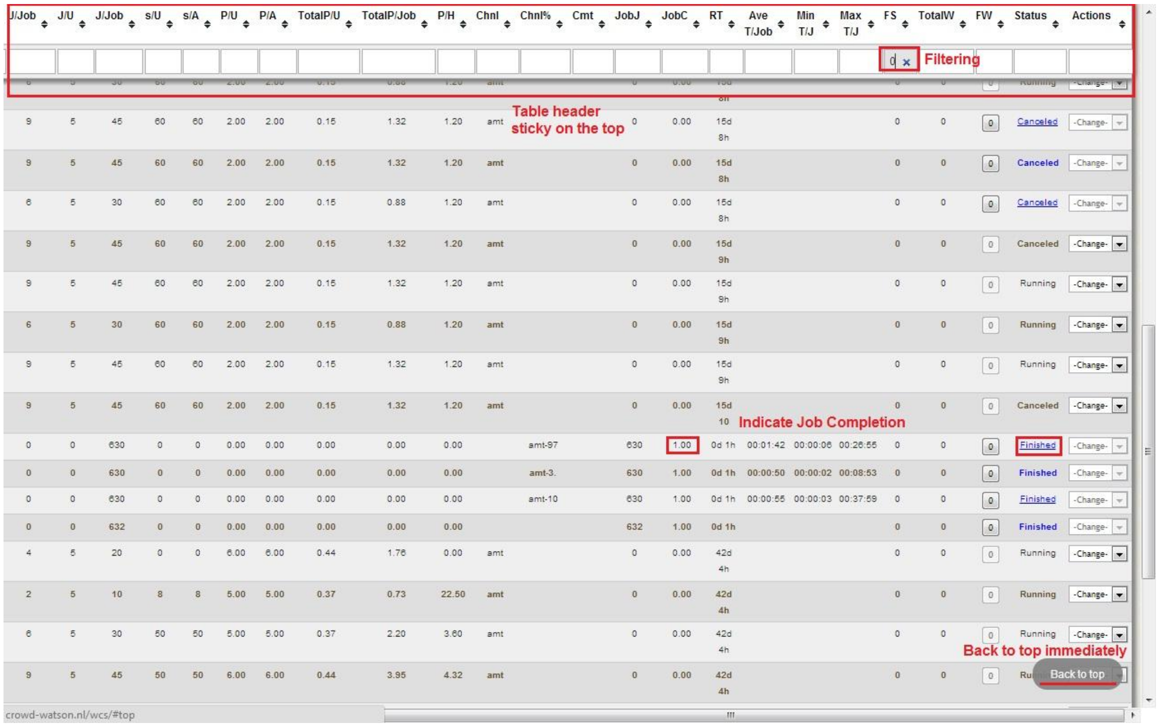
196344 Choose the valid RELATION(s) between the TERMS in the SENTENCE (B3-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag)
30 units, created on May 26, 2013.
Not Ordered

196309 Choose the valid RELATION(s) between the TERMS in the SENTENCE (B4-15-08) (tag)
450 judgments, 30 units, created on May 26, 2013.
Finished

196308 Choose the valid RELATION(s) between the TERMS in the SENTENCE (B5-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag)
459 judgments, 30 units, created on May 26, 2013.
Finished

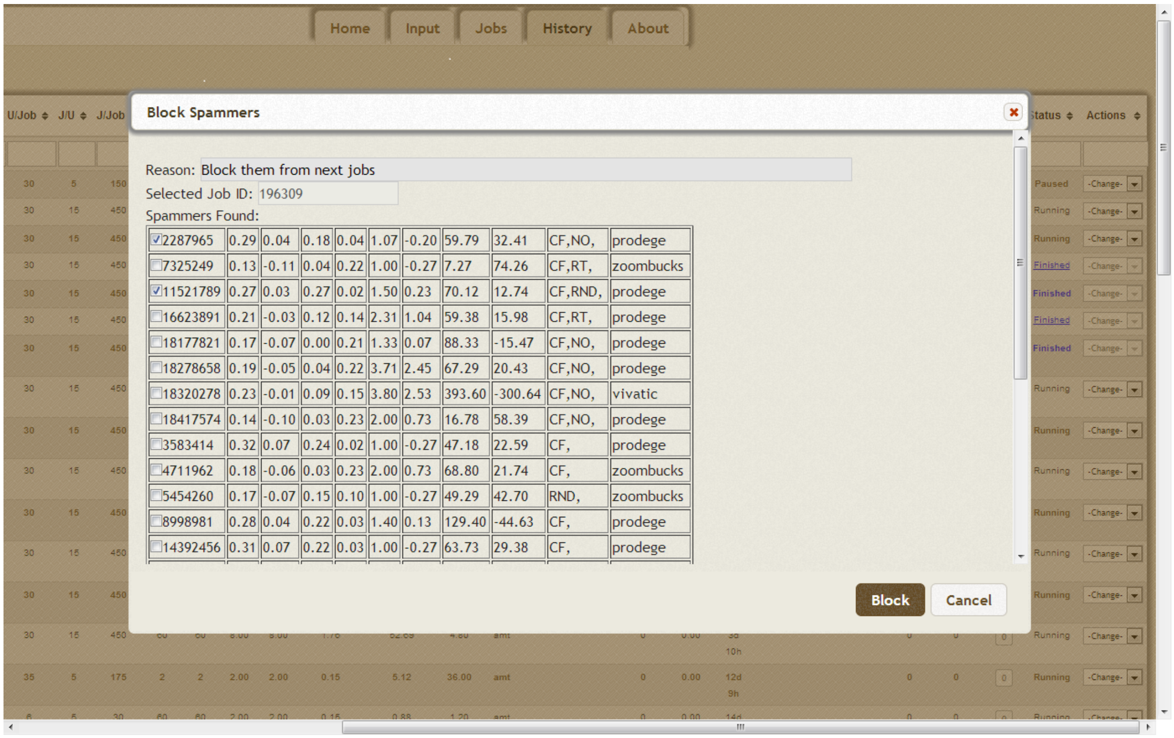
196306 Choose the valid RELATION(s) between the TERMS in the SENTENCE (B6-15-08) (medical relations, natural language processing, relation annotation, text annotation, tag)
450 judgments, 30 units, created on May 26, 2013.
Finished

