# Experts On Demand: Enabling Flash Organizations with Rapid Onboarding

A Thesis Presented for the Master of Science

Symbolic Systems Program, Stanford University

Alexandra To

June 2015

To the Directors of the Program on Symbolic Systems:

I certify that I have read the thesis of Alexandra To in its final form for submission and have found it to be satisfactory for the degree of Master of Science.

Advisor: Michael Bernstein
Computer Science

To the Directors of the Program on Symbolic Systems:

I certify that I have read the thesis of Alexandra To in its final form for submission and have found it to be satisfactory for the degree of Master of Science.

Second Reader: Melissa Valentine
Management Science & Engineering

# Acknowledgements

I would like to thank Michael Bernstein and Daniela Retelny for their constant encouragement and wonderful mentorship. Two years ago I never would have envisioned myself going to grad school. As I complete my Masters degree and look forward to starting my Ph.D. in HCI in the Fall I know I never could have done this without your support and the love of research you have both instilled in me.

Thanks to my amazing second reader, Melissa Valentine, who has introduced me to the world of organizational behavior research. Thank you to Negar Rahmati, Tulsee Doshi, and Jare Fagbemi for their efforts in improving Foundry and enabling flash organizations. I couldn't ask for better research collaborators!

As always, thanks to my incredibly supportive parents, John and Joanne, and sisters, Robyn, Caroline, and Rebecca. Additional thanks to my grandparents, Reiko and Michio Yoshizu and Stanley and Rosemary To for their support.

Thanks to my friends, Roger and Xiao Song, and to my partner Joe.

This research was conducted with support from a UAR Minor Grant and from a Computational Social Science Research Grant.

# Abstract

On-demand, online workers must develop strategies to rapidly onboard themselves onto a project to work cost effectively. For expert crowd workers, this is particularly challenging because they are completing complex and sometimes interdependent work. They may want to know the goal of the larger project, what they are asked to deliver when they are done working, their deadline, and how much they will be paid. However, they are hired on demand just at the moment at need, and arrive and are expected to begin working without any of this information. Using question-under-discussion models and information foraging theory, I study how workers rapidly onboard using primarily their first message about a job. This is achieved by manipulating how interactive a worker's initial information about a potential job is. We can model how people seek out important information through an information hierarchy of goals and sub-goals. I conduct a study in which front end developers from the expert crowd are offered job details in either interactive or non-interactive information hierarchy form. The results show that workers produce higher quality deliverables and follow instructions more closely in the interactive hierarchy condition, but that the interactive information hierarchy may also cause workers to leave before they can finish consuming the required information.

# Contents

# Chapter 1

# Introduction

Crowdsourcing systems provide requesters with access to large groups of people who can solve problems at scale. Microtasking systems, such as Amazon Mechanical Turk, allow requesters to manage and pay non-expert crowd workers. These workers are effective at completing simple, independent tasks such as image labeling. Freelance websites, such as Upwork (formerly oDesk) and Elance [2, 1] provide access to expert crowd workers, who can complete more complex tasks such as software development and design [7]. However, expert crowd work is largely independent and uncoordinated and thus highly interdependent complex projects remain out of reac.

Flash teams demonstrate that on-demand teams of paid experts from the crowd can succeed in completing complex and interdisciplinary projects [12]. Flash teams introduce a framework for dynamically assembling small teams of expert crowd workers using linked modular tasks that can be managed computationally. They also introduced Foundry, an online platform for authoring and managing flash teams.

With the success of flash teams, we now turn to the challenges of scale. Flash teams can create complex and interdependent projects such as a short animvated video or a web application prototype. However large scale projects such as a video game with

these techniques remain out of reach, despite our access to the expert crowd. As a project scales, the overhead for training new workers grows as they onboard into a context-heavy project and hierarchy becomes necessary to solve decision making and management. If a team of experts can complete a complex and interdependent project within days, can an organization be similarly crowdsourced using experts? Can we create *flash organizations*? We envision that using flash organizations, requesters can crowdsource the creation of large projects from brainstorming to prototyping to marketing and production. The expert crowd could create, for example, a fully functional iPhone app, video game, or a music album. Requesters with no management experience can access all of the expertise they need to enable their vision without the slow startup cost of creating a company.

Flash organizations face two major challenges that were not addressed in flash teams: 1) how do you efficiently onboard a worker into a complex project that is far downstream, and 2) what does an organizational hierarchy look like with all on-demand, temporary workers? Hierarchy becomes necessary at the organizational level in order to enable high and low level decision making. Flash organizations accomplishes this through an adaptive organizational structure. *Rapid onboarding* involves bringing a worker up to speed on the overall purpose of a project, its current state of progress, and the details of their task and role within the project. Onboarding is crucial to a worker's mental frame of a project and their role and is necessary to enable the speed of a flash organization from both the perspectives of the worker and the client. Flash organizations have a much larger amount of organizational history and context that a worker may need to understand before working than a flash team. Rapid onboarding aims to take advantage of the availability of the on-demand, online crowd while minimizing the cost of training and joining a large, complex project. In this thesis, I study one of the tools that we have built to enable rapid onboarding to a flash organization.

To better understand how we can enable rapid onboarding, this research proposes that workers may get the most information from their very first message detailing a project and task. This initial message is incredibly important for onboarding because it frames a workers understanding and they may interpret all future interactions from that frame. We study how either interactive or non-interactive information structures effect how well a worker follow the instructions of the task as well as the overall quality of the work a worker submits.

Human-computer interaction and linguistic theory offer compatible models for how people seek information. In linguistics, question-under-discussion (QUD) theory models how participants seek information through discourse and question asking. In this two-person game where the goal is to exchange information, agents traverse an information hierarchy of goals and subgoals through their conversation [3]. Information foraging theory reframes this flow of information in a one-way direction as a user of a digital tool attempts to seek information through a tool's information hierarchy [10]. When a user browses a website to seek information, they cannot ask questions as they might in a normal discourse. This puts the pressure on the designer of a digital tool to not only deliver the information a user may be seeking, but to design it with good "information scent" - signallers where users can find the information they are looking for.

