ABSTRACT

Crowdsourcing and Human-Computation are paradigms that are enabling new forms of collaboration between humans and computers. However when developing such a system there aren't many guidelines and it's known if there are benefits of applying both paradigms to the same problem. In this paper we explore what considerations need to be taken into account when developing a Crowdsourcing and a Human-Computation system. Then those considerations are used to analysis a new system proposal (Capturista Sobre Ruedas). And by using this technique the team was capable of doubling their knowledge about the system saving time, Money and frustration.

Keywords— Crowdsourcing, Human-Computation, Book Digitalization

———————— ◆ ————————

ACKNOWLEDGMENTS

I want to acknowledge the financing provided by the Consejo Zacatecano de Ciencia y Tecnología (COZCyT - Zacatecas, México) and from the Centro de Investigación en Matemáticas (CIMAT - Guanajuato, México) for the realization of this report.

Also I want to thank Dr. Hugo Mitre and José G. Hernández, MTI for their guidance and openness to consider a research in this area.

To my family my parents (Alejandro y Martha) to my brother Juan A. García for providing the initial motivation for the project Capturista Sobre Ruedas and to Claudia Huitrado for her continuous support.

TABLE OF CONTENT

TABLE OF FIGURES

TABLE OF TABLES

1. INTRODUCTION

Crowdsourcing and the related concept human-computation are enabling new forms of collaboration between humans and computers. Normally a human would use the computer to answer a problem however with Human-Computation a computer would use the help of a human to solve the problem and with Crowdsourcing a human will ask a set of humans (the crowd) the solution to a problem. However these paradigms are still evolving and there are unanswered questions about them:

1. For starters it seems there isn't a clear distinction between both paradigms. Are they really different? [1]
2. What considerations, must be taken into account when developing a Crowdsourcing system? [2]

The authors are working on Crowdsourcing system called: Capturista Sobre Ruedas (**CSR**) which will allow people with disabilities to digitize big government archives from their homes. Since this system has element of Crowdsourcing and Human-Computation, is worth pondering:

3. Can both paradigms be applied to the same problem?
4. And does applying both paradigms to the same problem actually provide any benefits?

The answer to the proposed questions would help in:
Clarifying the distinction between Crowdsourcing and Human-Computation
Reduce risk in building a Crowdsourcing system by identifying risk factors early in the process.
Identify if there is value in applying both paradigms to the same problem.

In the following sections is exposed: 2. what is Crowdsourcing, considerations and criticisms. 3 what is Human-Computation, categories, considerations, criticisms and examples. 4. Comparison between the two: similarities and differences. 5 CSR example: context, model of risk, how challenges and criticisms of the different models apply to it.

2. BACKGROUND IN CROWDSOURCING

2.1 Definition

Crowdsourcing is a contraction of the words: Crowd and Outsourcing. The term itself was coined by Jeff Howe in "The Rise of Crowdsourcing"[3] Howe defined Crowdsourcing as: *"...the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call."*

However Crowdsourcing is becoming an umbrella term for different forms of collaboration online and as such, it is difficult to define and is in risk of becoming meaningless as noted in: "'Crowdsourcing' is a relatively recent concept that encompasses many practices. This diversity leads to the blurring of the limits of Crowdsourcing that may be identified virtually with any type of Internet-based collaborative activity"[1].

For that reason, Estellés González went ahead and analyzed over 40 different definitions of Crowdsourcing in the literature and created an integral definition:

"Crowdsourcing is a type of participative online activity in which an individual, an institution, a non-profit organization, or company proposes to a group of individuals of varying knowledge, heterogeneity, and number, via a flexible open call, the voluntary undertaking of a task. The undertaking of the task, of variable complexity and modularity, and in which the crowd should participate bringing their work, money, knowledge and/or experience, always entails mutual benefit. The user will receive the satisfaction of a given type of need, be it economic, social recognition, self-esteem, or the development of individual skills, while the crowdsourcer will obtain and utilize to their advantage that what the user has brought to the venture, whose form will depend on the type of activity undertaken" [1]

The second definition although thorough is too wordy to develop an intuitive understanding of Crowdsourcing that's why in the following section several examples are presented.

2.2 Examples

The following table presents the Crowdsourcing taxonomy as was defined by the http://crowdsource.org industry council[4].

Category	Definition	Example	Details
OPEN INNOVATION	Use of sources outside of the entity or group to generate, develop and implement ideas.	http://innocentive.com	Site where companies post problems for rewards and scientist from all over the world try to solve it.
COMMUNITY BUILDING	Development of communities through active engagement of individuals who share common passions, beliefs or interests.	http://Ushahidi.org	A map generator for support effort in Haiti or other disaster zones.
COLLECLECTIVE CREATIVITY	Tapping of creative talent pools to design and develop original art, media or con-	http://threadless.com http://istockphoto.com	Design of T-shirts and the crowd votes on the best ones. Sale of stock photos

	tent.		
CIVIC ENGAGEMENT	Collective actions that address issues of public concern.	http://citizentube.com	After of state of the nation president Obama answered questions from YouTube citizens.
COLLECLECTIVE KNOWLEDGE	Development of knowledge assets or information resources from a distributed pool of contributors.	http://wikipedia.org	The open encyclopedia.
CROWDFUNDING	Financial contributions from online investors, sponsors or donors to fund for-profit or non-profit initiatives or enterprises.	http://kiva.org http://kickstarter.org	Loans for the poor financed by the crowd. Financing of tech prototypes
CLOUD LABOR	Leveraging of a distributed virtual labor pool, available on-demand, to fulfill a range of tasks from simple to complex.	http://mturk.com	Microtasks divided among humans for a fee.

Table 1 Crowdsourcing taxonomy plus examples.

2.3 Considerations when developing

What are the factors one should consider when developing a Crowdsourcing platform? Doan et al. [5] propose to consider the following dimensions:

1. Nature of collaboration

Refers to the kind of collaboration the users are performing. It can be explicit: the user knows she is part of the crowd (i.e. Amazon Mechanical Turk) or Implicit the user doesn't know he is in fact collaborating with Crowdsourcing effort (i.e. ESP Game).
In Explicit systems the users can: users can evaluate, share, network, build artifacts, and execute tasks.

2. Type of Target Problem

Goal of the system as defined by its creators.

3. How to recruit and retain users

Recruiting users for the crowd is one of the most important tasks in Crowdsoucing the major solutions to this problem are:

- Require users to do it.
  If they are employees you can order them to participate in the crowd.
- Pay users.
  What Amazon Mechanical Turk does.
- Volunteers.
  For example all the contributors to Wikipedia.
- Make the users work for a service.
  For example to post a comment on a blog a user must solve reCaptcha. And the reCaptcha is used to digitalize books.
- Piggyback.
  I.e. use the user's interactions with other established system to solve a Crowdsourcing problem.