Figure 21: After pausing 197235 from History Table, CF updated 197235 to paused



U/Job	J/U	J/Job	s/U	s/A	P/U	P/A	TotalP/U	TotalP/Job	P/H	Chnl	Chnl%	Cmt	JobJ	JobC	RT	Ave T/Job	Min T/J	Max T/J	FS	TotalW	FW	Status	Actions
6	5	30	60	60	2.00	2.00	0.15	0.88	1.20	amt			0	0.00	15d 8h				0	0	0	Running	-Change-
9	5	45	60	60	2.00	2.00	0.15	1.32	1.20	amt			0	0.00	15d 8h				0	0	0	Canceled	-Change-
9	5	45	60	60	2.00	2.00	0.15	1.32	1.20	amt			0	0.00	15d 8h				0	0	0	Canceled	-Change-
6	5	30	60	60	2.00	2.00	0.15	0.88	1.20	amt			0	0.00	15d 8h				0	0	0	Canceled	-Change-
9	5	45	60	60	2.00	2.00	0.15	1.32	1.20	amt			0	0.00	15d 9h				0	0	0	Canceled	-Change-
9	5	45	60	60	2.00	2.00	0.15	1.32	1.20	amt			0	0.00	15d 9h				0	0	0	Running	-Change-
6	5	30	60	60	2.00	2.00	0.15	0.88	1.20	amt			0	0.00	15d 9h				0	0	0	Running	-Change-
9	5	45	60	60	2.00	2.00	0.15	1.32	1.20	amt			0	0.00	15d 9h				0	0	0	Running	-Change-
9	5	45	60	60	2.00	2.00	0.15	1.32	1.20	amt			0	0.00	15d 10				0	0	0	Canceled	-Change-
0	0	630	0	0	0.00	0.00	0.00	0.00	0.00		amt-97		630	1.00	0d 1h	00:01:42	00:00:06	00:26:55	0	0	0	Finished	-Change-
0	0	630	0	0	0.00	0.00	0.00	0.00	0.00		amt-3.		630	1.00	0d 1h	00:00:50	00:00:02	00:08:53	0	0	0	Finished	-Change-
0	0	630	0	0	0.00	0.00	0.00	0.00	0.00		amt-10		630	1.00	0d 1h	00:00:55	00:00:03	00:37:59	0	0	0	Finished	-Change-
0	0	632	0	0	0.00	0.00	0.00	0.00	0.00				632	1.00	0d 1h				0	0	0	Finished	-Change-
4	5	20	0	0	6.00	6.00	0.44	1.76	0.00	amt			0	0.00	42d 4h				0	0	0	Running	-Change-
2	5	10	8	8	5.00	5.00	0.37	0.73	22.50	amt			0	0.00	42d 4h				0	0	0	Running	-Change-
6	5	30	50	50	5.00	5.00	0.37	2.20	3.60	amt			0	0.00	42d 4h				0	0	0	Running	-Change-
9	5	45	50	50	6.00	6.00	0.44	3.95	4.32	amt			0	0.00	42d 4h				0	0	0	Ru	

Figure 22: History Table and Back to top button



Block Spammers

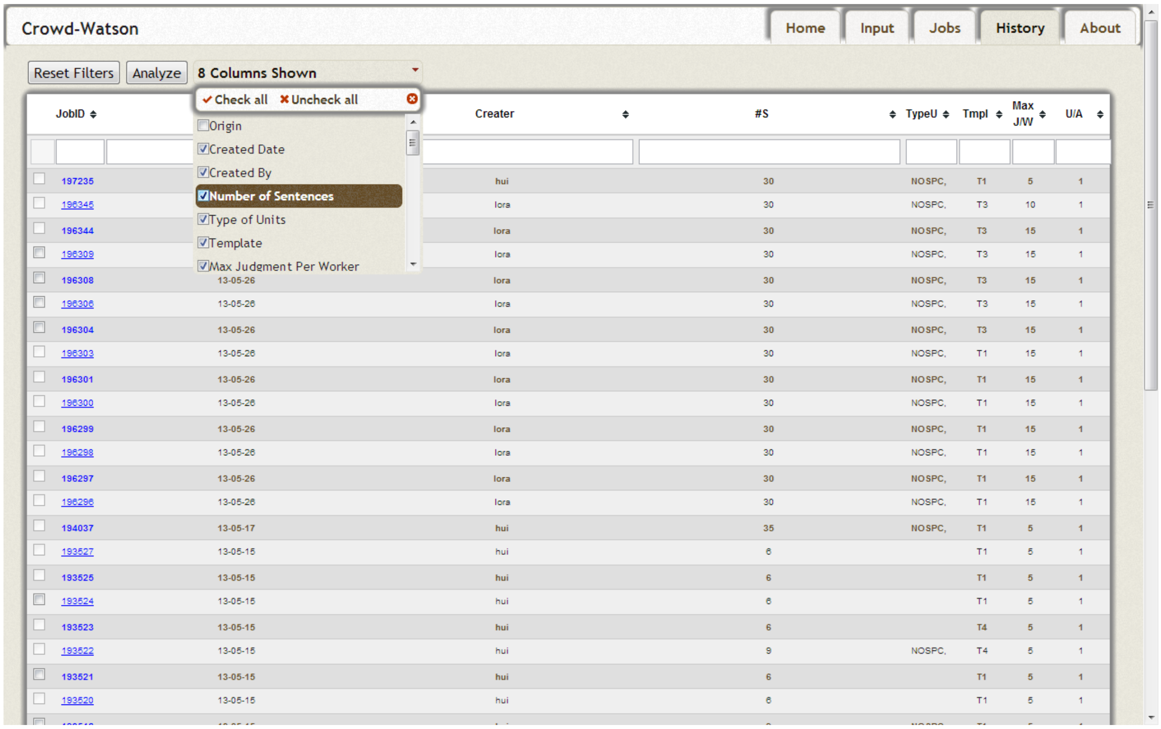
Reason: Block them from next jobs

Selected Job ID: 196309

Spammers Found:

☑2287965	0.29	0.04	0.18	0.04	1.07	-0.20	59.79	32.41	CF,NO,	prodege
□7325249	0.13	-0.11	0.04	0.22	1.00	-0.27	7.27	74.26	CF,RT,	zoombucks
☑11521789	0.27	0.03	0.27	0.02	1.50	0.23	70.12	12.74	CF,RND,	prodege
□16623891	0.21	-0.03	0.12	0.14	2.31	1.04	59.38	15.98	CF,RT,	prodege
□18177821	0.17	-0.07	0.00	0.21	1.33	0.07	88.33	-15.47	CF,NO,	prodege
□18278658	0.19	-0.05	0.04	0.22	3.71	2.45	67.29	20.43	CF,NO,	prodege
□18320278	0.23	-0.01	0.09	0.15	3.80	2.53	393.60	-300.64	CF,NO,	vivatic
□18417574	0.14	-0.10	0.03	0.23	2.00	0.73	16.78	58.39	CF,NO,	prodege
□3583414	0.32	0.07	0.24	0.02	1.00	-0.27	47.18	22.59	CF,	prodege
□4711962	0.18	-0.06	0.03	0.23	2.00	0.73	68.80	21.74	CF,	zoombucks
□5454260	0.17	-0.07	0.15	0.10	1.00	-0.27	49.29	42.70	RND,	zoombucks
□8998981	0.28	0.04	0.22	0.03	1.40	0.13	129.40	-44.63	CF,	prodege
□14392456	0.31	0.07	0.22	0.03	1.00	-0.27	63.73	29.38	CF,	prodege

