Deliverable D2.1:Task Composition, Prioritisation, and User Profiling Models

MODUL University Vienna

WP 2 (T2.2, T2.3)

Version History			
Version	**Author**	**Date**	**Comments**
1	Marta Sabou	20/11/2013	Main Structure, Sections 1, 2
2	Marta Sabou	26/11/2013	Sections 3, 4, 5,6 added
3	Marta Sabou	27/11/2013	First Complete Draft
4	Marta Sabou	28/11/2013	Major revision of first draft
5	Marta Sabou	29/11/2013	Final revision

Validation			
Role	**Organisation**	**Name**	**Date**

Abstract This deliverable reports on work towards defining models for managing and ranking HC tasks according to dynamic priorities set by the requester (primarily in terms of cost, speed, quality), various task types and information available about users and their contexts. As such it covers work performed as part of tasks T2.2 and T2.3.

A novel feature of the uComp platform is that it spans two diverse human computation (HC) genres: games with a purpose and mechanized labour. Therefore, a fundamental question relates to the pros and cons of these two crowdsourcing genres, which we investigated through a set of experiments. Our conclusion was that these two genres are highly complementary and can be combined into hybrid-genres workflows. Since our task composition models should be able to create such hybrid workflows, we performed a feasibility study of hybrid-genres workflows and showed that they lead to several advantages over single-workflow approaches. The deliverable also presents a model for managing the HC tasks in the uComp framework, with individual sections dedicated to the main components of this model, namely: the HC task types, user and context models and the task prioritization engine.

Table of Content

1 Introduction

D2.1 is the first deliverable of *WP2: Human Computation Framework* and sums up a subset of the work performed within this WP during the first year of the project, with a particular focus on task composition and prioritisation aspects. As such we report primarily on work performed within tasks T2.2 and T2.3. Although we do not explicitly report on task T2.1 (this task will be covered by D2.3 in M34) which was in charge of designing and building a first prototype of the actual uComp framework, the work described here has been performed in close cooperation with T2.1, making sure that the two lines of work are clearly aligned.

Content and Outline. The deliverable's content is structured in 7 main sections, as follows:

Section 2 reports on work performed in order to compare the two main HC genres that the uComp framework will combine, namely GWAPs and mechanised labour.

Section 3 introduces the notion of hybrid-genres workflows and presents a feasibility study which experimentally proves the benefits of hybrid-workflows over single genre ones.

Section 4 provides an overview of the main elements of the composition model, thus providing a detailed problem description of the HC task composition and profiling aspects of uComp.

The rest of the sections, detail components of the composition model, namely the various task types (**Section 5**), the user and context models (**Section 6**) and finally the task prioritisation engine (**Section 7**). **Section 8** concludes the report with outlook on future work.

Publications. Work performed within tasks T2.2 and T2.3 lead to two publications. These are summarized as part of this deliverable in Sections 2 and 3. The publications are:

1. Marta Sabou, Kalina Bontcheva, Arno Scharl, and Michael Föls. Games with a Purpose or Mechanised Labour? A Comparative Study. In Proc. of the 13th International Conference on Knowledge Management and Knowledge Technologies (iKNOW), 2013.

2. Marta Sabou, Arno Scharl, and Michael Fölls. Crowdsourced Knowledge Acquisition: Towards Hybrid-Genre Workflows. International Journal on Semantic Web and Information Systems (IJSWIS), 9(3), 2013.

2 A Comparison of the trade-offs of crowdsourcing genres

Mechanised labour and games with a purpose are the two most popular human computation (HC) genres, frequently employed to support research activities in fields as diverse as natural language processing, semantic web or databases. Mechanised labour (MLab) is a type of paid-for HC genre, where contributors choose to carry out small tasks (or micro-tasks) and are paid a small amount of money in return (often referred to as micro-payments). Popular platforms for mechanised labour include Amazon's Mechanical Turk (MTurk) and CrowdFlower(CF) which allow requesters to post their micro-tasks in the form of Human Intelligence Tasks (or HITs, or units) to a large population of micro-workers. Games with a purpose(GWAP) enable human contributors to carry out computation tasks as a side effect of playing online games [16]. An example from the area of computational biology is the Phylo game (phylo.cs.mcgill.ca) that disguises the problem of multiple sequence alignment as a puzzle like game [8].

Research projects typically rely on either one or the other of these genres, and therefore *there is a general lack of understanding of how these two genres compare and whether and how they could be used together to offset their respective weaknesses*. For the uComp project, this is a key question to be answered as the planned HC platform aims to span these two genres.

We addressed this question through a series of methods, documented in [12]. In this section we summarize our main finding, and point readers interested in details to the published paper. Our methodology consisted on two instruments: a literature study and an experimental investigation.

As the first part of our investigations, we identified the differences between the two genres, primarily in terms of *cost, speed and result quality*, based on **existing studies in the literature**. The findings illustrated in Table 2 lead to the conclusion that there is a significant complementarity between the two genres, along all key dimensions (cost, speed, quality) and that this fact could be leveraged for building hybrid HC systems that exploit the benefits of both genres simultaneously. For example, complex, interesting tasks could be performed by a dedicated, well-trained player base (on a longer term and virtually for free), while more "boring" tasks that would reduce the motivation of players might be more suitable for execution by intrinsically motivated micro-workers, for a small amount of money.

Starting from these hypotheses above, and as a second methodological step, we aimed to quantify these genre differences through a comparative study that involved performing the same knowledge acquisition task with the Climate Quiz game (see Figure 1) on the one hand and through a similar mechanised labour interface, on the other.

Table 3 sums up our observations when comparing the two HC genres and compares them to the results in [15], a similar study that compares the two genres experimentally albeit on another knowledge acquisition task. Overall, we conclude that the study's findings demonstrate that the two genres are highly complementary, which not only makes them suitable for different types of projects, but also opens new opportunities for building cross-genre human computation solutions that exploit the strengths of both genres simultaneously.

Feature	MLab	GWAP	References
Cost			
Set-up Price	Low(+)	High(-)	[10, 15, 17]
Price per task	Low(-)	None(+)	[10, 15]
Speed			
Set-up Time	Low (+)	High(-)	[10, 15, 17]
Throughput	High(+)	Low(-)	[5]
Throughput predictability	High(+)	Low(-)	[5, 15]
Quality			
Quality	Low(-)	High(+)	[5, 17]
	High(+)	High(+)	[15]
Maintaining motivation	Easy(+)	Difficult(-)	[15]
Incentive to cheat	High(-)	(Mostly) Low (+)	[5, 17]
Task complexity	Low(-)	High(+)	[5]
Importance of task interestingness	Low(+)	High(-)	[15, 19]
Worker diversity	Low(-)	High(+)	[15]
	High(+)	Low(-)	[17]
Other			
Ethical issues	Yes(-)	(Mostly) No(+)	[6]

Table 2: Advantages and disadvantages of mechanised labour and GWAPs.