Once the recruiting of the crowd has been completed, the Crowdsourcing System must also help with retention. The most common ways to retain users are:

- Provide instant gratification
- Provide an enjoyable experience or providing a necessary service
- Provide ways to establish a reputation
- Provide ownership situations, where a user may feel he or she "owns" a part of the system

4. What contributions can users make?

The contributions is the reason why the Crowdsourcing platform is built. They depend on the designer of the system. The Contributions can be as simple as: evaluate, users review, rate or tag. Can be medium complexity as contributing photos to iStockPhoto, designs to Threadless. And really complicated as scientific processes submitted to Innocentive or software solutions to TopCoder.

5. How to combine contributions?

The combination mechanisms can vary from the manual combination such as rating from other humans to the automatic combination of results. The automatic combination of results frequently involves some form of statistical aggregation.

6. How to evaluate users?

A Crowdsourcing system frequently needs to deal with malicious members of the crowd. When designing the system development must consider techniques that block, detect, and deter malicious members of the crowd from damaging the system.

7. Degree of manual effort

It's to the creators of the Crowdsourcing system to decide how much manual effort must be invested in the task it can vary from he extremely easy as casting a vote to the extremely complicated like folding a protein i.e. the http://fold.it game.

8. Role of human users

There are basically four roles for humans in the crowd[5]:

- Slaves:
  Humans help solve the problem in a divide-and-conquer fashion.
- Perspective providers:
  Humans contribute different perspectives, which

when combined often produce a better solution (than with a single human).
- Content providers:
  Humans contribute self-generated content (for example, videos on YouTube, and images on Flickr).
- Component providers:
  Humans function as components in the target artifact, such as a social network, or simply just a community of users (so that the owner can, say, sell ads).

9. Standalone vs. Piggyback architecture

An Standalone system is one that is built explicitly to be used for the crowd to collaborate examples of this are Amazon Mechanical Turk or Wikipedia. A Piggyback system is one where the users already use the system for other purposes and the developers exploit this information for other purposes.

2.4 Criticisms

Crowdsourcing criticisms can be classified in two groups, problems to the organization creating the work (crowdsourcer) and problems for the members of the crowd performing the task (crowdworkers).

Of the crowdsourcers:

CS 1. Ethical Concerns:
In sites such Amazon Mechanical Turk workers are making less than the minimum wage even in India. [6]

CS 2. Increased risk of not finishing on time.
If a project doesn't generate enough interest of the crowd it might be that some elements of the work actually don't get done. [7]

CS 3. Risk of low quality work.
Since tasks are paid on completion, quality might not be taken care of by the crowdworker. And then mechanisms such as verification or rework make the task more expensive. [8]

CS 4. Typically no confidentiality agreements.
Normally the members of the crowd don't sign any form of a contract which might present some liability problem for the crowdsourcer.

Of the crowdworkers:

CW 1. Below minimal wage income.
It has been researched that as Crowdsourcing gets more popular workers are making less and less money even to the point of making below minimum wage in India.[6]

CW 2. No form of work
contract or work stability. Since there are no contracts basically is on the discretion of the crowdsourcer to decide if pays for the work.[9]

CW 3. No communication with other members of the crowd.
Crowdworkers tend to work individually. This leads to an isolating experience as the examples from http://innocentive.com and the Netflix challenge shows. [10] [11]

3. Background in Human-Computation

3.1 Definition

The first use of the term Human-Computation seems to be the 2005 Luis Vohn Ann thesis with the same title: Human Computation. Ann defines Human-Computation as: *"...paradigm for utilizing human processing power to solve problems that computers cannot yet solve."* [12] This tasks often include: seeing what is an image, understanding a paragraph, or simply knowing common facts.

3.2 Examples

This is an example of a Human-Computation program using the TurKit framework [13]. It has 2 human tasks:
: a) generating five ideas for things to see in New York City, and b) sorting he list by getting workers to vote between ideas.

```
var ideas = []
for (var i=0; i < 5; i++) {
        ideas.push(mt.promptMemo(
"What's fun to see in New York City?"))
}

ideas.sort(function(a,b) {
        return mt.voteMemo(
                "which is better?", [a, b]) == a ? -1:1
}
```

Figure 1 A TurnKit script for a Human Computation algorithm.

The following table presents examples of Human-Computation systems.

Name of the System	Description
ESP Game	A collaborative game where humans tag images to help search engines.
http://reCAPTCHA.net	A login system that at the same time allows millions of user digitalize books.
Biomorphs software	A Genetic algorithm in which the fit function is set by a human.
http://fold.it	A system in which user try to 'fold' proteins to see if they can combine in a certain way.

Table 2 Examples of Human-Computation systems.

3.3 Considerations when developing a Human-Computation system

The following list is compiled from the Human-

Computation analysis dimensions proposed by [14].

1. What is the motivation?
   Reasons a person has to collaborate in this process. This could be: Pay, Altruism, Enjoyment, Reputation, Implicit work.

2. How is Quality going to be controlled?
   How is the owner of the system verifying that the quality of the human tasks is up to par to expectations. There are several mechanisms: Output agreement, Input agreement, Economic models, Defensive task design, Redundancy, Statistical filtering, Multilevel review, Automatic check, and Reputation system.

3. How are contributions going to be aggregated?
   How is the system going to organize, collect all the contributions from the different humans: Collection, Wisdom of crowds, Search, Iterative improvement, Genetic algorithm, None.

4. What Human Skill is needed to perform the task?
   Abilities that humans will use to do the task: Visual recognition, Language understanding, Basic human communication.

5. What is the Process Order?
   Collaboration order between Humans and Computers.

6. What is the Task-Request Cardinality?
   Depending on the nature of the problem the owner of the system may require the help of only one person to perform the task or many. This enables the following cardinality relationships: one-to-one, many-to-many, many-to-one, and few-to-one.

3.4 Criticisms

1. It's difficult to write code.
   When some of the functions, are done by humans How do you represent them in your code? Common approaches are functions that abstract displaying a user interface to interact with a human. An example of this approach is the TurKey [13] in which Human-Computations are abstracted as simple function calls.

2. It's difficult to debug.
   Most programming tasks require several attempts to get it right. However how is it possible to attempt to run the same program several times, if on each attempt a person must be there to provide input.

3. It's difficult to test.
   Related to the previous problem. Unit Testing is the most common used technique used for software quality assurance. The same problem arises if on each test run a person must be in front of the computer to provide input.