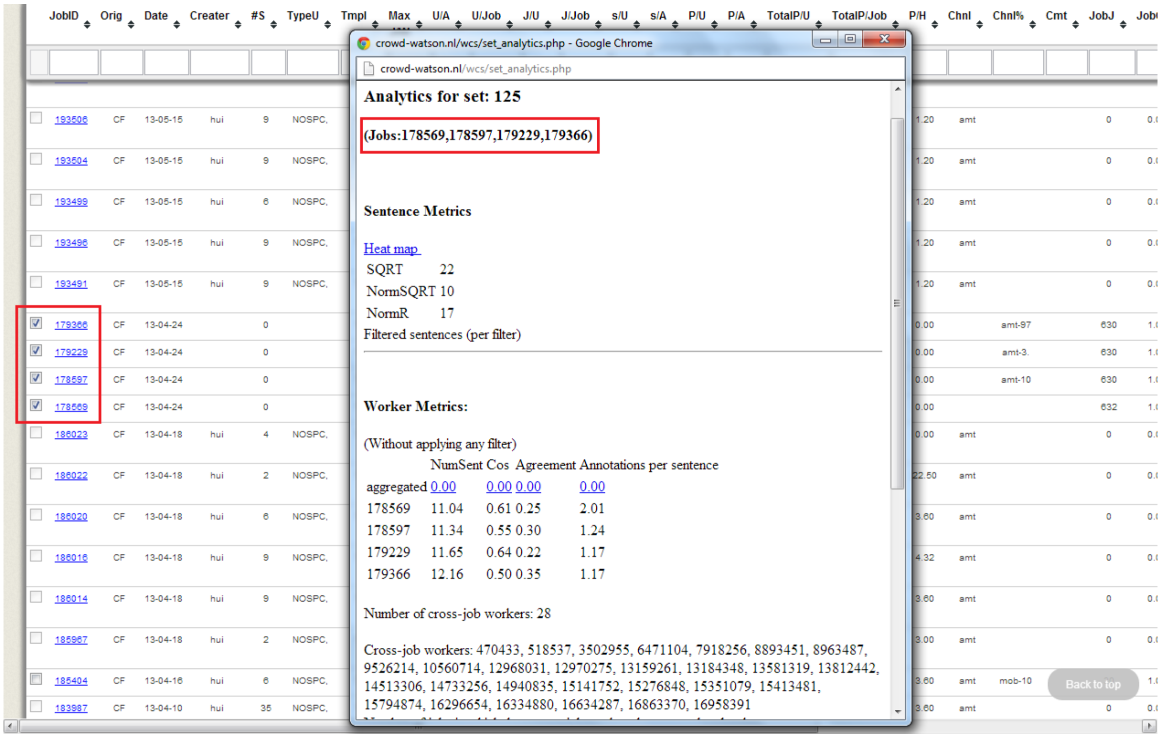
Figure 23: Enable to block spammers of a particular finished job



□Origin
☑Created Date
☑Created By
☑Number of Sentences
☑Type of Units
☑Template
☑Max Judgment Per Worker

	JobID		Creater	#S	TypeU	Tmpl	Max J/W	U/A
□	197235		hui	30	NOSPC,	T1	5	1
□	196345		lora	30	NOSPC,	T3	10	1
□	196344		lora	30	NOSPC,	T3	15	1
□	196309		lora	30	NOSPC,	T3	15	1
□	196308	13-05-26	lora	30	NOSPC,	T3	15	1
□	196306	13-05-26	lora	30	NOSPC,	T3	15	1
□	196304	13-05-26	lora	30	NOSPC,	T3	15	1
□	196303	13-05-26	lora	30	NOSPC,	T1	15	1
□	196301	13-05-26	lora	30	NOSPC,	T1	15	1
□	196300	13-05-26	lora	30	NOSPC,	T1	15	1
□	196299	13-05-26	lora	30	NOSPC,	T1	15	1
□	196298	13-05-26	lora	30	NOSPC,	T1	15	1
□	196297	13-05-26	lora	30	NOSPC,	T1	15	1
□	196296	13-05-26	lora	30	NOSPC,	T1	15	1
□	194037	13-05-17	hui	35	NOSPC,	T1	5	1
□	193527	13-05-15	hui	6		T1	5	1
□	193525	13-05-15	hui	6		T1	5	1
□	193524	13-05-15	hui	6		T1	5	1
□	193523	13-05-15	hui	6		T4	5	1
□	193522	13-05-15	hui	9	NOSPC,	T4	5	1
□	193521	13-05-15	hui	6		T1	5	1
□	193520	13-05-15	hui	6		T1	5	1

Figure 24: Hide and show columns of History Table



crowd-watson.nl/wcs/set_analytics.php - Google Chrome
crowd-watson.nl/wcs/set_analytics.php
Back to top

	JobID	Orig	Date	Creater	#S	TypeU	P/H	Chnl	Chnl%	Cmt	JobJ	Job
□	193506	CF	13-05-15	hui	9	NOSPC,	1.20	amt			0	0.
□	193504	CF	13-05-15	hui	9	NOSPC,	1.20	amt			0	0.
□	193499	CF	13-05-15	hui	6	NOSPC,	1.20	amt			0	0.
□	193496	CF	13-05-15	hui	9	NOSPC,	1.20	amt			0	0.
□	193491	CF	13-05-15	hui	9	NOSPC,	1.20	amt			0	0.
☑	179366	CF	13-04-24		0		0.00		amt-97		630	1.
☑	179229	CF	13-04-24		0		0.00		amt-3.		630	1.
☑	178597	CF	13-04-24		0		0.00		amt-10		630	1.
☑	178569	CF	13-04-24		0		0.00				632	1.
□	186023	CF	13-04-18	hui	4	NOSPC,	0.00	amt			0	0.
□	186022	CF	13-04-18	hui	2	NOSPC,	22.50	amt			0	0.
□	186020	CF	13-04-18	hui	6	NOSPC,	3.60	amt			0	0.
□	186016	CF	13-04-18	hui	9	NOSPC,	4.32	amt			0	0.
□	186014	CF	13-04-18	hui	9	NOSPC,	3.60	amt			0	0.
□	185967	CF	13-04-18	hui	2	NOSPC,	3.00	amt			0	0.
□	185404	CF	13-04-16	hui	6	NOSPC,	3.60	amt	mob-10		[illegible]	1.
□	183987	CF	13-04-10	hui	35	NOSPC,	3.60	amt			0	0.