An online worker's intuitive motivation for rapid information seeking is that any time wasted learning about a project that the worker might not take represents lost income. We therefore create two different information hierarchies for online expert workers to navigate as they seek the details of an available project and task - interactive and non-interactive. While information foraging theory tells us an interactive information hierarchy is optimal for information seeking in a digital environment because it allows the user to consume information in small contextual chunks [10], a non-interactive information hierarchy provides a worker with all of the information they need in one

long message and thus affords them at a glance to calculate how long it may take to onboard in the first place.

We conducted a study to examine whether workers perform better at a task when introduced to a project through either an interactive information hierarchy or a non-interactive information hierarchy. After recruiting front-end developers from freelance websites to our panel of on-demand workers, we invite them to take a 3-hour front-end web development task. If they express interest, they are sent the project details through one of the conditions and hired to work on the task.

Our results suggest that navigating through an interactive information hierarchy takes significantly more time than it does to read a non-interactive information hierarchy, but that it produces better work. We performed a design review of the code each worker delivered and used a list of features specifically requested to check whether the code fulfilled each item on the list. The interactive information hierarchy slightly increases the quality of work and significantly increases quantitatively how well a worker follows the instructions of a task. Unfortunately, the interactive hierarchy does also seem to intimidate workers, as several expressed interest in the task, but then declined to accept the task after receiving the interactive hierarchy condition.

Chapter 2 discusses flash teams, the groundwork for flash organizations. It also discusses both QUD models and information foraging theory. Chapter 3 introduces our current work on flash organizations and the Foundry system, which was introduced in the flash teams work [12]. As part of our work towards flash organizations, we have created new tools in Foundry to enable rapid onboarding. In order to illustrate how these tools are used, I describe a scenario through which a worker is invited to our panel, takes, and works on a job for flash organizations. I evaluate how the use of different information hierarchies in the very first time a worker receives information about a job influences the quality of their work (Ch.

4) and discuss the results of this study (Ch. 5). I conclude with the future work surrounding flash organizations (Ch. 6).

On-demand, online workers must develop strategies to quickly onboard themselves onto a project to work cost effectively. For expert crowd workers, this is particularly challenging because they are completing complex and sometimes interdependent work. In order to enable our on-demand flash organizations, we must develop strategies for rapid onboarding. With rapid onboarding, workers can come in and out of the project as needed with little overhead for either them-self or the client.

# Chapter 2

# Related Work

While some crowdsourcing systems such as Voyant have enabled expert work in non-expert workers through highly structured tasks [16], many have turned to freelance websites such as Upwork or Elance [2, 1] to complete complex tasks requiring specialized workers [7]. We refer to this as the expert crowd [7]. To enable complex tasks crowdsourcing systems such as Turkomatic use workers from the crowd to break down complex tasks into smaller context-free tasks [8].

With flash teams [12], we demonstrated that the expert crowd can complete complex, interdependent tasks by taking advantage of lightweight, modular team structures. Flash teams use on the order of 3-8 workers, but as we look forward, we want to enable *flash organizations*, which may consist of 10s or 100s of expert crowd workers.

One of the key components to enabling flash organizations is utilizing crowd workers on-demand. In order to solve the problem of limited attention, previous work has used clusters of information to provide workers with a global view of large sets of data [4]. Flash organizations, which are context-heavy projects at a much larger scale than flash teams, must similarly provide information about the organization to the worker.

While flash teams complete complex and interdependent projects, they are typically on a small scale and workers often have overlapping expertise (e.g. a web app prototype project may need a web developer and web designer) [12]. A flash organization may be much larger and require work from workers with non-overlapping domains (e.g., a project to create a video game may require a software developer and a creative writer). In order to allow on-demand workers to begin working in an organization that is laden with context, we must develop tools for rapid onboarding.

## 2.1 Question-Under-Discussion Models

Linguists have proposed that discourse can be framed as a game focused around the *question under discussion* or QUD [13]. When agents in a discourse have some information goal that they are trying to achieve, their discourse often follows a tree structure where goals are achieved through branching sub-goals [3]. That tree structure can be viewed as a type of information hierarchy, where the agents travel down the tree asking more and more specific questions in the discourse, and backing out to a higher level when they reach a dead end or have not yet achieved their information goal.

In discourse, an information structure may consist of the question under discussion or focus of the discourse, discourse moves or questions and answers you can utter appropriate or potentially inappropriate to the discourse, and presuppositions [13].

For our purpose of rapid onboarding to a project, the question under discussion is the project description and task requirements that a worker must complete. Discourse moves become extremely limited as we deliver all of the information at once to a worker. While workers may reply with clarifying questions, they may be considered inappropriate discourse moves depending on the volume and type of question. This leads us in to the presuppositions. In a discourse, pragmatic presuppositions are

the shared knowledge between two agents - the speaker's assumptions about the hearer's state of knowledge [9]. For example, if you are speaking to a student in an advanced Philosphy seminar, you might presuppose that they have knowledge of basic philosophical theories.

In the digital world, where information seekers look for the information they need on a screen, we need to adapt this information structure to represent a one-way flow of information. Speakers are limited in their options for normal discourse such as question asking, gesture, etc. and must make assumptions about the user's shared knowledge based on who they expect to use their tool. Speakers or in this case, designers of digital tools gain new tools online to emphasize their meaning such as boldface type and physical hierarchy representations on the screen.

## 2.2 Information Foraging

Human-computer interaction academics have theorized how information is transferred and consumed for the digital age [11, 10]. While QUD models discourse between individuals, today information is often conveyed digitally. Designers of digital tools must convey information one-way through a screen to a user.

Some tools provide users with real-world navigation signalling to navigate through information [15]. Others use tree-structures and clustering to navigate through large sets of data [10]. However, these foraging strategies may be easily disrupted if a user experiences delays while navigating and may terminate their search early [5].