4. Long lived processes.
   When running a Human-Computation program over a moderated size data set the program might run for days or weeks. In that case simply turning off a server or an electrical failure can waste several days of work.

4. COMPARISON OF BOTH PARADIGMS

As seen it's difficult to draw a line to distinguish between the Human Computing paradigm and Crowdsourcing. However Quinn and Bederson[14] make the distinction clear: *"...Human-Computation replaces computers with humans, Crowdsourcing replaces traditional human workers with members of the public."*

The following figure shows the relationship between both concepts:

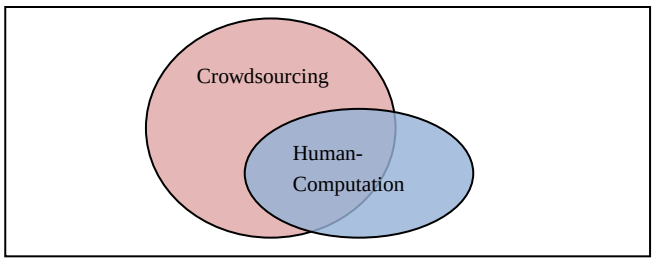


Figure 2 Reltionship of Crowdsourcing and Human-Computation

From the diagram we see there are 3 subsets of systems:

1. Human-Computation system's that aren't Crowdsourcing.

In this set we find the first works on Human-Computation. The clearest example is: "**human-based genetic algorithm** (HBGA) is a genetic algorithm that allows humans to contribute solution suggestions to the evolutionary process. For this purpose, a HBGA has human interfaces for initialization, mutation, and recombinant crossover. As well, it may have interfaces for selective evaluation. In short, a HBGA outsources the operations of a typical genetic algorithm to humans."[15]

2. Crowdsourcing systems that aren't Human-Computation.

In this category we have systems that allow persons to collaborate but that aren't related to computing tasks such as:

- Social computing:
  Blogs, forums, etc. The distinction between social software and Human-Computation is that social computation facilitates natural human behavior whereas in Human-Computation the interaction is directed by the Human-Computation system.

- Data mining:
  In general the use of data mining software does not encompass the collection of the data, whereas Hu-

man-Computation does.

3. Human-Computation that are also Crowdsourcing. Most of the Human-Computation systems are also Crowdsourcing systems because normally with Human-Computation we want to process a large amount of data. Most of the Crowdsourcing systems fall into this category such as the ESP Game mentioned or the fold.it games mentioned before.

5. THE PROPOSED SYSTEM: CAPTURISTA SOBRE RUEDAS (TYPISTS ON-WHEELS)

5.1 Context of the Problem

There is global tendency to digitize the paper archives, and books. However in order for this information to be useful the contents of the book also need to be processseed with Optical Character Recognition (OCR). OCR is the technology that allows a computer to decode text embedded in an image, however this method isn't 100% reliable and it is needed so that the text is searchable. However OCR isn't reliable in most cases, so there is a need for human verification.

Nevertheless using humans to verify and correct the errors of OCR is expensive, so expensive in fact that frequently this task has been outsourced to development countries were low salaries make digitalizing books more affordable. Could there be a way to provide digitalizing service's inside the country that is also cost effective?

5.2 Proposed solution

"Capturistas Sobre Ruedas" is a company whose mission is: "to provide dignify employment for persons with disabilities while digitalizing government paper archives". So our proposed solution is:

1. Digitize books
2. A computer OCR them,
3. humans (*with disabilities*) verify and correct the output of the system and
4. finally assembly a book.

The benefits for the crowdworkers are that they:

1. Can work from home
2. Save money from transportation and
3. Save time which allows them to have more available working hours.
4. They are free to work in their own time and decide how much they want to make.

The benefits for the crowdsourcer are that they:

1. Can pay market salaries (which tend to be low because digitalization services are considered low value added).

2. They get an "elastic" work force that can grow according to the size of the project.

5.3 Human Resources Considerations

The plan is to pay the crowdworkers an amount for each page verified and typed. This will allow them to work at any time and in any amount they want, and will let the crowdsourcer pay proportionally.

Now this raises another concern: How can payment be done in a variable way? So that crowdworkers can be paid fairly and legally. Fortunately the Federal Work Bill (Ley Federal del Trabajo)[16] in its article 89, provides the legal framework under which an employee can be paid according the units produced.

Payments in a crowdsourcing environment are contentious item. Since in the current market places the payment is set according to the desire of the crowdsourcers they tend to be below minimum wage, even for development economies such as India. [7]

So How much money does a crowdworker in Capturista-Sobre-Ruedas stands to make? From a personal interview with a Director of a Software division. This division was incubated by CIMAT and has participated in digitalization projects. From one of their projects I got the following data:

It got mxn $3,500,000 for digitalizing 5,000 books.
That translates into: $3,500,000 mxn / 5000 books = $700 mxn / book.
Each book has 300 pages so that is: $700 mxn / $300 pags. = $2.33 mxn /pag. That is the cost for the customer so now lets assume that the crowdworker receives half of that as compensation = $1.17 mxn /pag.

So how many pags can a fulltime crowdworker do? According to the Editorial Freelancers Association [17] a person indexing (reading the document and typing an index) can do 8 – 20 pags/ hour. So fulltime worker may produce 8-20 pags/hour * 8 hours /day = (64 – 120) pags / day. Finally 64-120 pags / day * $1.17 mxn/pag. = $74.67 - $186.67 mxn / day.

So a crowdworker working full time $8 hours per day stands to make from $74.67 mxn to $186.67. If we consider that the average minimum wage for workers in Zacatecas [18] is: $59.08 that means that a fulltime crowdworker will most probably make more than the minimum salary and there is a good chance that they can make 3 times the minimum.

5.4 Risk Analysis of Proposed Solution

The problem with the proposed solution is that team devel-

oping it, has never built such a system. So what are the risks to build a system like that?

In NASA[19] there is a simple technique to identify risks:
"we can distinguish risk in three possible ways:
a) known-known
we know the risk and have retired it,
b) known-unknown
we know that there is a risk and the risk is modeled and
c) unknown-unknown
we don't even know there is a risk.

Exploration is about diving in the unknown-unknown."

Under the technique the team could easily distinguish the risks:

a) known-known:
The system had to be distributed across different users, with user login and database persistence. This wasn't a problem since the team had experience developing web applications.

b) known-unknown:
The team knew they needed to process images digitally but they didn't know the tools or techniques needed to do that. This risk was mitigated by having one student from CIMAT build a prototype of the relevant image processing techniques.

c) unknown - unknown:
This is the difficult part. In order to discover these risks the team took three approaches:

1. Interviews with experts in archives.
   The team was able to explore early problems by talking with experts that have worked in big government archives before this helped identify some risks that weren't considered in the beginning.