Analytics for set: 125

(Jobs:178569,178597,179229,179366)

Sentence Metrics

Heat map

SQRT	22
NormSQRT	10
NormR	17

Filtered sentences (per filter)

Worker Metrics:

(Without applying any filter)

	NumSent	Cos	Agreement	Annotations per sentence
aggregated	0.00	0.00	0.00	0.00
178569	11.04	0.61	0.25	2.01
178597	11.34	0.55	0.30	1.24
179229	11.65	0.64	0.22	1.17
179366	12.16	0.50	0.35	1.17

Number of cross-job workers: 28

Cross-job workers: 470433, 518537, 3502955, 6471104, 7918256, 8893451, 8963487, 9526214, 10560714, 12968031, 12970275, 13159261, 13184348, 13581319, 13812442, 14513306, 14733256, 14940835, 15141752, 15276848, 15351079, 15413481, 15794874, 16296654, 16334880, 16634287, 16863370, 16958391

Figure 25: View statistical results dynamically in the pop-up Results pages

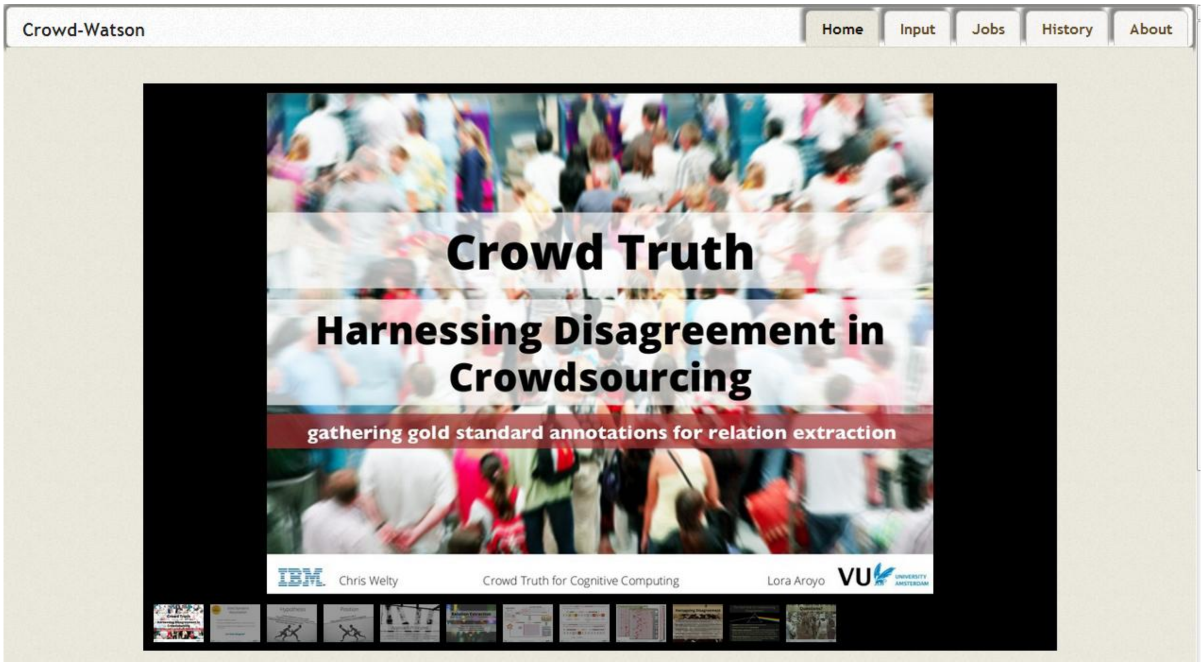
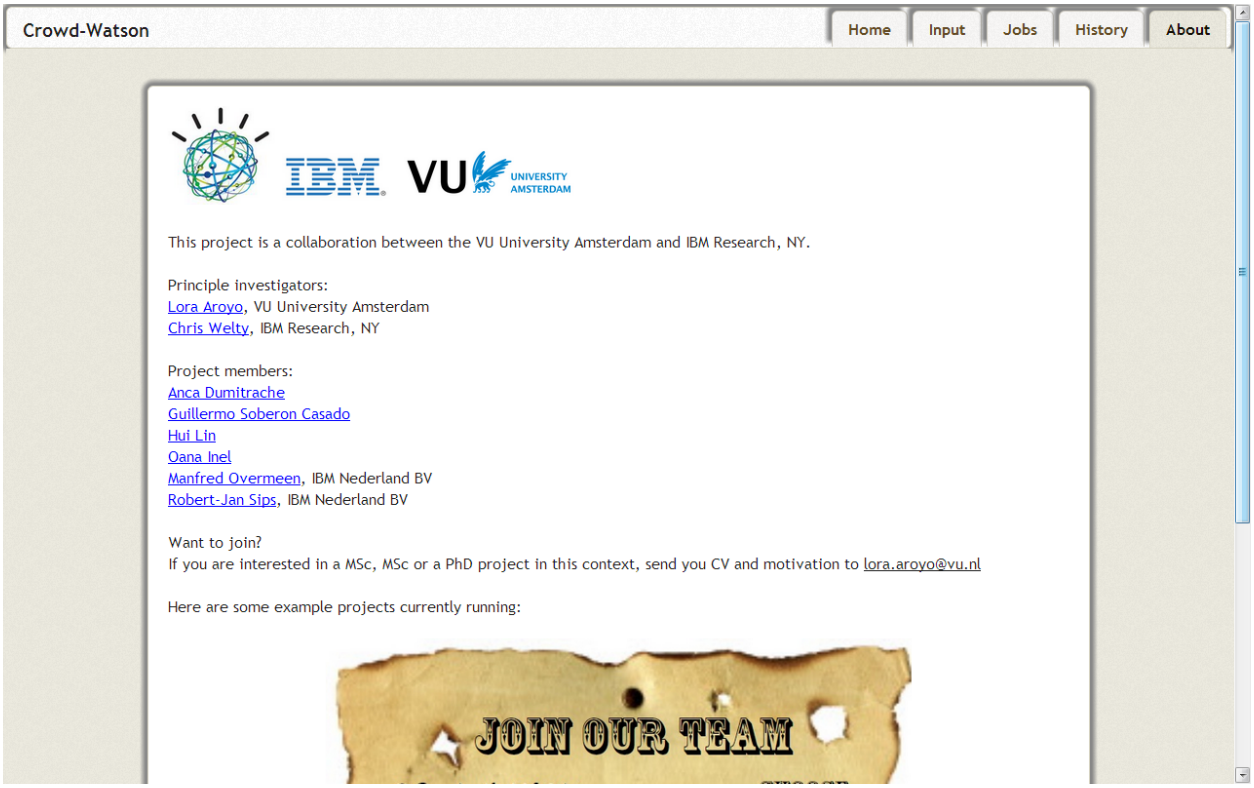

Crowd-Watson
Home
Input
Jobs
History
About
Crowd Truth
Harnessing Disagreement in Crowdsourcing
gathering gold standard annotations for relation extraction
IBM
Chris Welty
Crowd Truth for Cognitive Computing
Lora Aroyo
VU UNIVERSITY AMSTERDAM

Figure 26: Add a slideshow to indicate general details of the Crowd-Watson project [11]



This project is a collaboration between the VU University Amsterdam and IBM Research, NY.