3 Hybrid-genre workflows: A feasibility study

Continuing from the conclusions of our comparative study (Section 2), we experimented with the notion of hybrid-genre crowdsourcing workflows aiming to understand (1) if these are feasible and, if yes, (2) to estimate the improvement they bring over single-genre approaches. This part of our work has been documented in a journal paper [11], and therefore this section only provides a summary of the main outcomes, while readers interested in further details are pointed to the article.

3.1 Evaluation of a single-genre approach to knowledge creation

To create a baseline of single-genres approaches, we performed an evaluation of the Climate Quiz GWAP and compared the results with those of similar knowledge acquisition games.

As depicted in Figure 1, Climate Quiz invites Facebook users and their online friends to evaluate whether two concepts presented by the system are related (e.g. *environmental activism*, *activism*), and which label is the most appropriate to describe this relation (e.g. $is\ a\ sub-category\ of$). The system controls the types of relations between concept pairs, focusing both on generic ($is\ a\ sub-category\ of$, $is\ identical\ to$, $is\ the\ opposite\ of$) and on domain-specific

	Study Observations		Thaler et al. [15]	
Feature	**CrowdFlower**	**Climate Quiz**	**MTurk**	**OntoPronto**
Cost				
Set-up Price	$450	$9,000	est. $4,500	est. $22,500
Price per unit	$0.183	$0	$0.74	$0
Speed				
Set-up Time	2 days	2 months	1 month	5 months
Throughput	243	180	-	-
Throughput predictability	within hours	completion difficult to estimate	-	-
Quality				
Precision	CF1= 59%	72%	99%	97%
	CF2= 75%	72%		
Maintaining motivation	no effort to recruit micro-workers	significant effort for recruiting players	easy (financial)	difficult
Task complexity	similar	similar	similar	similar
Importance of task interesting	micro-workers solve all tasks	players skip many tasks	-	-
Worker diversity	83	648	16	270

Table 3: Comparison of mechanised labour and games based on our observations and [15].

(*opposes*, *supports*, *threatens*, *influences*, *works on/with*) relations. Two further relations, *other* and *is not related to* were added for cases not covered by the previous eight relations. The game's interface allows players to switch the position of the two concepts or to skip ambiguous pairs.

Climate Quiz acts as a game with a purpose with the main aim of collecting knowledge assets to support an ontology learning algorithm [18]. A human-machine workflow is therefore established as depicted in Figure 2. The "machine" part of the workflow is the ontology learning algorithm that extracts terms from unstructured and structured data sources. The term pairs that are most likely related based on the algorithm's input data sources are subsequently sent to Climate Quiz, where the human element of the workflow assigns relations to these pairs. These relations are fed back into the algorithm which uses them to perfect the learned ontology and to derive new term pairs that should be connected.

Evaluation Results: Based on the evaluation of the game (see details in [11]), we conclude that while Climate Quiz has attracted a significant number of players (the highest number of all knowledge acquisition games) and managed to build a 50+ core community of players (as opposed to only 10 in PhraseDetectives), it has achieved only medium average lifetime play (ALP) values. Additionally, its throughput was the lowest of all games and so was the agreement of the game results with the gold standard dataset.

The evaluation also revealed the high difficulty of the task which we assume to be the

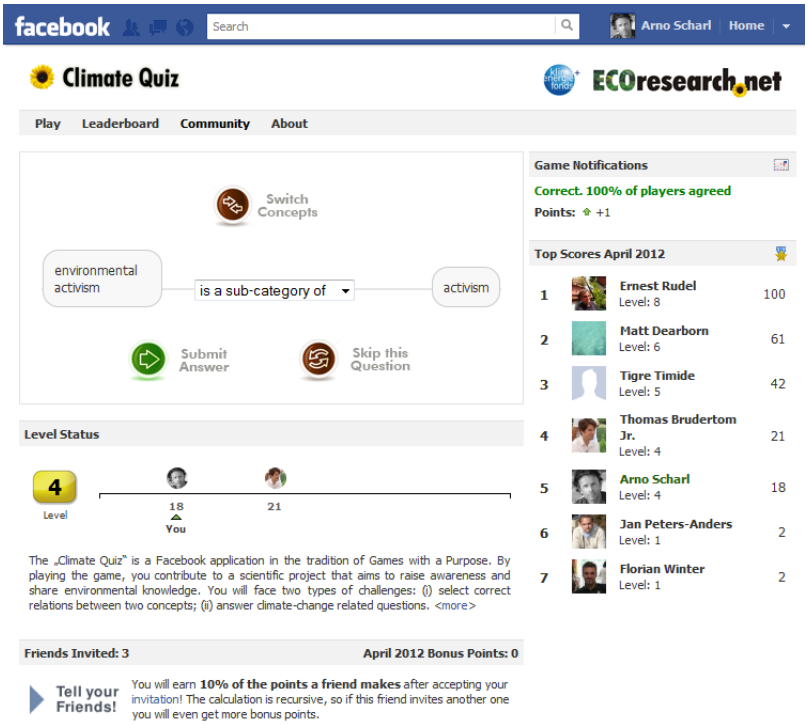

Figure 1: The Climate Quiz Interface.

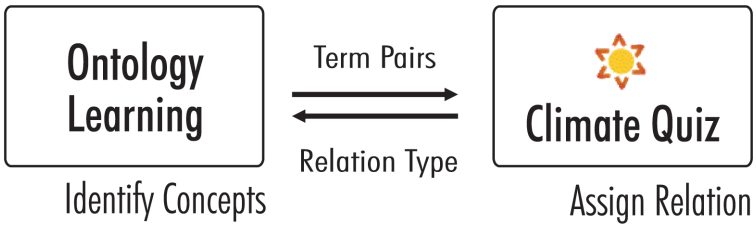

Figure 2: Human-machine workflow involving Climate Quiz and an ontology learning algorithm.

main cause of players playing the game for short intervals only (hence the average ALP) and providing results that have a low quality when compared to other games (although, in line with the quality provided by paid annotators). More specifically, we distinguish two core problematic issues that lead to the limitations of the game.