Information foraging theory is an approach that helps us understand strategies for information seeking and absorption [11]. The theory shows how people modify their strategies to maximize their rate of gaining valuable information. Some dimensions that a person may try to optimize include time costs, resource costs, and opportunity

costs, all of which are highly relevant to a freelance worker.

Some freelance workers have other full- or part-time jobs in addition to their freelance work, while others make their living on online freelance markets. For some, it may be most effective to choose a long-term contract that may last weeks or months. For workers interested in short-term jobs, optimal information foraging strategies are especially important. Novice workers attempting to learn more about their work can effectively use information foraging and information scent to learn quickly about their task if combined with the use of strategic calculations about the costs and benefits [6]. Workers must calculate the cost of time spent learning about and onboarding to a project weighed against the work required and payment made for the project.

This puts the onus on clients who use digital means to convey information about a project to design for optimal information foraging. In presenting information foraging theory, Pirolli and Card demonstrated that a dynamic information hierarchy can allow people to intuitively navigate through large amounts of information [10]. Specifically, an interactive information system using clusters of information can aid users of digital tools seeking specific information.

# Chapter 3

# Flash Organizations

Online expert crowds such as Upwork provide clients with little domain or management experience the ability to hire workers to complete complex projects. The global nature of the crowd allows a client to access to workers 24/7 and expert crowd provides enough specialization and expertise to complete highly complex projects.

We are currently working to enable flash organizations. Flash organizations are computationally-driven organizational structures comprised of online, on-demand experts from the crowd that dynamically adapt their structure. Flash organizations aim to complete highly complex, large-scale projects that require many types of expertise by taking advantage of the diversity of the online expert crowd as well as computational task structures as presented in flash teams. The two major mechanisms that enable flash organizations are dynamically assembled experts and a fluid and reconfigurable organizational structure. This thesis focuses on how expert workers can be dynamically assembled through rapid onboarding mechanisms.

## 3.1 Rapid Onboarding for Flash Organizations

The ability to hire experts on-demand to an organization is critical to the success of flash organizations. However, onboarding becomes increasingly difficult the further a project has progressed and the more the worker must rely on the work done by previous workers. In a typical brick-and-mortar organization, a worker may have days or even weeks to onboard into the organization and to their project, but this is neither time nor cost effective for online, on-demand work.

For example, in our study we ask a worker to contribute a new feature, the search results page, to the existing front end code of the project. We estimate this task to take 3 hours for a skilled front end developer. If we asked the worker to do a weeklong code camp to get up to speed on our code set-up and style, it would be hugely inefficient to only hire the worker to do actual work for 3 hours.

In this case, it may seem more efficient to hire workers for more long-term work, but this does not take as much advantage of the diversity and non-stop access to the online crowd.

## 3.2 Foundry

The Foundry system is an online platform for authoring and managing flash teams [12] and flash organizations. On Foundry, requesters can author team structures by adding role-specific members to the team and drawing tasks on the timeline. When the team is live, they can manage the team through timers on each task that display the status of a task as not started, in progress, delayed, or complete (Figure 3.1). Workers are given unique links to log-in to a specific project on Foundry. There they can view the progress of the team, track the progress of their task, upload deliverables to a

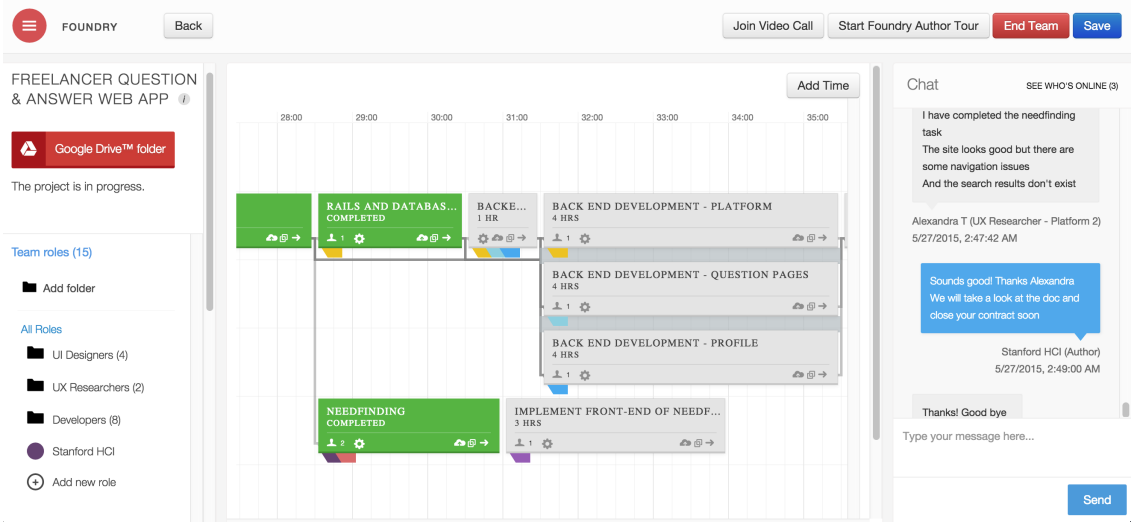


**Figure 3.1:** A client view of Foundry. The client can add and delete tasks and roles, monitor the progress of the team, and communicate with the workers via the chat.

shared Google Drive, and communicate with the client and their fellow team members.

We want to use Foundry to computationally enable flash organizations. Specifically we seek to have Foundry aid in on-demand hiring while ensuring that those workers hired understand the goal of the project and their role as quickly as possible. We have created several new tools designed to aid workers in rapidly onboarding to flash organizations. Workers are now vetted and invited to a panel of workers based on their skill set and given some training on Foundry and on-demand hiring. When they are hired for a project, workers are given a unique link to a project on Foundry that is tied to their role in that project. Through Foundry, they can look at all of the tasks that have previously been completed on the timeline, examine the details of their own task, and track their own progress on their task as they work.