Principle investigators:
Lora Aroyo, VU University Amsterdam
Chris Welty, IBM Research, NY

Project members:
Anca Dumitrache
Guillermo Soberon Casado
Hui Lin
Oana Inel
Manfred Overmeen, IBM Nederland BV
Robert-Jan Sips, IBM Nederland BV

Want to join?
If you are interested in a MSc, MSc or a PhD project in this context, send you CV and motivation to lora.aroyo@vu.nl

Here are some example projects currently running:

Figure 27: Introduce the Crowd-Watson project members in About page

7. Conclusion and Recommendation

Crowd-Watson Crowdsourced Text Annotations tool offers front-end GUI to help users to interact with different platforms with uniformity and to enable the further interactions with more platforms later on. With its back-end database and server, it is efficient to set up large-scale experiments and to reduce manual work. The overall custom design of the tool meets the needs for configurability, scalability and extendibility, which can support the extensive research. The tool would be extended with more functions, modules and processing nodes along with more research output of the Crowd-Watson project. In the end, as a part of the Crowd-Watson project, the tool would be released as open resource software.

There are some points to be mentioned for recommendations. The Crowd-Watson project is a type of university research project and focuses a lot on researching theories and discoveries. It does not specify software development requirements clearly like normal company projects. Developers need to make certain assumptions and trials before knowing what the clients really expect. Time spent on modifications and confirmations is more than normal projects. The communication between researching clients and developers is very essential. This project also requires higher researching ability from developers. It would be best if the Crowd-Watson project researchers can spend more time, in advance, defining software development requirements. Due to multiple modules, various data and a lot of statistical calculations, the final verification of tooling is essential.

8. Personal Evaluation

Through the graduation project at IBM and VU University Amsterdam, I have gained more knowledge of Web development by using PHP, jQuery, JavaScript and MySQL. Besides that, working together with the university professor, IBM researcher and master students in the Crowd-Watson project brought me a new challenge. It was a little difficult in the beginning to read through multiple research papers and make comparisons of two different frameworks. It also took some time to get familiar with main modules researched by the three master students. During the course of this project, I learned how to set the priorities of tasks based on the importances and impacts as well as how to handle relatively frequent changes of client requirements, which everyone can encounter in the job fields. Thanks to this internship chance offered by the Crowd-Watson team, I have enjoyed learning the new skill of programming Web architecture and gathered experience of framework researches.

I have gained experience and improved my skills in working and communication in an internationally oriented company and a research university environment. This has contributed to my English, reading, writing, reasoning and making argumentations skills. As for socialization in the company, sometimes I found it challenging since a lot of the employees in the department work from home or attend meetings often. But I was able to communicate regularly with the interns at CAS who are excellent, kind and acknowledgeable and often had nice conversations during lunch breaks.

9. Bibliography

[1] History of IBM, http://en.wikipedia.org/wiki/History_of_IBM, May 2013.

[2] S. Thaler, E. Simperl, S. Wolger, An Experiment in Comparing Human-Computation Techniques, IEEE Computer, 2012.

[3] A.J. Quinn, B.B. Bederson, Human Computation: A Survey and Taxonomy of a Growing Field, Vancouver, Canada, May 7-12, 2011.

[4] D. Wightman, Crowdsourcing Human-Based Computation, 2010.

[5] L. Von Ahn, L. Dabbish, Designing Games with a Purpose, IEEE Computer, Vol. 51, No. 8, August, 2008.

[6] S. Reeves, S. Sherwood, Five design challenges for human computation, October 2010.

[7] Chris Welty, http://en.wikipedia.org/wiki/Chris_Welty, June 2013.

[8] Salman Ahmad, Alexis Battle, Zahan Malkani, Sepandar D. Kamvar, The Jabberwocky Programming Environment for Structured Social Computing, Stanford University, Stanford, CA, USA, 2011.

[9] Patrick Minder, Abraham Bernstein, CrowdLang: programming human computation systems, University of Zurich, 2012.

[10] Schwaber, K. (1 Feb 2004), Agile Project Mangement with Scrum (Microsoft Professional). Grove City: Microsoft Press.

[11] Lora Aroyo, WebSci2013 Harnessing Disagreement in Crowdsourcing, April 28, 2013.

Fontys
University of Applied Sciences

Appendix A

Project Plan

Project Plan

Crowd-Watson

Crowdsourced Text Annotations

Hui Lin

Information Communication Technology

Fontys University of Applied Science

IBM Amsterdam

February, 2013

Contents

1. General Information

Student Name	Hui Lin	Student Number	2142435
		Study Program	ICT
		Tel	+31624689451
		Email	alice8linhui@gmail.com
Company Tutor	Manfred Overmeen	Position Title	IT Specialist
		Email	manfred.overmeen@nl.ibm.com
		Mobile Phone	+31 651 34 9190
School Tutor	Erik van der Schriek	Email	e.vanderschriek@fontys.nl
		Phone	0885076608

2. Project Statement

This chapter indicates the general details of the overall project.

Formal client

IBM Thomas J. Watson Research Center is the formal client

Project leader

Lora Aroyo

VU University Amsterdam

The Netherlands

Chris Welty

IBM Thomas J.Watson Research Center

USA

Current situation

IBM super computer Watson is a typical example of Artificial Intelligence. Watson is able to understand human natural language. It was named after IBM´s founder Thomas J. Watson and specially developed to answer questions on the quiz show Jeopardy. In 2011, Watson beat former winner Brad Rutter, and Ken Jennings on Jeopardy. However, Watson cannot think like the human brain but can only learn, synthesize and provide an output. Through various NLP (natural language processing) training, Watson is becoming more and more knowledgeable.

Watson would bring a lot of benefits to our daily life and business in terms of applying AI (artificial intelligence) technology. Currently, IBM research team focuses on how to improve Watson to contribute more and more to the medical fields.

Project justification

The output of the Crowd-Watson project will be the part of NLP training for Watson so that it can understand and analyze medical terms better.