2. Build a Trace Bullet prototype [20].
   A Tracer Bullet is inspired by the airplanes during the 2nd. World War in which of every fifth bullet shoot by an airplane would lead a trace so that the gunman could aim better. So a tracer bullet prototype is a prototype that covers "all" the layers of the system so that it can discover the communication problems between layers. The prototype for Capturista Sobre Ruedas tried to create an end-to-end system for only one user and one book. It uncovered several problems on the interaction of the different subsystems.

3. Review of the literature on Crowdsourcing and Human-Computation.
   Apply the considerations for designing a Crowdsourcing platform and considerations when designing a Human-Computation platform and try to identify things that the team didn't foresee.

5.5 Proposed Solution under the Crowdsourcing Paradigm

In the following table is the analysis of the proposed solution under the Crowdsourcing paradigm.

Concept	Was it considered before the analysis? (risk category)	Mitigation Strategy
Considerations when developing		
1. Nature of collaboration	known - known	Collaboration is going to be explicit.
2. Type of Target Problem	known-known	Members are going to verify and correct OCR output.
3. How to recruit and retain users?	unknown-unknown	The team expects to recruit to each government office of person with disabilities. Now to retain them the team is thinking of implementing a reputation strategy like badges, or points that the rest of the members of the crowd can show.
4. What contributions can users make?	known-known	Members will correct output from the OCR program and verify text.
5. How to combine contributions?	known-unknown	Corrected texts are going to be mixed with other texts until a customer has a fully digitized book.
6. How to evaluate users?	unknown-unknown	When user submits a corrected text another member is going to verify the text, this will rank members by quality of submissions and speed in verification.
7. Degree of manual effort	known-unknown	In the begging members of the crowd were going to type the whole document latter it was decided they would verify

		OCR text, therefore simplifying the text input.
8. Role of human users	known-known	According to the taxonomy they are going to be "slaves/members of the crowd"
9. Standalone vs. Piggyback architecture	known-known	It was already decided that the application was going to be standalone.
Criticisms		
CS1. Ethical concerns about paying less than minimal wage.	unknown-unknown	In order to prevent this problem members of the crowd that work at least 80 hours / month. Will be hired as permanent employees with minimum wage.
CS2. Increased risk of not finishing on time	unknown-unknown	If work starts to accumulate there will be bigger invitation to increase the size of the crowd.
CS3. Risk of low quality work	known-known	Members of the crowd are going to verify other members work in order to increase quality.
CS4. Typically no confidentiality agreements	unknown-unknown	In order to give customers confidence that their information is going to be secured only workers paid with minimum wage will work in projects that require confidentiality agreements.
CW1. Below minimal wage income	unknown-unknown	It is going to be resolved with contracted workers as CS1.
CW2. No form of work contract or work stability	unknown-unknown	It is going to be resolved with contracted workers as CS1
CW3. No communication with other members of the crowd	unknown-unknown	It was decided that crowdworkers are going to have a chat to communicate with other crowdworkers that happen to be working at the same time.

Table 3 Risk analysis under Crowdsourcing paradigm.

From this table we learn that from the Crowdsourcing Paradigm there are 16 concepts that we need to verify. Of those 7 were unknown - unknowns so the team learned 77% more of this analysis (9 was 100% of knowledge).

5.6 Proposed Solution under the Human-Computation Paradigm

The following table presents the analysis of the proposed solution under the Human-Computation paradigm.

Concept	Was it considered before? (risk category)	Mitigation Strategy
Considerations when developing a Human-Computation system		
1. What is the motivation?	known - known	Payment
2. How is Quality going to be Controlled?	known - known	Multilevel Review
3. How are contributions going to be aggregated?	known - known	Collection
4. What Human Skill is needed to perform the task?	known - known	Visual Recognition Language Understanding
5. What is the Process Order?	Known - Known	Task are going to be assigned in a First-In First-Out series
6. What is the Task-Request Cardinality?	Known - Known	one-to-many one book generates several workers.
Criticisms		
HCC1. It's difficult to write code.	Known - unknown	After building the Tracer Bullet prototype it was decided that the best approach was to use a pipe and filter-architecture with the pipes represented by

		queues such as ZeroMQ[21].
HCC2. It's difficult to debug.	unknown - unknown	The strategy is going to use memoization for the human tasks. So that it can run the debugger without asking humans again.
HCC3. It's difficult to test.	unknown - unknown	The same strategy as in HCC3 applies.
HCC4. Long lived processes.	unknown - unknown	The same strategy as in HCC1 applies. Since queues will facilitate the creation of short lived processes that will restart if they fail.

Table 4 Risk analysis under Human-Computation paradigm.

From the previous table we know that in Human-Computation there are 10 concepts to consider of those 3 where previously unknown - unknown, so the team was oblivion to them. This analysis yields 42% more knowledge.

5.7 Refined Solution

The simple known-unknowns technique of risk analysis did uncover several things that the team hadn't considered. The most important realizations are:

In order to avoid ethical concerns the team will hire members of the crowd that show that work on average 80 hrs / month on the system. This will help in paying fair wages, provide work stability for the workers and confidentiality agreements for the customers.

The members of the crowd apart from money need other motivators to spend time on the site. That's why the team is adding social features such as a chat so that crowdworkers can communicate with other members and badges and points so that there is some form of prestige that motivates workers.

In order to facilitate debugging and automatic testing, tasks delegated to humans will use the memoization technique[13]. So that a human functions are done once and only once.

A pipe and filter architecture based on queues such as ZeroMQ that will provide stability between processes that will run for long times.
As a result of this analysis it was possible to answer the following research questions:
1. Can both paradigms be applied to the same problem?
Yes, although Crowdsourcing and Human-Computation deal with different kinds of systems there is overlap and as we've seen in fact the most interesting problems that can be solved with Human-Computation will require a Crowdsourcing component to provide large amounts of human processing power. The inverse isn't necessarily true are there interesting Crowdsourcing solutions that not use Human-Computation? yes, there are many categories such as crowdfounding, or wisdom of the crowds that not require Human-Computation.

2. And does applying both paradigms to the same problem actually provide any benefits?
Yes; as seen on the analysis the team learned 77% unknown-unknowns from the Crowdsourcing paradigm and 46% from the Human-Computation. 123% more knowledge about the problem. Thanks to using both paradigms the team has more than doubled their knowledge of the problem which leads to savings in time, money and frustration.

Crowdsourcing will provide the business view of designing the system. Will focus on the human side of the system: recruiting people of the crowd, focus on their incentives, etc. Human-Computation will focus on the algorithmic side of the problem will help define the architecture and how the system will integrate the work of the humans.