### 3.2.1 Scenario

The following scenario provides an example of how a worker experiences being recruited for, onboarding to, and working on a flash organization. Each feature of Foundry that assists the worker into onboarding into the project or even Foundry and

the on-demand hiring system itself is underlined below. Despite all of the affordances Foundry gives a worker to understand the project he/she will work on, we expect that the first information a worker reads about a project most heavily influences their framing of their task.

Imagine a web developer, John, who works on a freelancer website such as Upwork. John has a full-time, 9 to 5 job, but likes to work on side projects when he has time. He receives an invitation on Upwork to join a panel of on-demand workers. Through this panel, John is told he will receive job invites via email, and can claim them and work immediately without going through the interview process for each job. He agrees, proposes his hourly rate to the client, and is directed to a short online registration where he watches some training videos about Foundry, the website the client uses to manage projects, and registers for the panel of front end developers.

Several days later, John receives a "job available" email for a job for a front-end developer (Figure 4.1). The email lists a project description, front-end development task description, expected duration of the task, and the materials he will have to work with and that he is asked to deliver. On the email is a link that will take him to a hiring queue if he wants to work on the task. He accepts and sees that he is in the number one spot on the queue. John is then given 10 minutes to review these details again, and click 'Accept' on the hiring queue website if he wants the job. He does so. In his email, John now receives a link to Foundry. He opens the site and immediately sees a small wizard pop-up that directs him to read his task, the task immediately preceding his task, and the task immediately following his task on the timeline (Figure 3.2). The pop-up tells John to review the details of his task and to review the details of the immediately preceding tasks, as the workers may have left notes about their work that could be important to him.

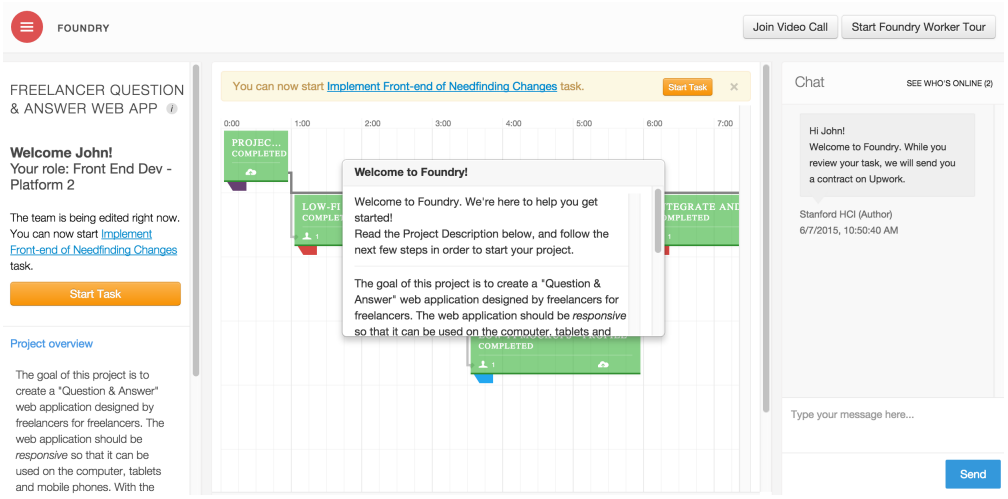


**Figure 3.2:** John's first view when he opens Foundry.

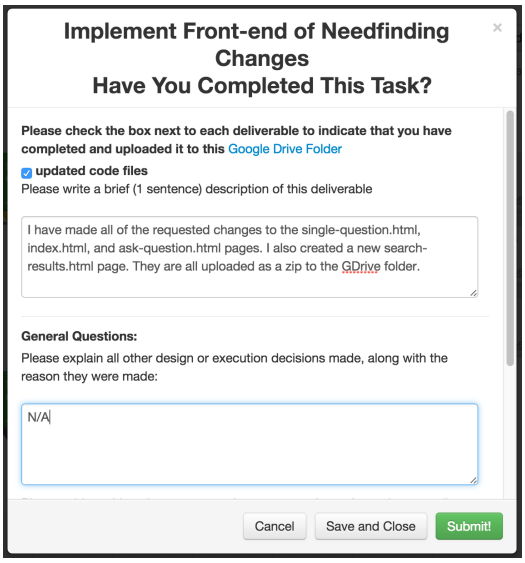


**Figure 3.3:** John fills out the documentation questions describing his code. Future workers who may need his code to continue working on the project can read his answers and see if he made any significant changes to the project here.

When John clicks on the linked preceding task to review it, he sees that indeed the previous workers have filled out a set of documentation questions about how they worked and describing the deliverables they have uploaded to the project's shared Google Drive (Figure 3.3). John can access a shared Google Drive folder for each task on the timeline, and opens the folder for the preceding task he is reviewing so he can see the work that the previous workers left form him.

Meanwhile in the Foundry chat feature, the client greets John and informs him that he will be sent a hiring contract on his freelance website of choice. John sees a button marked "Start Foundry Worker Tour" in the upper right and clicks on it. A wizard pops up