The goal is to build a text annotation tool with the combination of crowdsourcing and automation modules for research initiatives.

The tool should be able to efficiently assign and manage microtasks through various crowdsourcing platforms (e.g. Amazon Mechanical Turk, CrowdFlower, TurKit etc.)

Crowdsourcing is the process of getting work or funding, usually online, from a crowd of people. The word is a combination of the words ‘crowd’ and ‘outsourcing’. The idea is to take work and outsource it to a crowd of workers. By canvassing a large crowd of people for ideas, skills or participation, the quality of content and idea generation will be superior.

Microtasking involves breaking work up into tiny tasks and sending the work to a crowd of people and deciding the payment for each completed task. Work is done faster, cheaper, and usually with less errors.

The main topic of this research is to investigate (and implement) an efficient framework for handling microtask management. Such a framework should be able to:

- Process a –configurable- number of inputs (e.g. 1 sentence, 2 sentences, a text-fragment),

- Process the input(s) by means of a workflow, consisting of an arbitrary number of processing nodes, processing nodes may be human (e.g. crowdsourced), machine (thus: automated) and other (e.g. gaming), to cope with the “other” requirement, a standardized interface should be available towards these nodes,

- Process the output, in terms of default statistics, e.g. success rate, number of annotations per sentence, number of sentences per worker, etc.

Finally, give a view of the outcome / results (independent of the type of processing conducted).

• Project product

The Crowd-Watson text annotation application (codes and relative documents)

Final project report describing the development processes of the Crowd-Watson project and main activities

• Project deliverables and non-deliverables (for Fontys)

<u>Project deliverables</u>

Final project report describing the development processes of project, main activities and functional extensions.

PowerPoint document giving the introduction and showing certain prototypes of the Crowd-Watson product for the final presentation both at IBM and at Fontys.

Project non-deliverables

Complete coding files and relative database resources

The Crowd-Watson text annotation application product

(Due to IBM confidential Policy)

- **Project constraints**

Programming Languages	Java, JavaScript and PHP etc.
Development Tools	IBM software such as Rational Software Architect and IBM Lotus Symphony or Open source applications like Eclipse, Aptana and LibreOffice
Conducting Environment	IBM HQ Amsterdam and VU University (Web Sciences department)

- **Project risks**

Project Risks	Probability	Impact	Precaution	Repairing
Make improper decisions	Medium	High	- Certain meetings will be arranged on time to make the essential changes with better considerations and to adjust the project into the right directions. - Make some backup plans in order to adjust quickly	- Take the backup plans to be back to the right direction promptly - Arrange emergent meetings for more potential solutions and quicker reactions - Require extra resources or help
Out of time limits / Progress delay	High	Medium	- Keep individual progresses update regularly - Indicate problems or bottlenecks on time - Make sure efficient communications among team members - Ensure enough regular guidance or instructions - Access to relevant resources and help	- Find out the causes of the problem - Extra team members or time can be offered to certain extend for the specific project phasing
System crashes or hackers	Low	High	- Obey IBM security policy and install IBM security software - Do security scan for the laptop regularly - Watch out suspicious emails, hyperlinks and Web sites	Emergent actions and security solutions will be operated immediately to save work done stop or avoid certain damage
Project cancellation due to management decisions	Low	High	- Good communication - Active progresses - Efficient problem solving	The project group is not able to cope with the cancellation and stop of the project due to management decisions

3. Individual Project Phasing

This section covers how the intern will conduct her part within the project team from February to June (Refer to the workflow chart at Page 11 for the project scope of the intern).

- ### Framework

(4th Feb 2013 – 15th Feb 2013)

The first step would be to investigate the current state of the art frameworks in Crowdsourcing Microtask Management, more particularly to compare the ManReduce framework and the CrowdLang framework and to determine whether these frameworks could (partially) fulfill the demands.

- ### Architecture

(18th Feb 2013 – 11st March 2013)

After a framework has been chosen, it has to be included into an architecture, taking into account the need for configurability (workflow, processing, in- /output and view-templating), and scalability (potentially scaling up to a large number of tasks / interactions) and extendibility (the number of processing node types and in- /output types defined at current may be extended at a later stage).

- ### Implementation

(4th March 2013 – 29th June 2013)

Finally, the designed architecture should be implemented (preferably using Java). Certain testing is also conducted during the implementation.

4.Management plan

This part describes the management details of the overall project through MOSQLUITO.

- ## Money

	Costs	Profits
During the Project	• Employees' wages • Licensed fees of third parties' applications • Daily utilities costs	• Teamwork • Better communications and cooperation • Project experiences
After the Project	• Training • Administration • Extension • Maintenance	• Able to efficiently assign and manage microtasks through various crowdsourcing platforms • Configurability, scalability and extendibility • Fit for use

Due to IBM confidential policy, the exact budget is kept private.

- **Skills & Time**

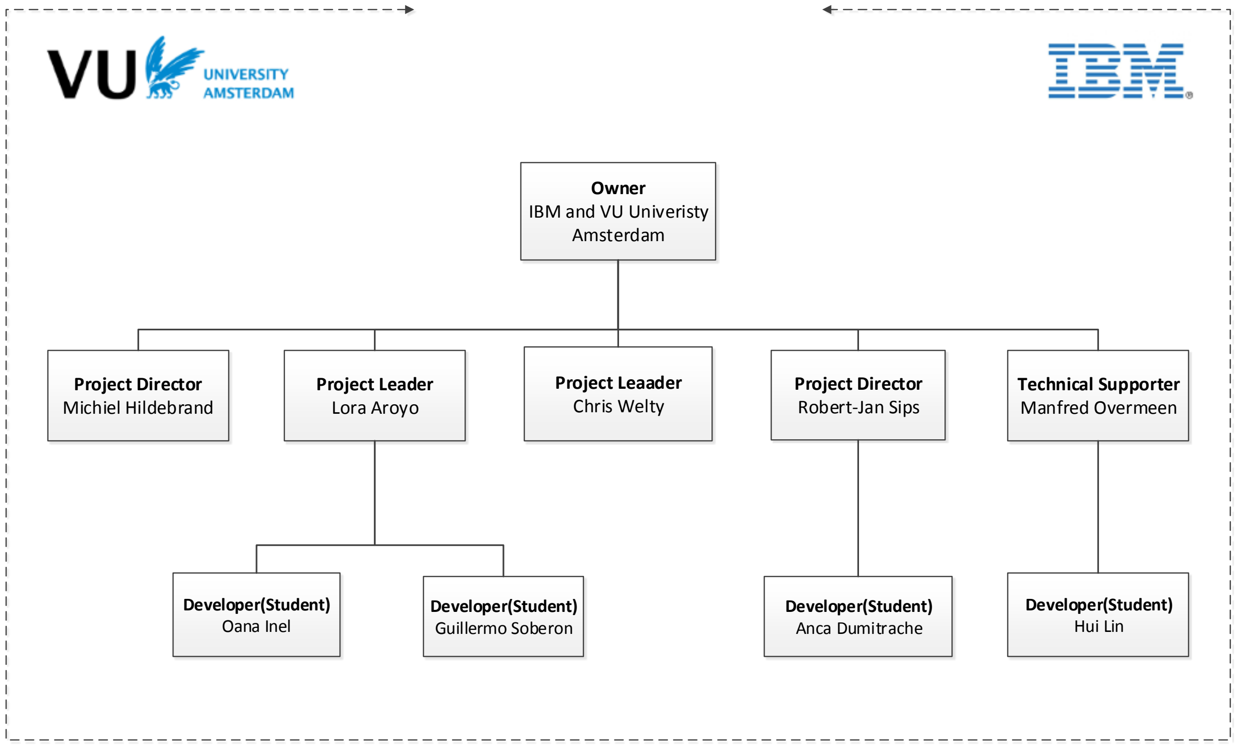
Skills	Responsibilities	Team member & Time	
Planning	Start-up plan and initial analysis	Lora Aroyo Chris Welty Robert-Jan Sips	160 hours each
Analysis	Create Project Charter, Research Case, Project Product Description (URS) and Project Schedule	Lora Aroyo Chris Welty Manfred Overmeen Michiel Hildebrand	60 hours each
Instruction	Offer relative guidance and feedbacks		
Implementation	Researching, coding and testing of the tooling	Guillelmo Oana Inel Anca Dumitrache Hui Lin	450 hours each 450 hours each 800 hours each 800 hours each