6 Conclusions and Future Work

In the Crowdsourcing and Human-Computation paradigms it was identified what concerns and criticism should be taken into consideration when developing such a system. Then those concerns were used to analyze the Capturista Sobre Ruedas system which more than double the knowledge about the system.

In the future the team intends to build the system. Taking in to account the a few considerations raised from this paper:

1. Capturista Sobre Ruedas should consider the possibility of integrating with other companies. In away that they can "lease/rent" their work force i.e. the crowd to other companies that are also doing digitalization projects. There are considerations such as the API for integration between companies and how would the team deliver the results. Still is an important business opportunity.
2. Another important aspect in case of the "leasing/ rent" option is to give the customer the opportunity to define their own quality control mechanisms and have the system be smart enough to adapt to this demands.

Contributions of this paper are:

a) Use of dimensions of taxonomy as considerations when developing a system. Since there are very few guidelines when developing a Crowdsourcing or Human-Computation system, in this paper we used the taxonomy criteria as concerns that a team must consider.
b) Tools for discovering risks in the unknown-unknown category. It was proposed the use of two tools to uncover unknown-unknown: 1. Build a tracer bullet prototype and 2. Review the literature

to identify concerns and criticisms of similar problems.

Finally as a proof of concept the author went ahead and developed a smaller system (ClickFactura) as a proof of concept of the challenges that a system such as Capturista Sobre Ruedas would require. The report on that experience is included in section 7 Appendix.

"The question that motivates my research is, if we can put a man on the Moon with 100,000 [people], what can we do with 100 million?" (Vohn Ann) [22]
"of those 100 million, 10 will have a disability, we must include them." (Alejandro García F.)

7 Appendix: ClickFactura a proof of concept for Capturista Sobre Ruedas

7.1 Introduction

ClickFactura is an application that was developed as a proof of concept of how the Capturista Sobre Ruedas system could be developed. ClickFactura is a mobile application in which a customer goes to a retail store such as Wal-Mart1 and takes a picture of his receipt and from that picture an invoice is generated and sent automatically to its email. Currently in order to generate an invoice for tax purposes in México a customer must have its ticket and then login to http://walmart.com.mx click in "Facturación Electrónica" and from there type several data that are in his purchase ticket. ClickFactura would save all this typing and replace it by just taking a photo of the receipt.

That is from the point of view of the customer, however in the implementation side what is really going to happen is that the photo of the ticket will be sent to a human (Human Computation) that human will type the data from the ticket that is needed from the website. Those data are going to be sent to a web robot2. The web robot will type the data into the stores. The stores will generate the invoice and from there it will send it to the customer.

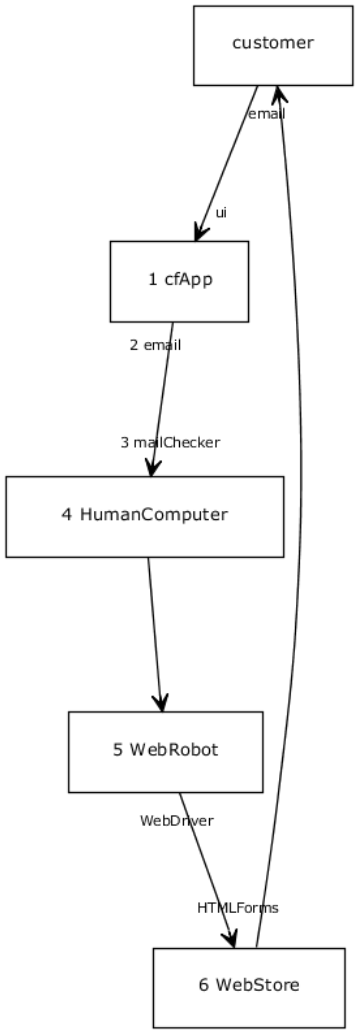
ClickFactura covers some of the same basic operations that happen in Capturista Sobre Ruedas as can be seen in the following table.

Concept	Capturista Sobre Ruedas	clickFactura
Scanner	A full size document scanner would be used.	A smartphone camera is used.
Computer Pre-Processing	An OCR will be used to pre-process images.	--
Human Computation	A human will verify and correct the output from OCR.	A human types the data contained in the ticket.
Computer Post-Processing	A program will integrate all OCR images into a book.	Web robot will send typed data to Stores websites.
Verification	Another human will verify the pages OCR.	The stores website will validate if the typed data was correct or not.

Table 5 Similarities between clickFactura and Capturista Sobre Ruedas.

7.2 Components of the system and development

Figure 4 Components of clickFactura



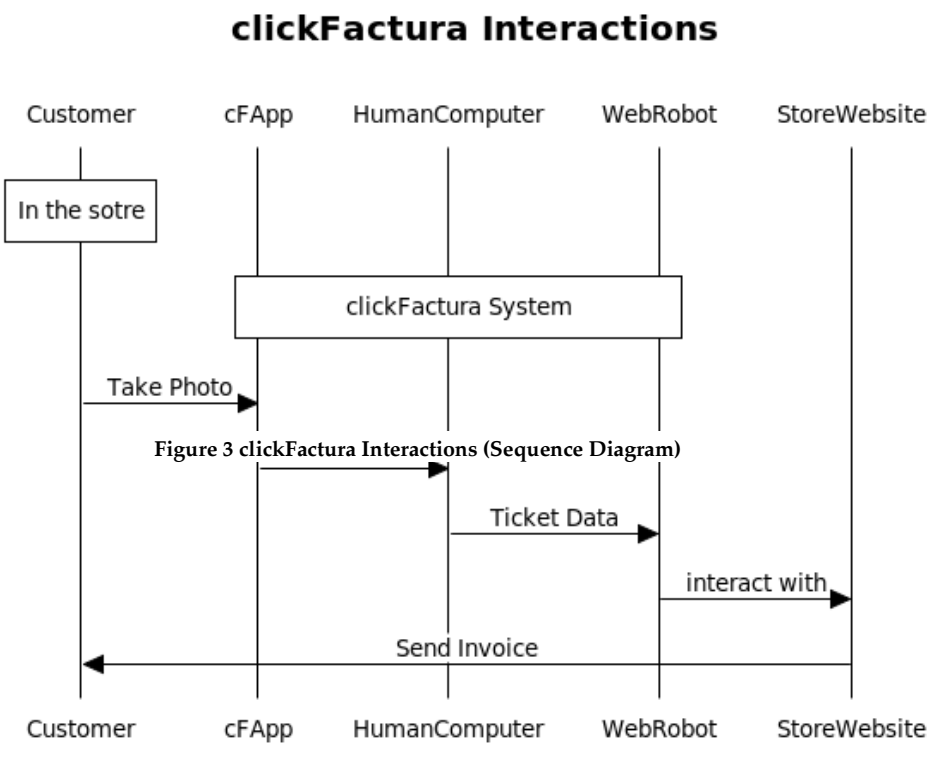


Figure 3 clickFactura Interactions (Sequence Diagram)

[1] Walmart (
[2] Web Robo
man. (Wikipe

The clickFactura system is easier to develop since on it the team focused in getting the system done without concerns out optimizations. The attitude when the system was developed was "just get it done" don't worry about its speed or performance or even quality. This won't be a production system just a proof of concept. With this in mind the components developed are presented in Figure 4 Components of clickFactura.