1. Firstly, the game is fed *noisy input data*, generated automatically by the ontology learning algorithm and containing terms that are ambiguous, obscure or do not make sense at all. A severe negative effect is that such confusing terms frustrate players and reduce the enjoyment of the game, which is the main motivational factor Climate Quiz relies on. Therefore, frustrated players play less (lower ALP) and are likely to lose motivation and leave the game, thus preventing the game from maintaining a stable community over long periods of time and jeopardizing its long-term success. Noisy input data also leads to wasting precious game resources (i.e., players' time and effort) on obscure terms and

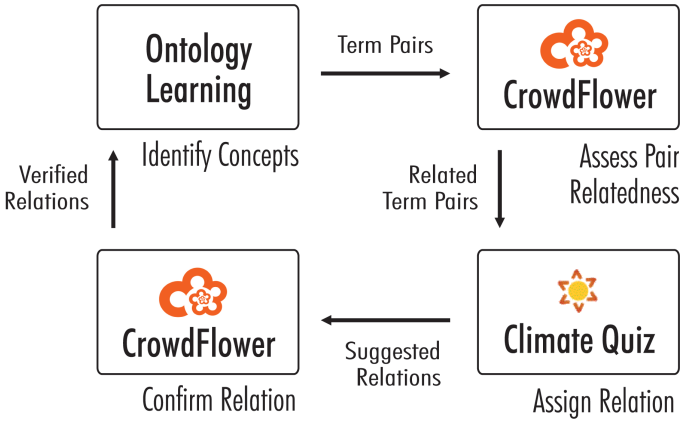

Figure 3: Hybrid-genre workflow.

inherently to low performance as disagreement tends to be high on these ambiguous pairs.

2. Secondly, the game *loses good quality output data* As discussed in [11], the high number of relations to choose from and their semantic overlap often lead to cases when a pair of terms can be correctly related with multiple relations (e.g., *threatens* and *influences*), Climate Quiz, however, derives a single relation between any input pair using a majority voting based mechanism causing that less popular but still correct relations are excluded from the final result set. For example, the game assigned the relation *works on/with* to the pair (*green industry*, *clean energy products*) as the most popular one, and therefore did not include the relation *supports*, which was the second most popular relation voted by the players and can be considered a correct relation.

3.2 Hybrid-genre crowdsourcing workflows

As a way to mitigate the problematic issues discussed above, we propose a workflow that combines two different crowdsourcing genres in order to leverage their complementary strengths, hence the term hybrid-genre workflow. In our context, and considering the pros and cons of crowdsourcing genres discussed in Section 2, we assign simple (and boring) tasks to microworkers and keep more complex (but interesting) tasks for game players thus ensuring game enjoyment and reinforcing players' intrinsic motivation. Therefore, our workflow is novel compared to the existing workflow types (described in Section 2.3 of [11]), which either relied on a single crowdsourcing genre (most frequently mechanised labour) or combined machine and human computation. Concretely, our workflow has three stages (see Figure 3).

1. **Stage 1: Judge Pair Relatedness.** This stage addresses the problem of noisy input data by asking CrowdFlower workers to check which pairs of terms extracted by the ontology learning algorithm might be related before feeding these into the game. Acting similarly as the "Find" phase of the Soylent workflow [2], this stage detects the problem instances worth investigating and therefore reduces the ambiguity of the input data. We hypothesize that this will lead to several positive effects such as (i) a more enjoyable

game resulting in higher player motivation and retention as well as (ii) higher quality game results in terms of better agreement with the gold standard.

2. **Stage 2: Assign Relation.** Climate Quiz is used in this stage to solve the complex problem of assigning one of ten relations between term pairs resulting from Stage 1. As such it corresponds to the "Fix" phase of the Soylent workflow which solves the problem instances identified in the previous Find phase.

3. **Stage 3: Check Relation Correctness.** This stage asks workers to assess the correctness of the relations assigned in stage 2 above (similarly to Soylent's "Verify" stage). As such, it should further increase the quality of the game's output but also extend it with potentially correct but rejected relations thus alleviating the problem of losing good quality output data.

By **evaluating** the precision of the results, the introduction of Stage 1 already provided an improvement of 4% over the precision obtainable by Climate Quiz alone (76% up from 72%). Stage 3 further raised the precision of the task to 78%. Therefore, the hybrid workflow, could, in principle, lead to a **6% increase in the quality of results obtainable by single-genre approaches**. Additional benefits, which were not explicitly evaluated during our experiments but need to be verified in the future include: **reducing task execution times** (as a significant number of obscure pairs are quickly filtered out by CF in Stage 1), **improving the quality of input data** and as a result providing a **more positive player experience**, which will presumably lead to **longer average lifetime play** and more contributions.

In the rest of the deliverable we focus on issues related to composing tasks and outsourcing them to diverse platforms as part of hybrid-genre workflows.

4 Composition model

In this section we provide a detailed description of the composition model in terms of its inputs and outputs. Figure 4 depicts graphically the main elements of the composition model, which are explained in detail next:

Input from Requester:

- J is a job provided by the requester, which contains N tasks;
- JPT is the number of judgments requested for each task T of job J;
- t_{dl} is the deadline for finishing a job (that is obtaining all $N * JPT$ judgements from the platform);
- $Q = \{Q_m, Q_{tv}\}$ is a tuple specifying the output quality expectations of the requester, in terms of a quality measure (Q_m) and its expected threshold value (Q_{tv}). The quality measurement can be performed in various ways, for example, (1) as the number of judgements expected per task; (2) by computing the average agreement of workers for a task (e.g., all tasks with agreement lower than a threshold value Q_{tv} should be kept in the system and more judgments should be collected for them); (3) as the agreement of the collected judgements with a gold standard, if available; (4) as the agreement with objectively verifiable questions included in the HIT. Defining and implementing quality control measures is part of task T2.4, to start in M13 of the project.
- M monetary resources as a sum of available money;
- $W = \{(W_{ch}, W_v)\}$ - a collection of objectively measurable worker constraints and their values e.g., geo location, skills, previous accuracy (on similar task types), etc;
- $mode = \{quiz, crowdflower, hybrid\}$ - the requester can choose the type of crowdsourcing genres to be used for his task; for now, we envision three possibilities: using only GWAPs ($quiz$), using only CF ($crowdflower$ - in this case, the uComp framework will act as a wrapper to CF), using hybrid-genres approaches, in which case the uComp framework dynamically decides, based on priorities, which part of the tasks is solved via games and which via CF. Current implementation covers the first two modes.