- **Quality**

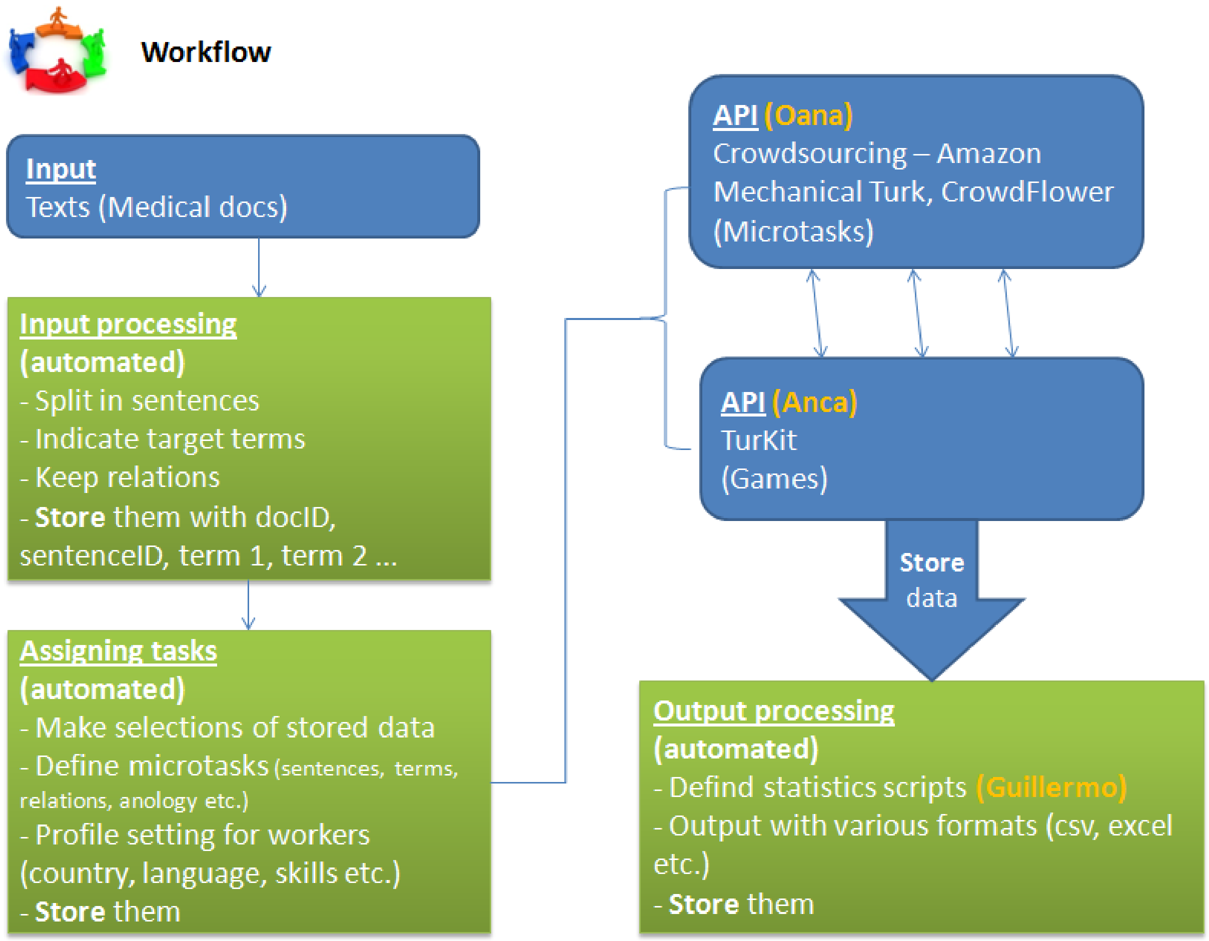
Quality means being precise and on time. The project is well-considered and the project team mainly consists of university researchers and master students. Roles and responsibilities are precisely divided combining with clear deadlines. Important documents are well created and stored in order to offer better understanding and to track step by step. Regular meetings are arranged for checking, making proper changes and further discussions. The testing period is to verify the tooling.

- **Information & Organisation**

Team member	**Role**	**Responsibilities**
Lora Aroyo	Researcher at VU University	- Start-up plan and initial analysis - Create Project Charter, Research Case, Project Product Description (URS) and Project Schedule - Offer relative guidance and feedbacks
Chris Welty	Researcher at IBM New York (One of the developers of Watson)	
Robert-Jan Sips	University Relations Manager at IBM Amsterdam	
Manfred Overmeen	IT Specialist at IBM Amsterdam	Offer the guidance and feedback
Michiel Hildebrand	Research at VU University	
Oana Inel	Computer Science Master student at VU University	Researching on various crowdsouring platforms
Guillermo Soberon Casado	Artificial Intelligence Master student at VU University	Researching on statistics scripts
Anca Dumitrache	IBM Intern / Artificial Intelligence Master student at VU University	Researching on TurKit
Hui Lin	IBM Intern / ICT Bachelor student at Fontys	Researching on the framework and integration



Crowd-Watson



Workflow

Input
Texts (Medical docs)

Input processing (automated)
- Split in sentences
- Indicate target terms
- Keep relations
- **Store** them with docID, sentenceID, term 1, term 2 ...