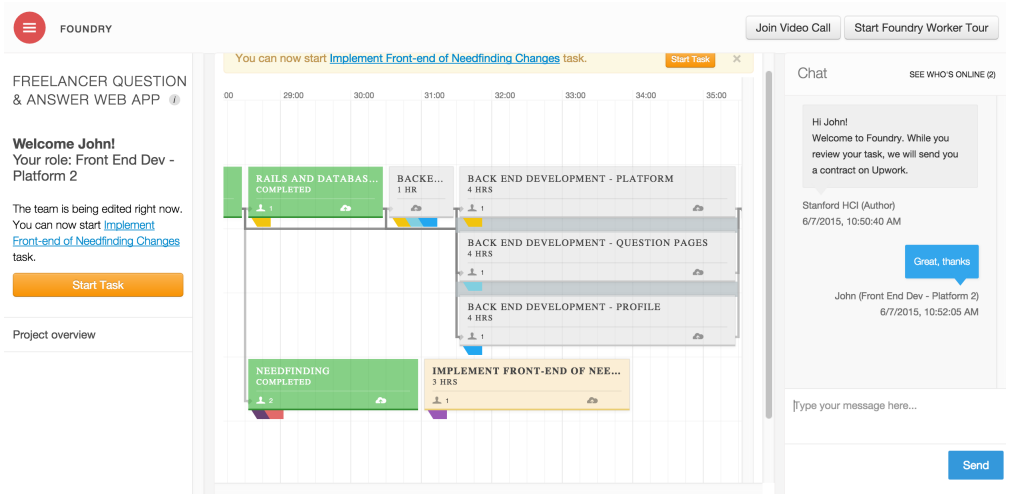


**Figure 3.4:** John's view of the team before he has started his front end development task. He can review the completed tasks in green and see his own task highlighted in yellow.

and points out to him the various features of the website and instructs him on how he can use them.

He sees his task highlighted in yellow on the timeline (Figure 3.4). He reviews his task and accepts the hourly contract and begins his work by clicking 'Start' on his task on Foundry (which starts a timer on the task) and by tracking his time through a time log on the freelance website.

When John starts his task on Foundry, it turns blue. If it becomes delayed, it turns red. John may also pause his task if he needs to step away from his computer by clicking "Take a Break" on his task. Finally, when John is finished working, he can hit "Complete" on his task and will be asked to answer the documentation questions to describe the deliverables he produced and explain any decisions he may have made along the way.

Now that he is done, John can move on to other work while the client reviews what he did, closes his contract, and gives him a positive review on his freelance website.

We can see that John has many opportunities to review the project and its current progress. He can review notes that each worker has left after completing a task, review the inputs to his task, and review his task. He can also chat with the client and ask clarifying questions before and during his work. Despite all of these opportunities to confirm what he needs to do, we propose that it is in the very first time John reads about the project that most strongly influences his understanding about the project.

# Chapter 4

# Evaluation

The purpose of this study is to evaluate the impact that the first piece of information a worker receives about a job shapes their understanding. We are introducing a manipulation of the "job available email" stage of hiring and examining whether we can improve a worker's understanding of the project as a whole as well as their specific task.

In particular, we are using QUD models and information foraging theory to examine how workers extract important information about a job through either a non-interactive or an interactive information hierarchy. While Foundry has many tools to help a worker onboard into a project, because online experts must work time and cost effectively, we propose that the first information they get is the most crucial. I expect workers to perform both qualitatively and quantitatively better when onboarded to a project using an interactive information hierarchy because it allows them to more explicitly examine clusters of information in small, easily understandable chunks.

## 4.1 Methods

To evaluate rapid onboarding of online, on-demand workers, I conducted a study of 13 participants from the Upwork and Elance freelance websites. This between-subjects

study had two conditions through which the workers were offered details about the project and task they would complete.

The task we asked the workers to complete is for front end developers. Prior to this study, we ran several stages of a project to create a Q&A Website for Freelancers including design mockups, design review, design iteration, front end development, and back end set-up. For this task, the developer is asked to review a needfinding report that a researcher submitted and apply the requested changes to the exisiting front end code. Specifically, there were several navigation errors as well as a missing html page for the results of a search on the website. I chose this task because it required the worker to adhere to the design constraints of the existing website, follow the requirements of the needfinding document, but it still open-ended enough so we can assess the quality different workers deliver.

In the Non-Interactive Information Hierarchy condition, freelancers were delivered all of the project information through a single email that had non-interactive information hierarchy. The sections of the project information were all signalled with boldface labels in the email (Figure 4.4).

In the Interactive Information Hierarchy condition, freelancers received a short email reporting that a job was available, and they could click a link that would take them to a website where they could click through the details of the project as an interactive information hierarchy. The website opens to a statement noting that the job is for a front end developer and that they can click through the site to learn more and be hired. They click next to go to a new page each for the project overview and task and the task page expands to show detail about the expected time and the final deliverable (Figure 4.3). After this onboarding, all participants were given a link to the Foundry website and could begin working on their development task.

Workers were recruited via job invitations on Upwork and Elance. They were invited to join a panel of front end developers to receive job offers on-demand via email. Pre-requisites to be accepted to the panel include prior experience with html, css, javascript, and twitter bootstrap, as well as having worked at least 10 hours on a freelance website. Some workers had worked over 1000 hours. Finally, they needed to have either tested or self-reported at least conversational proficiency with the English language.

47 workers were recruited to this panel, of which 13 were available and accepted our job. Those workers came from a wide variety of countries: Germany, India, Pakistan, the United States, China, Ukraine, and Venezuela. Three participants were female. On both websites, when users apply and accept the job, they set their own hourly rate. The hourly rate for these workers ranged from $6.67/hr to $50/hr.

After accepting the job offer, workers went through the panel training mentioned previously in Chapter 3. Workers were sent on-demand job invitations via email over the course of two weeks. The initial email said a 3-hour job was immediately available for a front end developer and asked workers to respond within an hour if they were available (Figure 4.1). This allowed us to avoid pre-assigning workers to a condition, given that it is unclear how many workers would end up responding to the job post. Conditions were assigned alternating between each person to respond to the post.

After responding to the email within the given time frame, workers were sent the project and task details via one of the two conditions. In the Non-Interactive Information Hierarchy condition, workers are sent an email with all of the details of their specific job and asked to reply saying "I accept" to take the job (Figure 4.4). In the Interactive Information Hierarchy condition, workers are sent a shorter email with a link to a website that will contain job details and hiring information (Figure 4.2). After clicking through the website, the worker submits a form asking them to

Hello,

This is a message from the Stanford HCI Team to let you know that a job for a front-end developer is available. The task is 3 hours long.

If you are interested and available to work today, please respond to this message within the next hour to receive more information about the task.