Input available in the uComp platform:

- $U = \{(U_{ch}, U_v)\}$ - a collection of objectively measurable user characteristics and their values e.g., geo location, skills, previous accuracy (on similar task types), etc;
- TT - task types (for now only classification).

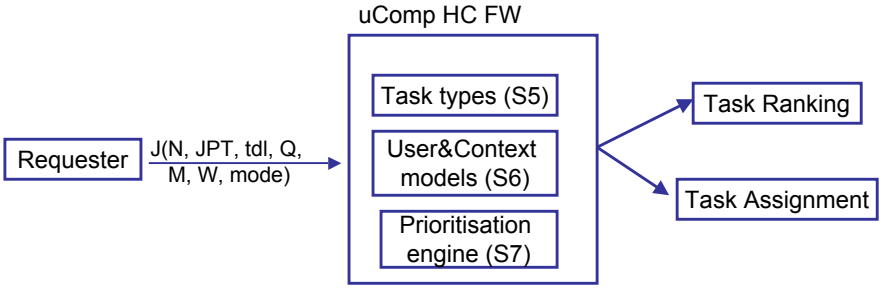

Figure 4: Overview of the composition model.

Expected Output:

1. **Ranking of HC tasks by priority**: $R = \{(T_k, P_{T_k,J_i,t_1})\}$, where each task k of each current job i, T_k, is assigned at time point t_1 a priority value P_{T_k,J_i,t_1}. The tasks with the highest priority values should be resolved first. A set of initial prioritisation algorithms are described in Section 7.

2. **Assignment of HC tasks to the appropriate platform/contributor**: each task k of each job i, T_k, is assigned to an appropriate platform Pf_m and a suitable user u_n, namely: $A = \{(T_k, Pf_m, u_n)\}$. Note that the development of matching algorithms between user profiles and task requirements is envisioned as taking part during the third stage of T2.3, and therefore will not be discussed in this deliverable.

5 HC Task Types in uComp

On a long term, the uComp framework could support a wide range of HC task types, including:

Classification tasks through which contributors select one or more values from a set of given values. Several variants of this task are discussed in Section 5.1.

Selection through highlighting tasks involve contributors being able to highlight a portion of some (text) input as a way to select it. The output of such tasks is a set of non-overlapping bounds at token/byte level. Examples of such tasks include: identification/validation of named entities (e.g., persons, locations, organisations, dates etc); named entity chunking; temporal expression annotation.

Association tasks in which relations are specified between one or more individual text snippets. Co-referencing, identifying event chains, determining the arguments of connectives (i.e., co-ordination) are some examples of NLP problems solvable through association tasks.

Subset selection tasks are characterised by having a very large set of values from which a worker selects a small number of examples, searching being dealt with as part of the task's interface. The large set may likely be the same for all tasks. The interesting part of it is typically so small that decomposing the task to binary decisions is inefficient. Examples of problems that would need such HC tasks are building a topic cloud from a large number of terms; finding documents relevant to a theme (e.g. a person, an event, a question); choose preferred items with a shared property (e.g. choosing items, perhaps graphics, and arranging them according to instructions/text information; mannequin task, scene assembly task).

Content generation tasks require contributors to create new content for example as part of translation tasks or text summarization tasks.

From the tasks types above, classification style tasks tend to be the most frequently used. Therefore, for the first version of the uComp Framework, we focused on these types of tasks as discussed in more detail in the next section (Section 5.1). Additionally, we present the API-level mapping between the uComp classification tasks and CrowdFlower HITs in Section 5.2 and summarize the implementation details of the uComp to CrowdFlower bridge, a software component that allows automatic crowdsourcing of uComp tasks through the CrowdFlower mechanised labour marketplace, thus enabling the construction of hybrid-genre workflows.

5.1 Classification Tasks

Classification style tasks allow users to select one (or more) values from a given set of values. Depending on the type of the values as well as the size of the value set, we distinguish two major categories of classification tasks.

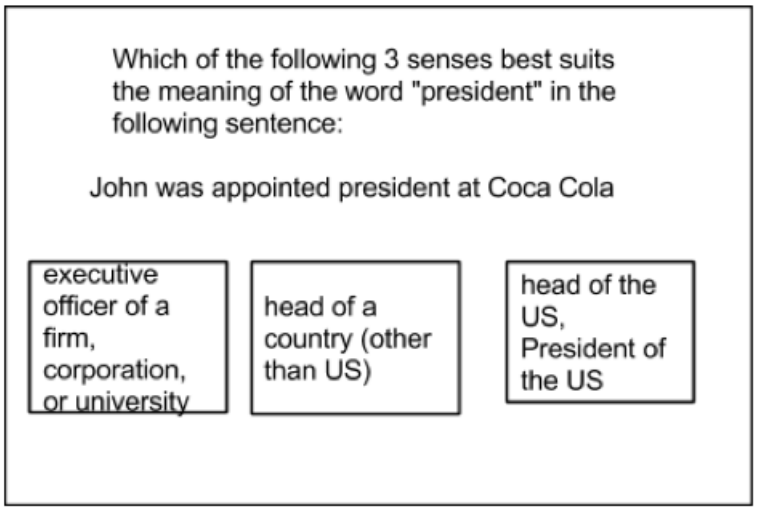

Figure 5: Interface mock-up classification task .

Selecting between multiple categories - in this case, the set of values consists of a set of discrete values. When the number of these values is two, the classification problem becomes a *Binary Choice* problem.

Selecting from a range of values - in this case, users select a point in a continuous interval of values.

To exemplify these tasks, we provide examples from the seminal work of Snow [14]. For each task type, we detail the input/output data and provide a mock-up of the expected uComp interface as well as a set of uComp API parameters that would need to be instantiated to create such a task. Identifying and detailed these HC tasks was performed with the entire consortium (lead by MOD/WU and relying on feedback from USFD/LIMSI) and helped in defining the uComp API in a way that meets the expectations of all partners[1]. We believe that this material will also be useful for documentation purposes at later stages in the project.

Task Title: TC1: Multiple Categories

Task Description Snow et al perform a word sense disambiguation (WSD) task where the workers are presented with a text snippet containing the word *president* and they have to decide which one of three possible word senses is the most appropriate to describe the word *president* in that text (see section 4.5 of [14]).

Task Interface See Figure 5

Task Input Data A set of sentences containing the word *president*:

- John was appointed president at Coca Cola company
- President Obama loves apples.