Assigning tasks (automated)
- Make selections of stored data
- Define microtasks (sentences, terms, relations, anology etc.)
- Profile setting for workers (country, language, skills etc.)
- **Store** them

API (Oana)
Crowdsourcing – Amazon Mechanical Turk, CrowdFlower (Microtasks)

API (Anca)
TurKit
(Games)

Store data

Output processing (automated)
- Defind statistics scripts **(Guillermo)**
- Output with various formats (csv, excel etc.)
- **Store** them

Hui Lin will be in charge of researching on the framework, creating the framework and making the final integration. Please view the attached document called Hui Research Description for the more details of the project scope of Hui Lin.

Approved by the company tutor

Data:

Signature:

Approved by the school tutor

Date:

Signature:

Appendix B

Assignment Descriptions

A general framework for the management of crowdsourcing microtasks

In order to understand text, computers should be able to deal with *ambiguity* and *vagueness* in the language used. Humans deal with this by understanding *context* and using background knowledge they may have on a certain subject.

For computers this is not trivial. Computers may reason about text, by utilizing ontologies and taxonomies, which allow understanding (and generalization) of concepts.

However, this type of annotation doesn't solve the language understanding problem.

For instance, consider the sentences

> "For the prevention of catheder-related central-line infections, a
>
> 2% chlorhexidine (CHG) and 70% isopropyl has the highest success rate." **(A)**
>
> and
>
> "The use of chlorhexidine gluconate rather than povidone-iodine
>
> solution for catheter site care may be an effective and simple measure for improving patient safety by reducing vascular catheter-related infections." **(B)**

An ontology may (or may not) contain the concept of "catheter-related infection", as an "illness", and contain that "chlorhexidine gluconate" and "povidone-iodine" are used for treatment.

For a computer, to be able to understand this sentence three tasks should be conducted (as a first step):

1. **Term factorization**:

   Term factorization would be the task to determine the medical terms mentioned in a sentence. For instance, a computer should learn to understand that the medical terms in sentence **(A)** are "catheder-related central-line infections", "2% chlorhexidine (CHG)" and "70% isopropyl", and more importantly the exact words used to describe those terms.

As an example, would the concept denoted the chlorhexidine solution be best described by “chlorhexidine”, “chlorhexidine (CHG)”, “2% chlorhexidine”, or “2% chlorhexidine (CHG)”?

Moreover, to reason effectively, it should know that [chlorhexidine (CHG)] would probably mean [chlorhexidine gluconate] and, hence, is the same concept as mentioned in sentence **(B)**.

2. Relation-extraction

Provided we reach agreement on the terms in the aforementioned sentence, a next step would be to extract the relation between those terms. A human reader may understand that by *combining* [2% CHG] and [70% isopropyl], we may *prevent* [catheter-related central-line infections]. Understanding this relationship from the sentence is called relation extraction.

3. Contextual reasoning / analogy

Some sentences (phrases) may be considered analogous, i.e. they would have the same meaning, but different wording. Sentence **(A)** and **(B)** may be considered roughly analogous. Or two phrases in two sentences can be considered synonyms, e.g. “not to be use with” and “contraindicated”.

An important question in determining analogy is whether phrases are *generally analogous*, or *contextually analogous*, the first meaning that the sentences will have the same meaning, independent from the context in which they are used, while the latter would mean that they are analogous, only within a certain context.

As an example, consider sentence **(B*)**, a modification of sentence **(B)**:

“The use of chlorhexidine gluconate rather than povidone-iodine

solution for catheter site care may be an effective and simple measure for improving patient safety.” **(B*)**

Where **(A)** and **(B)** where generally analogous, **(A)** and **(B*)** are contextually analogous, e.g. in an article on catheter-related infections.

Recently, progress has been made in some of the tasks above, by measuring human disagreement arising from crowdsourcing. For more details we refer to Aroyo and Welty [1].

To investigate the use of Crowdsourcing in text understanding further, extensive research on the mix of Human (crowdsourced) tasks and Machine (automated) tasks would be necessary. However, at current, research is hampered by the lack of:

1. uniformity in interaction with the various crowdsourcing frameworks, and
2. efficiency in setting up large-scale experiments (all work and analysis is conducted manually)

Therefore, a general framework, tackling the issues of uniformity and efficiency is needed, which is the topic of the proposed B.Sc. research.

Research goals

The main topic of this research is to investigate (and implement) an efficient framework for handling microtask management. Such a framework should be able to:

- Process a –configurable- number of inputs (e.g. 1 sentence, 2 sentences, a text-fragment),
- Process the input(s) by means of a workflow, consisting of an arbitrary number of *processing nodes*, processing nodes may be *human* (e.g. crowdsourced), *machine* (thus: automated) and *other* (e.g. gaming), to cope with the "other" requirement, a standardized interface should be available towards these nodes,
- Process the output, in terms of default statistics, e.g. success rate, number of annotations per sentence, number of sentences per worker, etc.
- Finally, give a view of the outcome / results (independent of the type of processing conducted).

For any views, we may consider the use of (HTML/XML) templates.

Steps

A. Related frameworks

A first step would be to investigate the current state of the art frameworks in Crowdsourcing Microtask Management, more particularly to compare the *ManReduce* framework [2] and the *CrowdLang* framework [3] and to determine whether these frameworks could (partially) fulfill our demands.

B. Architecture

After a framework has been chosen, it has to be included into an architecture, taking into account the need for *configurability* (workflow, processing, in- /output and view-templating), and *scaleability* (potentially scaling up to a large number of tasks / interactions) and *extendibility* (the number of processing node types and in- /output types defined at current may be extended at a later stage).

C. Implementation

Finally, the designed architecture should be implemented (preferably using Java).

References

[1] Aroyo, L., Welty, C. “Harnessing Disagreement for Event Semantics”. Proceedings of the Workhop on Detection, Representation, and Exploitation of Events in the Semantic Web. 2012, Boston, USA.

[2] Salman Ahmad , Alexis Battle , Zahan Malkani , Sepander Kamvar, “The jabberwocky programming environment for structured social computing”. Proceedings of the 24th annual ACM symposium on User interface software and technology, October 16-19, 2011, Santa Barbara, California, USA

[3] Minder, P., Bernstein, A. “Crowdlang - first steps towards programmable human computers for general computation”. Proceedings of the 3rd Human Computation Workshop (HCOMP 2011). AAAI Press (January 2011)