Best,
Stanford HCI

P.S. Do not worry if you do not respond within the hour or are not available now. More jobs will be sent out this week. You may also respond and tell us times that would work better for your schedule.

**Figure 4.1:** "From Stanford HCI: A Job is Available" Email. Every worker received this email first and if they replied, they would be sent one of the two conditions containing information about the project and the task.

A job requiring a Front End Developer - Platform 2 for the Implement Front-end of Needfinding Changes task for the *Freelancer Question & Answer Web App* project has become available.

To review the job and start the hiring process, go to this link: ato1120.github.io/flashinfohierarchy/

Best,
Stanford HCI Research Team

**Figure 4.2:** The Interactive Information Hierarchy condition initial email. Workers must click the link in the email to access the website where they can navigate through the information for the project and task.

summarize the goal of the project and of their specific task (Figure 4.3).

When the worker has accepted the task, they receive an invitation to Foundry where they can view the project, their task, the input to their task, and chat with the client. They are greeted when they log in, sent an offer on the freelance website through which they applied, and can begin working.

In order to evaluate the effectiveness of our onboarding, we will measure a number of qualitative and quantitave metrics - assessing both the manner in which the worker worked as well as their final code deliverable.

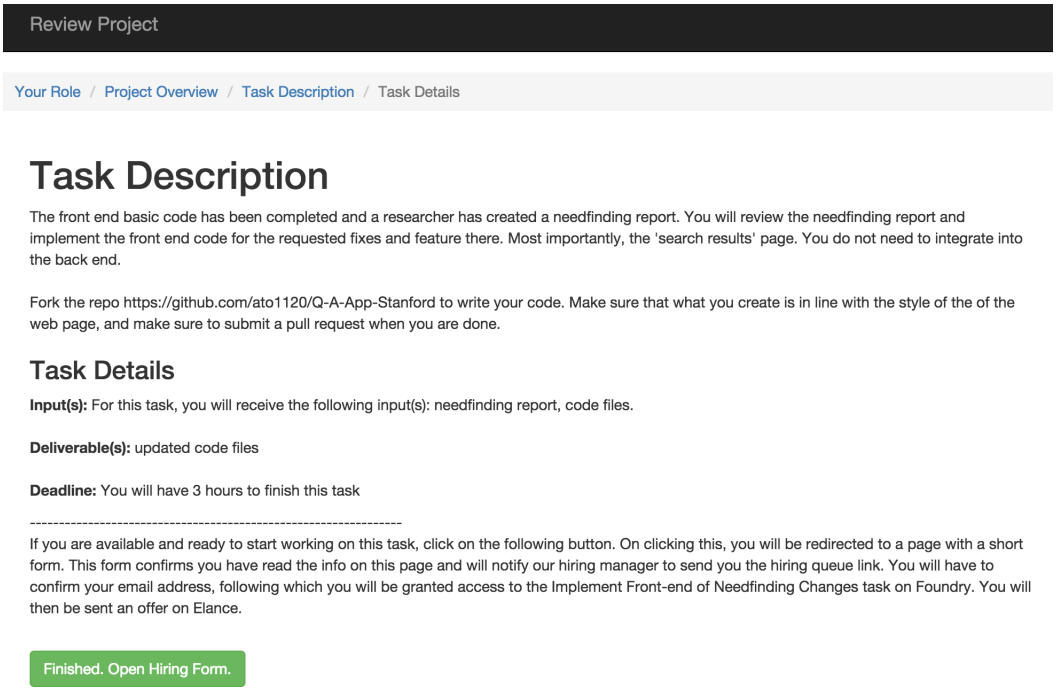


**Figure 4.3:** The Interactive Information Hierarchy website. Clicking next or going back up the navigation of the site expands or contracts the information on the screen.

For each worker we were able to measure:

1. Time it took to accept the job after reviewing the project and task details
2. How many of the 4 navigation errors were fixed correctly
    a. Navigation bar made consistent on each page.
    b. Link to single-question.html when you click on a question.
    c. Link "Join Now" to the registration page.
    d. Link "Ask" to the ask question page.
3. How many of the 4 search results specs are fulfilled
    a. Query reiterated at top of page
    b. List questions on page
    c. Link tags on home page to search results.

This is an email from the Stanford HCI Group notifying you that a job requiring a Front End Developer - Platform 2 for the Implement Front-end of Needfinding Changes task for the *Freelancer Question & Answer Web App* project has become available. Please take a look at the following job description to see if you are interested in and qualified to complete this task within the specified deadline.

**Project overview:** The goal of this project is to create a "Question & Answer" web application designed by freelancers for freelancers. The web application should be responsive so that it can be used on the computer, tablets and mobile phones. With the web application, users should be able to create and maintain a profile, ask a question, answer a question, view all existing questions and answers, and view the profiles of other users.

More details can be found in the project specification document.

**Task description:** The front end basic code has been completed and a researcher has created a needfinding report. You will review the needfinding report and implement the front end code for the requested fixes and feature there. Most importantly, the 'search results' page. You do not need to integrate into the back end.

Fork the repo https://github.com/ato1120/Q-A-App-Stanford to write your code. Make sure that what you create is in line with the style of the of the web page, and make sure to submit a pull request when you are done.

**Input(s):** For this task, you will receive the following input(s): needfinding report, code files.

**Deliverable(s):** updated code files

**Deadline:**You will have 3 hours to finish this task

If you are available and ready to start working on this task within 30 minutes of being hired, reply to this email saying “I accept.”. On replying, you will be sent a link to Foundry, where you will be granted access to the Implement Front-end of Needfinding Changes task on Foundry. Further, you must confirm that you have started working by accepting the contract on Upwork and signing in to Foundry via the link sent to you.

Best,
Stanford HCI Research Team

**Figure 4.4:** The Non-Interactive Information Hierarchy condition.

d. Link "Search" to search results page.