[1]The documentation of the uComp API can be found at `http://soc.ecoresearch.net/facebook/election2008/ucomp-quiz-beta/api/v1/documentation/`

- The President of Austria is the federal head of state of Austria.

Task Output Data

- the individual category choices of each worker, e.g., President Obama loves Apples, 2, 3, 3 (meaning that senses 2, 3, 3 were chosen in the case of this sentence).
- an aggregated value based on the chosen aggregation method (majority vote, weighted vote considering worker trust). Some aggregation methods might also return a confidence value. e.g., President Obama loves Apples, 3, 66%

API Call

- General
  - Title: Selecting a word sense for the word president.
  - Category: Classification
  - Instruction: Which of the following 3 senses.
- Specific
  - Default Categories: (1, "executive officer of a firm, corporation or university"), (2, "head of a country (other than US)"), (3, "head of the US, President of the US")
  - Multiple Choice: false
  - Judgements per unit: 10
  - User Characteristics (location, languages, trust): (loc, US), (lang, EN), (trust, 80)
  - Aggregation: majority vote
  - Task deadline: 30.11.2013
- Data
  - John was appointed president at Coca Cola company, [cat-1, cat-2, cat-3]
  - President Obama loves apples.
  - The President of Austria is the federal head of state of Austria.

Task Title: (TC2) Binary Choice

Task Description Snow et al (see Section 4.3 of [14]) ask workers to judge whether a sentence can be inferred logically from the other. The workers are presented with two sentences (this is the variable data) and can choose True or False depending on whether the second sentence can be inferred from the first one (i.e., it is an entailment of the first text).

Task Interface See Figure 6

Task Input Data A set of sentence pairs:

- ("Crude oil prices slump", "Oil prices drop")

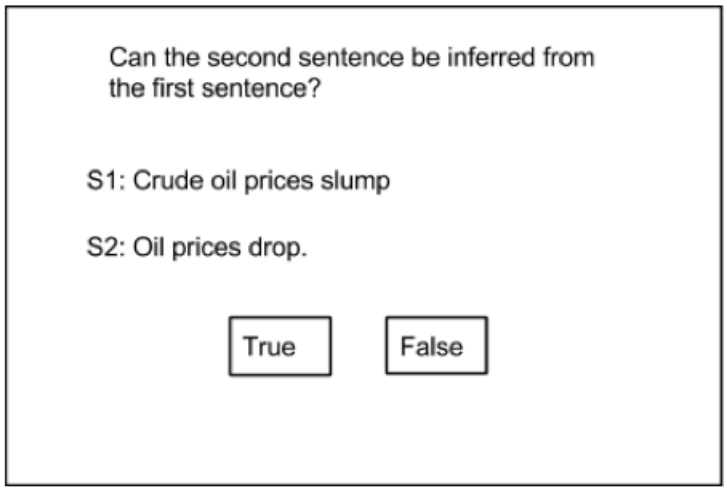

Figure 6: Interface mock-up for the binary choice task.

- ("The government announced that it plans to raise oil prices", "Oil prices drop")

Task Output Data

- the individual category choices of each worker, e.g., e.g., ("Crude oil prices slump", "Oil prices drop"), 1, 1, 2
- an aggregated value based on the chosen aggregation method (majority vote, weighted vote considering worker trust). Some aggregation methods might also return a confidence value. e.g., ("Crude oil prices slump", "Oil prices drop"), 1, 66%

API Call

- General
  - Title: Judging textual entailment.
  - Category: Classification
  - Instruction: Can the second sentence be inferred from the first one?
- Specific
  - Default Categories: (1, "True"), (2, "False")
  - Multiple Choice: false
  - Judgements per unit: 10
  - User Characteristics (location, languages, trust): (loc, US), (lang, EN), (trust, 80)
  - Aggregation: majority vote
  - Task deadline: 30.11.2013

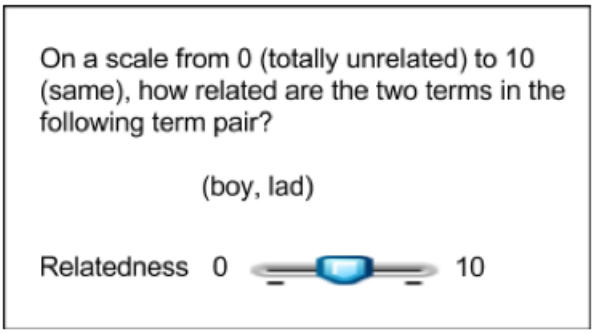

Figure 7: Interface mock-up for the simple slider task.

- Data
  - ("Crude oil prices slump", "Oil prices drop")
  - ("The government announced that it plans to raise oil prices", "Oil prices drop")

Task Title: (TC3-1) Simple Slider
Task Description Snow et al (see Section 4.2 of [14]) describe a task for judging word similarity where workers are presented with pairs of words (e.g., boy, lad, noon, string) and for each pair their have to specify how related the words are on a range of [0,10]
Task Interface See Figure 7
Task Input Data a set of term pairs:

- (boy, lad)
- (noon, string)
- (cow, grass)

Task Output Data

- the individual score selected by each worker, e.g., (boy, lad), 8, 9, 10
- an aggregated value based on the chosen aggregation method. In the case of the slider average fits better than majority vote. No confidence value (unless it is a majority vote weighted based on worker performance). e.g., (boy, lad), 9

API Call

- General
  - Title: Judging word relatedness.
  - Category: Classification - Simple Slider
  - Instruction: On a scale from 0 (totally unrelated) to 10 (same), how related are the two terms in the following term pair?

- Specific
  - Slider name: Relatedness
  - Slider Range: (0,10)
  - Judgements per unit: 10
  - User Characteristics (location, languages, trust): (loc, US), (lang, EN), (trust, 80)
  - Aggregation: average
  - Task deadline: 30.11.2013
- Data
  - (boy, lad)
  - (noon, string)
  - (cow, grass)

Task Title: (TC3-3) Multi Slider

Task Description For the task of affective task annotation (see Section 4.1 of [14]), workers are shown a headline and have to rate it on a scale of [0,100] in terms of six emotions (anger, disgust, fear, joy, sadness, surprise) as well as on a scale of [-100,100] for the overall positive or negative valence of the emotional content. e.g., "Outcry at N Korea nuclear test" could lead to the following ratings by one worker: (anger:30, disgust:30, fear:30, joy:0, sadness:20, surprise:40, Valence:-50)

Task Interface See Figure 8

Task Input Data list of sentences to be evaluated

Task Output Data

- **individual scores**: "Outcry at N Korea nuclear test", (anger:30, disgust:30, fear:30, joy:0, sadness:20, surprise:40, Valence:-50), (anger:20, disgust:50, fear:50, joy:0, sadness:70, surprise:10, Valence:-70), (anger:50, disgust:10, fear:80, joy:0, sadness:90, surprise:10, Valence:-80)
- **aggregated scores**: computed as the average of the values provided for each slider. "Outcry at N Korea nuclear test", (anger:30, disgust:30, fear:30, joy:0, sadness:20, surprise:40, Valence:-50)

API Call

- General
  - Title: Judging emotions in text snippets.
  - Category: Classification - Multi Slider
  - Instruction: For the provided text, please rate...