7.2.1 ClickFactura App (cfApp)

Is the mobile component of the system is where the user takes the photo and it 's sent to the human computer. It was developed by using a tool from the Massachusetts Institute of Technology (MIT) called AppInventor.

AppInventor is a Rapid Application Development (RAD) tool for mobile devices running on the Android Operating system. It has three main components:

1. Interface Designer Figure 6.
2. Block Editor Figure 5.
3. Smartphone for debugging

It uses a visual programming language that allows non experts in programming to develop applications. The Advantages of using AppInventor where clear using the Drag-n-Drop interface for programming allowed a complete beginner develop the clickFactura with no problem.

However, there were some problems with development:

- Currently it isn't possible to send emails with an attachment (the ticket photo) from AppInventor without an external component.
- The external component used is called MailAttach and from AppInventor you can't send two attachments. Originally it was desired to send to photos one of the top of the ticket and one of the bottom.
- And it was also discovered that you can't set the resolution on the camera from AppInventor.

However all those where minor details, and it was possible to work around them in pretty fast. The application was developed in 30 hours by a complete beginner in mobile development.

7.2.2 MailAttach

Once the photo of the ticket was taken it was needed to send it to the Human Computer. So we needed a form of communication that was:

- Asynchronous.
  Because maybe the user didn't have internet at the mall. Fast
- Fast.
  Could deliver a "big" file (the image) and
- Small
  It had to fit in the smartphone
- Reliable
  Wouldn't lose the photos.

And the answer was, of course: Email. The email client that is already available on the smartphone covered all this prerequisites.

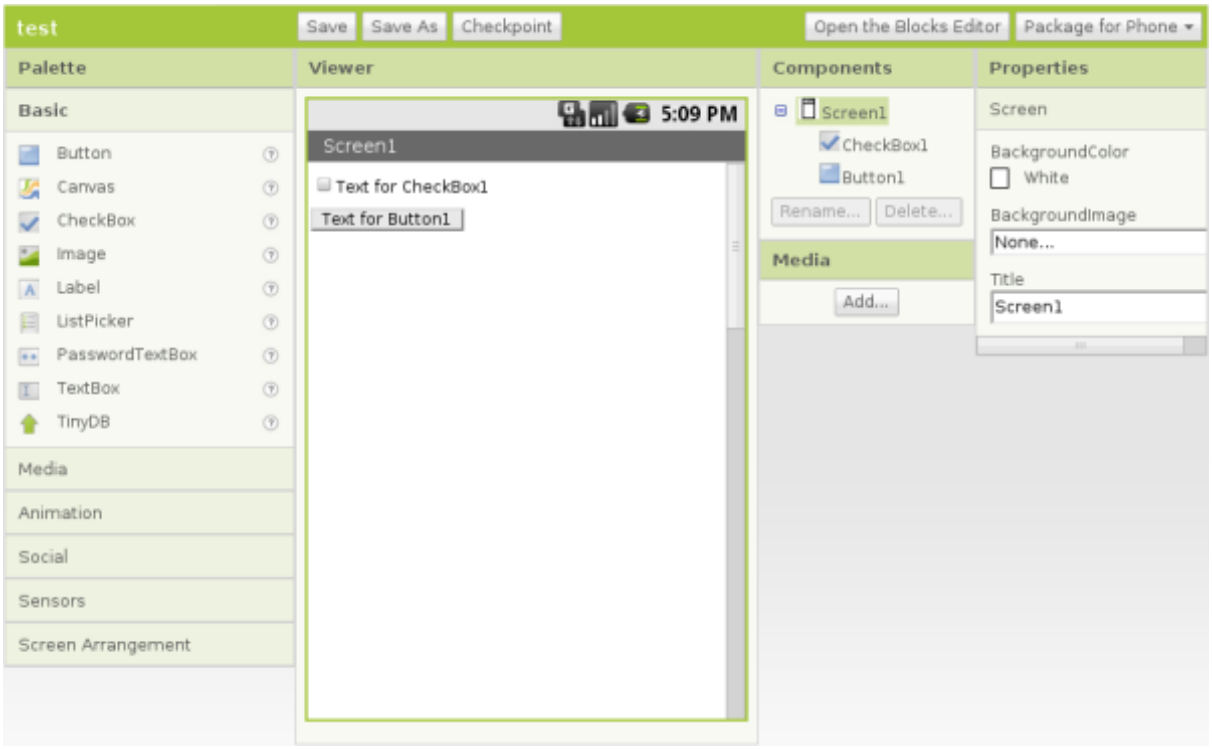


Figure 6 AppInventor Interface Designer [23]

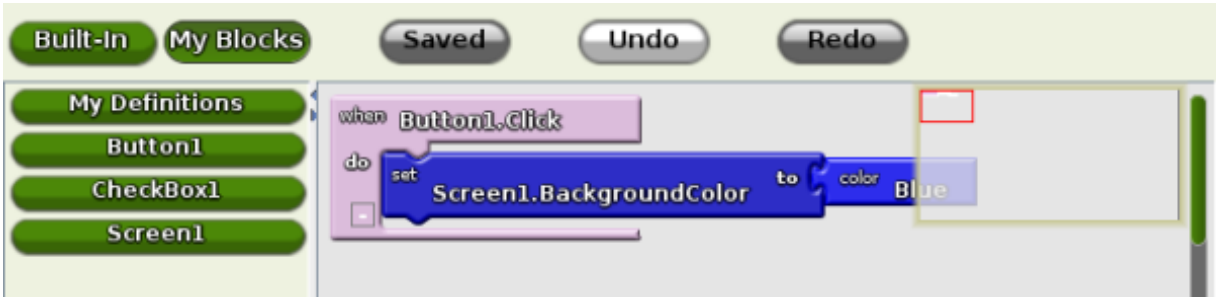


Figure 5 Block Editor (Visual Programming language) [23]