In order to study the quality of the final code, we asked three Computer Science Ph.D. candidates from Stanford University who focus in human-computer interaction to act as design reviewers and rate the search results page on a scale of 1 to 9. For the design review, they were asked to assess how well the page addressed the prompt from the needfinding report, how well the page fit in with the design of the rest of the website, and the overall feel of the page. These reviews were performed condition-blind and the reviewers were given no information about the workers who completed the task.

Information foraging theory supports two hypotheses about the behavior we expect to see. While an interactive information set should yield better foraging results due to easily navigable and digestable information chunks [10] , the barrier of clicking through too much information may hinder an information search [5]. A non-interactive information hierarchy has more information scent in that it displays all of the relevant information at once with clearly outlined section headers, but the overwhelming amount of information may cause a worker to skim rather than to read thoroughly as they would in the interactive setting.

## 4.2 Results

The results from this evaluation suggest that workers quantitatively and qualitatively perform better when onboarded using an Interactive Information Hierarchy (Table 4.5).

In our results we see that workers in the Interactive Information Hierarchy (mean =6.333, std. dev = 1.041, var = 1.084) condition received an overall better design score for their work than in the Non-Interactive Information Hierarchy (mean 6.048, std. dev = 2.119, var = 4.49). The highest and lowest score from the Non-Interactive

| Worker ID | Acceptance Time (min) | Navigation Errors Fixed | Search Results Specs | Design Score (1-9) |
|---|---|---|---|---|
| **Interactive Information Hierarchy** | | | | |
| 118 | 49 | 3 / 4 | 3 / 4 | 5.667 |
| 121 | 43 | 4 / 4 | 4 / 4 | 6 |
| 122 | 13 | 3 / 4 | 2 / 4 | 5 |
| 124 | 122 | 4 / 4 | 3 / 4 | 6.333 |
| 125 | 140 | 4 / 4 | 4 / 4 | 7.167 |
| 127 | 17 | 4 / 4 | 3 / 4 | 6 |
| 130 | 32 | 4 / 4 | 4 / 4 | 8.167 |
| **AVG** | **59.429** | **3.714 / 4** | **3.286 / 4** | **6.332** |
| **Non-Interactive Information Hierarchy** | | | | |
| 115 | 16 | 4 / 4 | 4 / 4 | 8 |
| 114 | 23 | 0 / 4 | 2 / 4 | 5.167 |
| 116 | 23 | 4 / 4 | 3 / 4 | 5.333 |
| 120 | 37 | 4 / 4 | 3 / 4 | 7.667 |
| 126 | 2 | 3 / 4 | 1 / 4 | 2 |
| 128 | 11 | 4 / 4 | 4 / 4 | 6.5 |
| 129 | 23 | 3 / 4 | 4 / 4 | 7.667 |
| **AVG** | **19.29** | **3.143 / 4** | **3 / 4** | **6.048** |

**Figure 4.5:** Table of results measuring the workers' performance. On average, the workers in the interactive hierarchy condition qualitatively better in the design of their final deliverable and quantitively more accurately fix all of the errors they are asked to fix. Workers in the interactive hierarchy also take much more time to respond and accept the task.

Information Hierarchy condition were both lower than their counterparts in the Interactive Information Hierarchy condition.

We also see that workers in the Interactive Info Hierarchy Condition (mean = 3.714, std. dev = 0.488, var = 0.238) out-performed the workers in the Non-Interactive Info Hierarchy condition (mean = 3.143, std. dev = 1.464, var = 2.143) in the number of specific criteria they were asked to fix in the code for both the navigation errors. The Interactive condition workers(mean = 3.286, std. dev = 0.756, var = 0.572)

also completed more features for the search results page than did the Non-Interactive (mean = 3, std. dev = 1.155, var = 1.334).

Although workers who complete walking through the interactive hierarchy perform well, 7 workers who expressed interest in the task initially decided not to accept the job when given the interactive hierarchy as compared to 3 who turned the job down in the non-interactive information hierarchy.

Finally, workers in the Interactive Info Hierarchy condition took far longer to accept the job than workers in the Non-Interactive Info Hierarchy condition.

# Chapter 5

# Discussion

Although the Interactive Information Hierarchy is clearly effective, it does provide a barrier to information gathering that workers must overcome. Several more workers in the interactive hierarchy condition backed out of the task than in the non-interactive condition. This is backed up by research noting that in information foraging on the web, the presence of delays or multiple web pages may stunt foraging and cause users to quit prematurely [5]. While workers who complete walking through the interactive website seem to complete better work, future iterations of this onboarding method may need to be tweaked to signal to workers earlier how long it will take to complete this onboarding.

The Interactive Information Hierarchy also provided a seemingly obvious but unexpected affordance. By asking workers to summarize what they just read in the hiring form, we were able to assess the worker's understanding of the project before they began. This ended up being very useful. One of the 7 workers who turned down the job did so because after we assessed her responses, we discovered she had a complete misunderstanding about the job. We expected workers to submit summaries along the lines of "The project's goal is to complete a freelance Q & A website" and "my job is to review the needfinding report and update the front end code." However,

this worker wrote, "We are building a mobile app that shares your current location with your friends. You can invite friends to join you on certain activities" and that her job was to conduct a design review. This review period is crucial for hiring workers reliably and preventing overeager workers from completing tasks that are unrelated or unproductive to a project's goals.

In the flash teams work and in pilot organizations we have seen overconfident workers begin working without asking any clarifying questions. On occasion these workers will similarly have a complete misunderstanding about the purpose of the project or their specific task within it. In those cases, their work must be paid for but usually goes unused. Although instant hiring is fast, reviewing a worker's understanding of their task importantly avoids this waste of time and money. With the hiring form, we can avoid this issue without having to manually quiz each worker before they start working if they really understand the task.