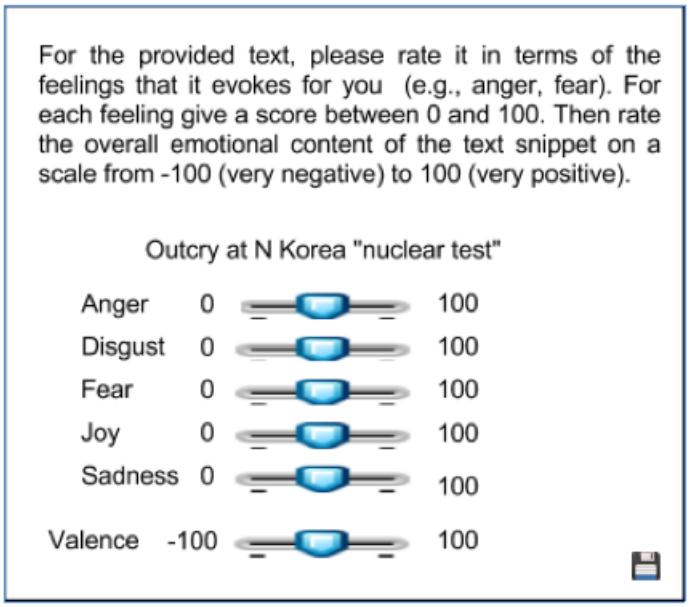

Figure 8: Interface mock-up for the multi slider task.

- Specific
  - Sliders: (anger, 0, 100), (fear, 0, 100), (joy, 0, 100),..., (Valence, -100, 100)
  - Judgements per unit: 10
  - User Characteristics (location, languages, trust): (loc, US), (lang, EN), (trust, 80)
  - Aggregation: average
  - Task deadline: 30.11.2013
- Data
  - (Outcry at N Korea nuclear test)

5.2 Mapping uComp HC tasks to the CrowdFlower API

In order to realize hybrid-genre workflows, it is important that the platform can translate game tasks into HITs on mechanised labour platforms. This task is foreseen by point (d) of T2.2: " translation of certain tasks with lower skill requirements into Human Intelligence Tasks (HITs) to be carried out through mechanised labour through marketplaces such as MTurk and CrowdFlower."

Therefore a mapping between the components of a Classification tasks and the API of a concrete platform must be clarified. We chose CrowdFlower (CF) as the concrete platform given our earlier experience with it, its reliability, its access to a large number of third-party crowdsourcing marketplaces (including Amazon Mechanical Turk - AMT) and its availability

to requestors without a US bank account (a constraint imposed by AMT). Table 18 sums up this mapping.

uComp API	**CF API**
task_title	Job title
task_type: "Classification"	cml:checkboxes
task_description	cml:checkboxes (parameter "label" for all checkboxes)
ts_default_categories	cml:checkboxes (parameter "label" for each checkbox)
ts_multiple_choice	Create different cml templates for different task modes (e.g. multiple or single choice questions)
Judgements per Unit	Jobs/ judgments_per_unit
Worker characteristics	Jobs allows specifying a desired channels (e.g., amt) and excluding/including certain countries (included_countries)
Aggregation	Various strategies described at https://crowdflower.com/docs/cml/#aggregation
task_duration	CF support only for Premium accounts

Table 18: A mapping between the uComp and the CF APIs.

5.3 uComp to CF bridge - Implementation details

The uComp API[2] was built using PHP. A uComp to CF bridge was built, which allows pushing requests to CrowdFlower(CF). This component was implemented using an Open Source PHP library, which matches commands to corresponding HTTP requests and allows to easily access the CF API[3].

At this stage it is necessary for the requester to manually specify the crowdsourcing genres used for solving his task: one can either send a job and its units to the quiz and let players solve the tasks ($mode = quiz$), or he can opt for directly sending the tasks to CF ($mode = crowdflower$). By creating a job via the uComp API and setting the mode parameter to $crowdflower$, the corresponding CF job creation call is issued. No complex parameter transformations are required given the straightforward mapping between the two APIs (see Table 18) which is handled by the uComp platform: since the uComp API was designed with the CF API in mind, they are very similar and therefore it is possible to just adapt a few parameters and use the uComp API to send jobs to CF, upload units and receive results. When a job is created on CF the id of the created job is returned.

Data to the newly created job is sent as a CSV file via the uComp API. If the mode parameter is set to $crowdflower$, the uploaded CSV file will be slightly modified: Headers are added and if the user has specified to include gold standard in the CSV file, the last column of the file will be marked as gold data. In addition a CML template will be created that uses

[2] `http://soc.ecoresearch.net/facebook/election2008/ucomp-quiz-beta/api/v1/documentation/`
[3] `https://github.com/supertom/php-crowdflower`

the newly created headers as placeholders for the data. If the user chooses to send data to CF all information about the job will be also stored in the uComp database in order to provide all functionality of the API and to create an automatic logging function of the activities.

The pausing and resuming of a job, as well as the retrieving of results in CSV format are straightforward and require just the *jobid* as a parameter.

6 User and context models

The content of this section covers the work envisioned by task *T2.3: User Profiling and Contextualisation*. Although T2.3 only started in M11 of the project, that is, one month prior to finalizing this deliverable, the uComp HC framework was designed in task T2.1 in such a way that it catered for acquiring a variety of user related information. In this section, so far we can report on:

- an overview of the information expected to be covered by the user and context models, based on what is already collected by the uComp framework and a range of derived information which will be computed once the platform is operational;
- a list of ontologies that could be re-used for modeling this information.

The actual ontological modeling of uComp specific user and context models will be performed after the submission of this deliverable, as part of T2.3.