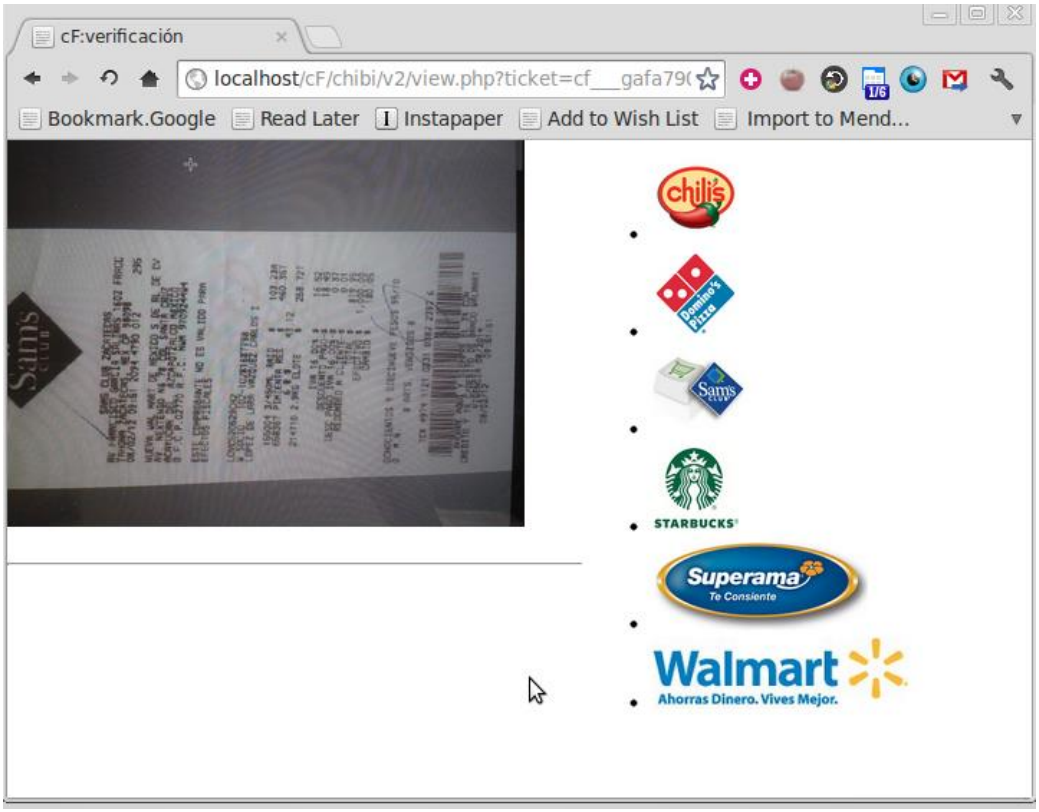


Figure 8 Step 1 identify to which store the ticket belongs to.

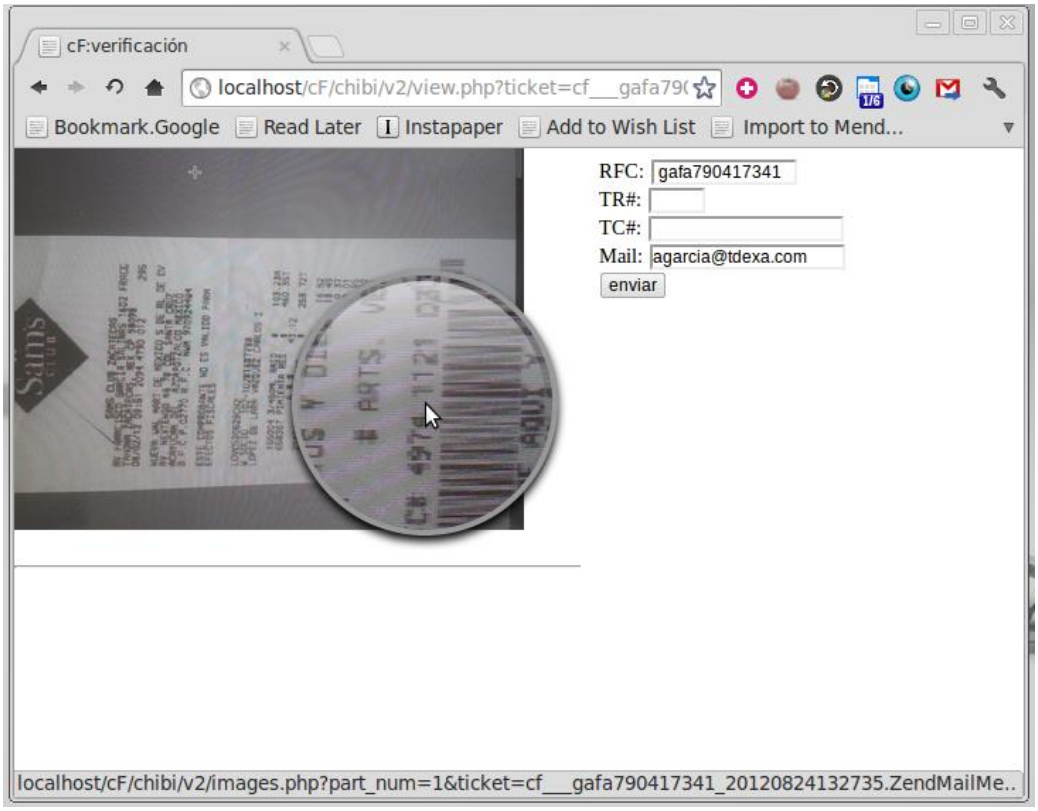


Figure 7 Step 2 Type expected data. (Notice the magnifying glass)

7.2.3 MaiChecker

Once the mobile app sent the email, a script called MailChecker.php connects to the email account every 60 seconds and downloads new emails using the IMAP email standard. Then it downloads the email, the email comes with the image already. Then it is saved. In the beginning it was considered that MailChecker would open the mail and save each field on to a relational database and the images as files. However, it was later considered to just save the whole email text as a file (even with the images encoded on it) . So it was basically treating each email as "record" every part as a "field" and the whole set of email files as a "document database". This once again speeds up development. So maybe for a more professional version of clickFactura or even for Capturista Sobre Ruedas one must consider a real document database.

7.2.4 Human Computer

Now a set of humans would get the ticket photo and will start to work on it.

- The first step was to identify to which store the ticket belonged to. (Figure 8)
- The second step is to type the data from the ticket in predefined fields. (Figure 7)
- And finally send it to the Web Robot.

This script was developed with PHP and JavaScript, languages which are pretty common when developing web applications.