As of now, it is unclear whether the prolonged job acceptance time in the Interactive Information Hierarchy condition is good or bad. While a longer time may indicate that the worker is reading the job more carefully, worker 124 and 125's response times of over two hours indicate that they may have been distracted and forgotten about the task because the details were not immediately available as they may be in the Non-Interactive Information Hierarchy. Similarly, a short time as in the Non-Interactive Information Hierarchy enables flash organizations to move faster, but a 2 minute response from worker 126 indicates that likely very little of the email was read thoroughly.

## 5.1 Limitations

This study uses a small sample size, and thus needs to be replicated at a larger scale to confirm the results.

Another difficult factor with research using global markets is the language barrier. We communicate with many workers who are non-Native English speakers and highly varying degrees of English language proficiency. Our technologies must be able to overcome the language barrier, but in a sample size this small, it is unclear how much a difference in proficiency may have aided or hindered subjects in each condition. While the content for each condition was the same, those with high proficiency presumably will understand a task better than someone with a lower proficiency despite the condition. The interactive condition should still show an improvement in understanding despite the level of English proficiency.

Finally, with a sample size this small, we can also expect a high variance in the skills and experience of each individual worker. While each worker was recruited as an expert in their field and vetted for skills in html, css, javascript, and twitter bootstrap, some workers may have worked for many years on such projects, while other have worked a much shorter time span. Again, with a larger sample size, we may be able to mitigate the skew that this could create.

# Chapter 6

# Future Work

Rapid onboarding is just one aspect of dynamically assembling crowd workers to work on a flash organization. As discussed earlier, hierarchy and adaptive organization structure are also crucial to enabling an on-demand organization. Current work on flash organizations is piloting methods to enable an adaptive organizational structure.

There are also many fine-grained problems that can be addressed with rapid onboarding. As the organization scales, the need for more long-term middle management may become necessary. The idea of a manager who would stay on for a week to run several tasks in a part of the organization runs counter to the idea of on-demand hiring. One challenge is that a single manager cannot be online 24/7 and thus may have difficulty managing workers stemming from a global workforce. We can look to organizational behavior research to find structures that can work constantly but maintain organizational memory [14].

As an organization continues to grow and develop, we may also need to address the issue of re-onboarding workers to a project. If a project has developed significantly since a worker last saw it, their framing of the project must be updated and it may be worthwhile to assess their preconceived notions about the direction of the project

as a means for using their expertise to help improve the project.

We will also continue to assess and refine the tools we have built into Foundry for rapid onboarding. Given the results of this study it is important first to increase the sample size to confirm our results. We can include the interactive information hierarchy in the onboarding process but will want to brainstorm ways to avoid losing workers who may have otherwise accepted the job.

# Chapter 7

# Conclusion

Onboarding to a project using an interactive information hierarchy allows workers to rapidly get up to speed on a project, understand the work expected of them, and decide whether or not to accept a job quickly. This is cost and time effective both for the workers and for the clients and is crucial to enabling flash organizations.

We show that by introducing a more interactive set of information in a worker's first message about a job, we can improve the worker's understanding of the project, their role, and the quality of their final deliverable. Tools that aid the worker's understanding require further work, in particular to overcome the fatigue barrier that navigating the interactive information hierarchy presents.

Creating an on-demand, online organization is an exciting prospect that is within our conceptual grasp. Taking advantage of the online expert crowd enables a client, who may have little to no experience with large-scale management, to complete complex projects at scale. By using rapid onboarding techniques workers can begin work on a project with minimal management overhead for the client and receive much more consistent and high-quality results.

# Bibliography


[1] Elance. 1, 6

[2] Upwork. 1, 6

[3] Sta an Larsson. Questions under discussion and dialogue moves. 1998. 3, 7

[4] Paul André, Aniket Kittur, and Steven P Dow. Crowd synthesis: Extracting categories and clusters from complex data. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 989–998. ACM, 2014. 6

[5] Alan R Dennis and Nolan J Taylor. Information foraging on the web: The effects of acceptable internet delays on multi-page information search behavior. *Decision Support Systems*, 42(2):810–824, 2006. 8, 23, 26

[6] Lyn Gattis. Planning and information foraging theories and their value to the novice technical communicator. *ACM Journal of Computer Documentation (JCD)*, 26(4):168–175, 2002. 9

[7] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 1301–1318. ACM, 2013. 1, 6

[8] Anand Kulkarni, Matthew Can, and Björn Hartmann. Collaboratively crowdsourcing workflows with turkomatic. In *Proceedings of the ACM 2012*

*conference on Computer Supported Cooperative Work*, pages 1003–1012. ACM, 2012. 6

[9] Knud Lambrecht. *Information structure and sentence form: Topic, focus, and the mental representations of discourse referents*, volume 71. Cambridge university press, 1996. 8

[10] Peter Pirolli and Stuart Card. Information foraging in information access environments. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 51–58. ACM Press/Addison-Wesley Publishing Co., 1995. 3, 8, 9, 23

[11] Peter Pirolli and Stuart Card. Information foraging. *Psychological review*, 106(4):643, 1999. 8

[12] Daniela Retelny, Sébastien Robaszkiewicz, Alexandra To, Walter S Lasecki, Jay Patel, Negar Rahmati, Tulsee Doshi, Melissa Valentine, and Michael S Bernstein. Expert crowdsourcing with flash teams. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 75–85. ACM, 2014. 1, 4, 6, 7, 11

[13] Craige Roberts. Information structure in discourse: Towards an integrated formal theory of pragmatics. *Working Papers in Linguistics-Ohio State University Department of Linguistics*, pages 91–136, 1996. 7

[14] Melissa Valentine. Team scaffolds: how minimal in-group structures support fast-paced teaming. In *Academy of Management Proceedings*, volume 2012, pages 1–1. Academy of Management, 2012. 29

[15] Alan Wexelblat and Pattie Maes. Footprints: history-rich tools for information foraging. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 270–277. ACM, 1999. 8

[16] Anbang Xu, Shih-Wen Huang, and Brian Bailey. Voyant: generating structured feedback on visual designs using a crowd of non-experts. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 1433–1444. ACM, 2014. 6