6.1 Content of the User and Context Models

Table 22 sums up the various user and context information provided currently by the uComp platform and CF. The uComp information is gathered through the Facebook API from the information made public by the user, and which he consents to share with the uComp games when he first installs them as one of his Facebook applications. We have divided the available data into various categories, including a digital profile, personal data, context information, skills, community data and game/session related data. From this table, it is apparent that through the uComp platform we can access much richer game play data that allows building complex models of user behavior over time.

Additionally to this data, the uComp platform will monitor skill/performance indicators as follows:

- user performance on certain task types (e.g., classification vs. co-reference);
- user performance on specific tasks: e.g., sentiment evaluation, relation detection, quiz answering;
- user performance on specific domains: e.g., climate change, medical, general knowledge, finance, sports etc
- overall user performance, e.g., overall trust levels;
- average speed per task;
- availability - when was the user last seen online? what is the probability of the user coming online before the task deadline, based on his activity history?

These indicators will be derived after the launch of the platform, once sufficient user data has been gathered.

uComp API	**CF API**
Digital Profile	
Facebook ID URL of Facebook profile email	CF specific worker ID external ID last IP address recruitment channel
Personal Data	
name gender birthday	
Context Information	
location locale	geographical location (country, region, city)
Skills	
spoken languages	
Community	
list of facebook friends	
Game/session related information	
first/last login (date) number of visits scores per months nr. of right, wrong, unclear answers nr. of skipped relations min, max, avg times spent on tasks	trust level = precision of answering gold units submission rate

Table 22: User and context information provided by the uComp framework and CF.

6.2 Candidate Ontology Models

An initial review of existing ontology models, revealed the following models as providing a potential basis for building the uComp user and context models that would cover the information described in the previous section:

Human Computation Ontology [4] Although not primarily focused on user profiling aspects, this ontology conceptualizes the connection between contributors and crowdsourcing tasks. It also relies on the Provenance Ontology (PROVO)[5] to describe the provenance of each contribution. See [4] for a description of this ontology.

SmartProducts user model [6] focuses on supporting ambient intelligent applications and therefore could contain reusable ontological models for specifying context information.

FOAF - Fried-of-a-friend [7] is a good candidate to be used for specifying personal information as well as links (relations) between players.

SIOC - Semantically-Interlinked Online Communities [8] provide ontological models for describing forums and social networking sites, including posts. It could be valuable at a later stage, if uComp decides to also record/archive relevant user content from social networking sites.

Future work will focus on a detailed analysis of these ontologies and their reuse and extension towards an ontology based user model that is tailored to the needs of the uComp framework.

[4] http://swa.cefriel.it/ontologies/hc.html
[5] http://www.w3.org/TR/prov-o/
[6] http://projects.kmi.open.ac.uk/smartproducts/ontologies/v2.6/
[7] http://xmlns.com/foaf/spec/
[8] http://rdfs.org/sioc/spec/

7 Prioritisation algorithms

Although many aspects of the composition model have not yet been addressed (e.g., quality control measures), we hereby provide a set of prioritisation algorithms that primarily focus on finalizing the given jobs by the given deadline. We start with an overview of related work before presenting these algorithms.

7.1 Overview of related work

There is a rich literature on approaches for assigning and scheduling crowdsourcing tasks under some constraints. For example, Minder et al [9] propose CrowdManager, a framework for optimizing the price and task allocation based on time, quality and budget constraints specified by the requester. Singer et al [13] focus on a pricing mechanism that 1) maximizes the number of tasks achievable within a budget and 2) minimizes payments for the tasks.

If the simplifying assumption is made that the worker's skills, speed and expected quality are known, then the assignment problem becomes a classical assignment problem solvable through linear programming, as has been done by Minder et al, who used an integer programing solution to the problem [9]. In a realistic crowdsourcing setting, however, there are several major challenges, from which we mention two:

1. **Workers arrive in an online fashion** in any realistic crowdsourcing setting and this makes assignment and planning of tasks difficult. To solve this problem some approaches opt for strategies to manage crowd-latency. For example, [9] relies on a **retainer model** in which case a set of workers are paid a low price to be available for whenever the tasks are posted. Such a retainer model approach has been originally introduced by [3]. For now, there is no retainer model planned for uComp, therefore, task assignment will be performed among players that are currently online.

2. **There is little/no knowledge about workers** - crowdsourcing markets maintain little information about their workers, although prioritisation approaches require information about the speed, quality and acceptable costs of workers. To overcome this lack of information, various approaches implicitly ask workers to bid for tasks (e.g., specify their acceptable costs) as well as to perform some sample tasks in order to estimate their potential speed and expected quality [9, 13]. Other authors, more realistically, create worker models which they try to estimate from historic data gathered about the users [1] [7]. This approach seams to be more feasible for uComp and will be considered in the later stages of task T2.3 when developing algorithms for matching between user profiles and task requests.

7.2 Mechanisms for task prioritisation

We define t_0 as the starting time of J_b, t_1 as the current time ($t_1 > t_0$) and t_{dl} as the deadline for the job.

At time point t_1:

- the time left to deadline is $t_{dl} - t_1$
- the number of missing judgments is the difference between the number of expected judgements for this job ($N * JPT$) and the judgments gathered so far (J_{J,t_1}), namely: $N * JPT - J_{J,t_1}$
- the completion speed Cs_{t_1} is $\frac{J_{J,t_1}}{t_1 - t_0}$

We can then compute an estimated time to completion ($T_{tc@t_1}$) as:

$$T_{tc@t_1} = \frac{N * JPT - J_{J,t_1}}{Cs_{t_1}} \tag{1}$$

A job's J_i priority at t_1, is the ratio of the estimated time needed for completion and the actual time available.

$$P_{J_i,t_1} = \frac{T_{tc@t_1}}{t_{dl} - t_1} \tag{2}$$

Ideally this ratio should always be < 1. Jobs should be ranked based on this priority. The priority of all jobs will be recomputed at given intervals of time. The frequency of this recomputation will be established experimentally.

Within a particular job J_i, individual tasks have their own priority level depending on:

1. the number of judgments they gathered so far. There are two options here: a) to prioritise those tasks that already have some of the required judgments in an effort to elicit all needed judgements or b) to prioritise those tasks that have no judgements yet in order to gather some judgments for them. We adopt the first approach and make task priority directly proportional to the completion rate of the task T_k at time t_1, which is $CR_{T_k@t_1} = \frac{JPT_{T_k@t_1}}{JPT}$, i.e., the ratio of judgments gathered for T_k at time t_1 and the expected judgements pert task.