However the most interesting findings came from the UI. First in some of the tickets the image was too small so we added a magnifying glass tool. Second some images where in horizontal orientation so it was needed to turn them. Then also on the typing fields there wasn't any validation and even though the fields where very small and with very little data humans made errors that could be easily prevented, such as not allowing letters in a number only field, or not allowing 10 characters in a field that required 12, etc. So the conclusion is in Human Computation the interface for humans is crucial in making the system less error prone.

7.2.5 Web Robot

The Web Robot was an application actually extracted from the Selenium and phpUnit projects. Normally Selenium is used to test web applications by recording human interactions with a website and then reproducing them as part of a test suite that runs automatically. In this case the team extracted the part that made all the typing on the website and used it as a means to interact with stores websites.
The challenges on this component were that currently Selenium is using a new version called Web Driver and some of the functionality isn't present on the phpUnit plugin.
But most importantly is that Web Stores have some complex components that are actually quite difficult to control from the robot. So it became a form of trial an error on this application.

7.2.6 Web Stores

Here it is a normal website that just asks some data to generate the invoice for customers. And then sends a pdf document to the customer with his electronic invoice.
The challenges as where mentioned before are that the sites use complex user interactions that are difficult to control from the Web Robot, after all that wasn't their first intention and also the fact that if you want to generate the same invoice twice some stores will only send an email while others will only let you download a file. So clickFactura will need to make that a homogeneous operation.
However there were some nice surprises, turns out that the same system is used by several different stores like SAM's and Wal-Mart use the same system. So this has the advantage that as soon as one Web Robot is made for one store it can actually work for several different others.

7.3 Conclusions and Future Work

The clickFactura application was developed as a prototype of the Capturista Sobre Ruedas system. To do it in fast time RAD and hobbyist tools where used. (AppInventor) also proven communication mechanisms (mail and Selenium) and it was assumed that Human Computers would the heavy weight work. (I.e. no OCR system was used).

The emphasis on this project was on learning the challenges associated with developing a crowdsourcing system and on speed, leaving aside considerations such as speed and ease of use.

Developing the clickFactura prototype will help in developing the Capturista Sobre Ruedas system, because in essence clickFactura is just a small subset of all that Capturista Sobre Ruedas will need to do. Developing the system using a RAD tool such as AppInventor and proven components such as Email and Selenium was essential in developing the prototype at a fast speed. The whole prototype took 2 weeks end to end by only one developer.
And the most important thing is that this proof of concept proved a nice overview of what developing Capturista Sobre Ruedas will be like.

As future work we have:
Develop the Capturista Sobre Ruedas system.
Also for Capturista Sobre Ruedas a Queue mechanism that works as good as email for images, and tasks.
Develop a mechanism that allows a fair distribution of work among crowdworkers, so that each crowdworker has a small pool of work to do

8 REFERENCES

[1] E. Estellés-Arolas and F. González-Ladrón-de-Guevara, "Towards an integrated crowdsourcing definition," *Journal of Information Science*, vol. 38, no. 2, pp. 189–200, Apr. 2012.

[2] L. von Ahn and L. Dabbish, "Designing games with a purpose," *Communications of the ACM*, vol. 51, no. 8, p. 57, Aug. 2008.

[3] J. Howe, "The rise of crowdsourcing," *Wired magazine*, no. 14, pp. 1–5, 2006.

[4] "Crowdsourcing Industry Taxonomy by Crowdsourcing.org (V2)." [Online]. Available: http://www.crowdsourcing.org/editorial/crowdsourcing-industry-taxonomy-by-crowdsourcingorg-v2/2852. [Accessed: 31-Jul-2012].

[5] A. Doan, R. Ramakrishnan, and A. Y. Halevy, "Crowdsourcing systems on the World-Wide Web," *Communications of the ACM*, vol. 54, no. 4, p. 86, Apr. 2011.

[6] G. Norcie, "Ethical and Practical Considerations For Compensation of Crowdsourced Research Participants Introduction," 2011.

[7] P. Ipeirotis, "Analyzing the amazon mechanical turk marketplace," *XRDS: Crossroads, The ACM Magazine for Students*, no. 2, pp. 16–21, 2010.

[8] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on Amazon Mechanical Turk," *Proceedings of the ACM SIGKDD Workshop on Human Computation - HCOMP '10*, p. 64, 2010.

[9] B. B. Bederson and A. J. Quinn, "Web workers unite! addressing challenges of online laborers," *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems - CHI EA '11*, p. 97, 2011.

[10] J. Ellenberg, "This psychologist might outsmart the math brains competing for the Netflix prize," *Wired Magazine, March*, pp. 1–5, 2008.

[11] K. R. Lakhani, P. A. Lohse, J. A. Panetta, and L. B. Jeppesen, "The Value of Openess in Scientific Problem Solving," *Biotech Business*, vol. 18S, no. Spec No 2, pp. 5533–464, 2007.

[12] L. von Ahn, "Human Computation," in *2008 IEEE 24th International Conference on Data Engineering*, 2008, vol. 0085982, pp. 1–2.

[13] G. Little, "Programming with human computation," MIT, 2011.

[14] A. J. Quinn and B. B. Bederson, "Human computation: a survey and taxonomy of a growing field," *… conference on Human factors in computing …*, 2011.

[15] A. Kosorukoff, "Human based genetic algorithm," in *2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace (Cat.No.01CH37236)*, vol. 5, pp. 3464–3469.

[16] CÁMARA DE DIPUTADOS DEL H. CONGRESO DE LA UNIÓN, "Ley Federal Del Trabajo," *Diario Oficial de la federación*, pp. 1–228, 2012.

[17] Editorial Freelancers Association, "Editorial Rates," 2012. [Online]. Available: http://www.the-efa.org/res/rates.php. [Accessed: 01-Aug-2012].

[18] S. del T. y P. S. (STPS), *Salario Mínimo General por Área Geográfica.* 2012, p. 2012.

[19] D. A. Maluf, Y. O. Gawdiak, and D. G. Bell, "On Space Exploration And Human Error - A Paper on Reliability and Safety," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 2003, pp. 79–79.

[20] F. P. Brooks, *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition (2nd Edition)*. Addison-Wesley Professional, 1995, p. 336.

[21] "ZeroMQ." [Online]. Available: http://en.wikipedia.org/wiki/%C3%98MQ. [Accessed: 31-Jul-2012].

[22] "Luis von Ahn | Profile on TED.com." [Online]. Available: http://www.ted.com/speakers/luis_von_ahn.html. [Accessed: 31-Jul-2012].

[23] "App Inventor for Android." [Online]. Available: http://en.wikipedia.org/wiki/App_Inventor_for_Android.