2. the agreement level of the judgments gathered so far (for those tasks where inter-worker agreement can be computed). Tasks where the disagreement is high should be given higher priority than those for which an agreement seams to emerge already. Therefore, in the case of a classification style task, the task priority is indirectly proportional with the maximum inter-worker agreement achieved for one of the c categories of the classification task, that is $\max_{l=1,c}(IAA_{cat_l})$.

Therefore, for each task T_k of a job J_i at time t_1, the individual task priority can be computed as:

$$P_{T_k,J_i,t_1} = P_{J_i,t_1} + \frac{CR_{T_k@t_1}}{1 + \max_{l=1,c}(IAA_{cat_l})} \tag{3}$$

If there are no judgments available for T_k then its priority is identical to the job's priority.

Assignment to appropriate platform. If the work mode is set to *quiz*, then the above algorithm can be used to prioritise the tasks. High priority tasks can be then assigned to workers that have a higher speed, however, user assignment is still future work (T2.3).

Should the work mode be set to *hybrid*, and, if the requestor has monetary resources, then high priority tasks (especially those with priority >1) should be outsourced to CrowdFlower entirely or partially. This algorithms are also part of future work.

8 Future Work

This deliverable presented an overall model for task prioritization and reported on completed (or ongoig) work related to several components of the model, including the expected task types, the user and context models as well as initial task ranking mechanisms. Future work is envisioned towards finalizing all elements of the model, as follows:

1. Implementing the task ranking models above and evaluating them experimentally. This basic ranking models should then be extended with the components that will be developed in the future, for examples, the ones mentioned next.

2. Finalising the ontology based modeling of the user and context models as part of T2.3.

3. Defining and implementing quality control measures is part of task T2.4, to start in M13 of the project.

4. Implementing the *hybrid* mode of the framework, which will allow tasks to be seamlessly outsourced through different genres based on their priorities (T2.1/T2.2).

5. Defining and implementing mechanisms for matching task requirements and user profile/context (T2.3).

References

[1] David F. Bacon, David C. Parkes, Yiling Chen, Malvika Rao, Ian Kash, and Manu Sridharan. Predicting Your Own Effort. In *Proc. of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '12, pages 695–702, 2012.

[2] M. S. Bernstein, G. Little, R. C. Miller, B. Hartmann, M. S. Ackerman, D. R. Karger, D. Crowell, and K. Panovich. Soylent: A Word Processor with a Crowd Inside. In *Proc. of the 23rd ACM Symposium on User Interface Software and Technology*, 2010.

[3] Michael S. Bernstein, Joel Brandt, Robert C. Miller, and David R. Karger. Crowds in Two Seconds: Enabling Realtime Crowd-powered Interfaces. In *Proc. of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 33–42. ACM, 2011.

[4] I. Cellino. Geospatial Dataset Curation through a Location-based Game. *Semantic Web Jornal, Accepted for publication, Available at http://www.semantic-web-journal.net/content/geospatial-dataset-curation-through-location-based-game-0*.

[5] J. Chamberlain, K. Fort, U. Kruschwitz, M. Lafourcade, and M. Poesio. Using Games to Create Language Resources: Successes and Limitations of the Approach. In I. Gurevych and K. Jungi, editors, *The People's Web Meets NLP. Collaboratively Constructed Language Resources*. Springer, 2013. To Appear.

[6] K. Fort, G. Adda, and K.B. Cohen. Amazon Mechanical Turk: Gold Mine or Coal Mine? *Computational Linguistics*, 37(2):413 –420, 2011.

[7] John Joseph Horton and Lydia B. Chilton. The Labor Economics of Paid Crowdsourcing. In *Proc. of the 11th ACM Conference on Electronic Commerce*, EC '10, pages 209–218. ACM, 2010.

[8] A. Kawrykow, G. Roumanis, A. Kam, D. Kwak, C. Leung, C. Wu, E. Zarour, and Phylo players. Phylo: A Citizen Science Approach for Improving Multiple Sequence Alignment. *PLoS ONE*, 7(3):e31362, 2012.

[9] P. Minder, S. Seuken, A. Bernstein, and M. Zollinger. CrowdManager - Combinatorial Allocation and Pricing of Crowdsourcing Tasks with Time Constraints. In *Workshop on Social Computing and User Generated Content in conjunction with ACM Conference on Electronic Commerce (ACM-EC)*, 2012.

[10] M. Poesio, U. Kruschwitz, J. Chamberlain, L. Robaldo, and L. Ducceschi. Phrase Detectives: Utilizing Collective Intelligence for Internet-Scale Language Resource Creation. *Transactions on Interactive Intelligent Systems*, 2012. To Appear.

[11] M. Sabou, A. Scharl, and M. Föls. Crowdsourced Knowledge Acquisition: Towards Hybrid-Genre Workflows. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 9(3), 2013.

[12] Marta Sabou, Kalina Bontcheva, Arno Scharl, and Michael Föls. Games with a Purpose or Mechanised Labour? A Comparative Study. In *Proc. of the 13th International Conference on Knowledge Management and Knowledge Technologies (iKNOW)*, 2013.

[13] Yaron Singer and Manas Mittal. Pricing Mechanisms for Crowdsourcing Markets. In *Proc. of the 22nd International Conference on World Wide Web*, WWW '13, pages 1157–1166, 2013.

[14] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and Fast—but is it Good?: Evaluating Non-Expert Annotations for Natural Language Tasks. In *Proc. of EMNLP*, pages 254–263, 2008.

[15] Stefan Thaler, Elena Simperl, and Stephan Wölger. An Experiment in Comparing Human-Computation Techniques. *IEEE Internet Computing*, 16(5):52–58, 2012.

[16] L. von Ahn. Games With a Purpose. *Computer*, 39(6):92 –94, 2006.

[17] A. Wang, C.D.V. Hoang, and M. Y. Kan. Perspectives on Crowdsourcing Annotations for Natural Language Processing. *Language Resources and Evaluation*, 47(1), 2013.

[18] Gerhard Wohlgenannt, Albert Weichselbraun, Arno Scharl, and Marta Sabou. Dynamic Integration of Multiple Evidence Sources for Ontology Learning. *Journal of Information and Data Management*, 3(3):243–254, 2012.

[19] L. Wolf, M. Knuth, J. Osterhoff, and H. Sack. RISQ! Renowned Individuals Semantic Quiz - a Jeopardy like Quiz Game for Ranking Facts. In *Proc. of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 71–78. ACM, 2011.