Improving the Quality and Efficiency of Data Collection in Developing Regions - Thesis Proposal

Kuang Chen

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2010-134
http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-134.html

November 2, 2010

Data in the Last Mile: Improving the Quality and Efficiency of Data Collection in Developing Regions

Thesis Proposal

Kuang Chen
kuangc@cs.berkeley.edu
University of California, Berkeley

July 26, 2010

1 Introduction

In today's world, information is a form of infrastructure, no less important to modern life than roads, power and water. Low-resource organizations in the developing world struggle not only with traditional infrastructural issues, but as well with obtaining accurate and timely data. They are challenged by the lack of expertise, resources and literacy of workers, as well as the inertia of existing paper workflows, and an ever increasing demand for data.

In some domains, such as health care, data collection is especially critical: an error or a delay can have drastic consequences. For example, the World Health Organization estimated there to be 350 million cases (low end) of malaria last year; if half of these cases resulted in a clinic visit, documented by a single form field with 1% error rate, then 1.8 million cases of malaria would be incorrectly recorded. In reality, we have seen error rates in health programs to often be much higher.

Data collection involves one or more *capture* and *entry* stages, depending on particular data collection technologies and quality assurance techniques. Paper processes capture information in printed forms, which data entry clerks enter into an electronic interface. The practice of *double-entry* – repeating the entry step and comparing results – is a standard quality assurance technique. Mobile and kiosk interfaces combine the capture and entry steps into one, and rely on user interface feedback for quality assurance.

Current best practices for quality and efficiency come from the field of survey design [3], which offers principles that include manual form layout, question orderings and input constraints, as well as double-entry of paper forms. These decades old practices merit reconsideration. For both paper forms and direct electronic entry, we posit

that data-driven and more computationally sophisticated approaches can significantly outperform existing methods in both accuracy and efficiency.

To help these organizations cope with the quality challenges in data collection, we developed USHER [1], a theoretical, data-driven foundation for improving data quality during entry. Our key insight is that existing form data can automatically inform the forms design and usage, improving quality and accuracy. Based on prior data, USHER learns a probabilistic model over the questions of an arbitrary form. USHER then applies this model at every step of the entry process to improve quality and efficiency. First, before entry, USHER helps with design to capture the most important values of a form instance as quickly as possible. During entry, it dynamically adapts the form to the values being entered, and enables real-time feedback to guide the data entry clerk toward intended values, based on an underlying cognitive model of data entry. After entry, USHER re-asks questions that it deems likely to have been entered incorrectly. We used simulation as well as user study *in situ* to verify the effectiveness of our approaches.

To address the efficiency challenges in data entry systems, we propose SHREDDR, a work in progress. Our key insight is that the entry task is an *entropy-reduction* problem: we can drastically improve digitization throughput if we compress the input to the human brain, and design input interfaces that appropriate match expected information in a set of tasks. First, SHREDDR provides a simple interface for schema extraction from a scanned form, generating a corresponding electronic form specification. Next, the system shreds scanned forms into image fragments and estimates values via optical character recognition (OCR). Fuzzy OCR values are used to build an USHER model, which we use to drive novel algorithms that assign and optimize the confirmation of fuzzy values against the corresponding image fragment in a variety of input interfaces. Tasks are assigned over an elastic labor pool (crowd sourcing) to manage latency and availability.

2 A Foundation for Data Accuracy

USHER is a principled foundation for data entry. Using previous form submissions, USHER learns a probabilistic model over the questions of the form. USHER then applies this model at every step of the data entry process to improve data quality.

Since form layout and question selection is often ad hoc, USHER optimizes question *ordering* according to a probabilistic objective function that aims to maximize the information content of form answers as early as possible. Applied before entry, the model generates a static but entropy-optimal ordering, which focus on important questions first; during entry, it can be used to dynamically pick the next best question, based on answers so-far — appropriate in scenarios where question ordering can be flexible between instances.

Applying its probabilistic model during data entry, USHER can evaluate the conditional distribution of answers to a form question, and make it easier for likely answers to be entered. For difficult-to-answer questions, such as those with many extraneous choices, USHER can opportunistically *reformulate* them to be easier and more congruous with the available information. In this way, USHER effectively allows for a

principled, controlled tradeoff between data quality and form filling effort and time.

Finally, the stochastic model is consulted to predict which responses may be erroneous, so as to *re-ask* those questions in order to verify their correctness — we call this the *contextualized error likelihood* principle. We consider re-asking questions both during the data entry process (integrated re-asking) and after data entry has been finished (post-hoc re-asking). In both cases, intelligent question re-asking approximates the benefits of double entry at a fraction of the cost.

We evaluated these components of USHER using two real-world data sets – direct electronic entry of survey results about political opinion and transcription of paper-based patient intake forms from an HIV/AIDS clinic in Tanzania. We performed simulation experiments, in which we aimed to verify hypotheses regarding three components of our system: first, that our data-driven question orderings ask the most uncertain questions first, improving our ability to predict missing responses; second, that our re-asking model is able to identify erroneous responses accurately, so that we can target those questions for verification; and third, that question reformulation is an effective mechanism for trading off between improved data quality and user effort.

We recall the results from the *patient* data in Figure 1, which demonstrate that USHER can improve data quality considerably at a reduced cost when compared to current practice:

- Chart A shows the results of the ordering simulation experiment, in which we posited a scenario where the data entry worker is interrupted while entering a form submission, and thus is unable to complete the entire instance. Our goal is to measure how well we can predict those remaining questions under four different question orderings: USHER's pre-computed static ordering, USHER's dynamic ordering (where the order can be adjusted in response to individual question responses), the original form designer's ordering, and a random ordering. The USHER orderings are able to predict question responses with greater accuracy than both the original form ordering and a random ordering for most truncation points

- Chart B shows the results of the reformulation experiment, in which we simulate form filling with a background error rate and time cost in order to evaluate the impact of reformulated questions. During simulated entry, when a possible response a is at the mode position of the conditional probability distribution and has a likelihood greater than a threshold t, we ask whether the answer is a as a reformulated binary question. If a is not the true answer, we must re-ask the full question. Results are averaged over each instance in the test set. Our results confirm the hypothesis that the greater the number of additional reformulated questions we ask, the lower the error rate.

- Chart C shows the results of the re-asking simulation experiment, in which our hypothetical scenario is one where the data entry worker enters a complete form instance, but with erroneous values for some question responses. Specifically, we assume that for each data value the data entry worker has some fixed chance p of making a mistake. When a mistake occurs, we assume that an erroneous value is chosen uniformly at random. Once the entire instance is entered, we feed the

entered values to our error model and compute the probability of error for each question. We then re-ask the questions with the highest error probabilities, and measure whether we chose to re-ask the questions that were actually wrong. Our error model is able to make significantly better choices about which questions to re-ask than a random baseline. In fact, for $p = 0.05$, which is a representative error rate that is observed in the field [10], USHER successfully re-asks all errors over 80% of the time within the first three questions in both data sets.

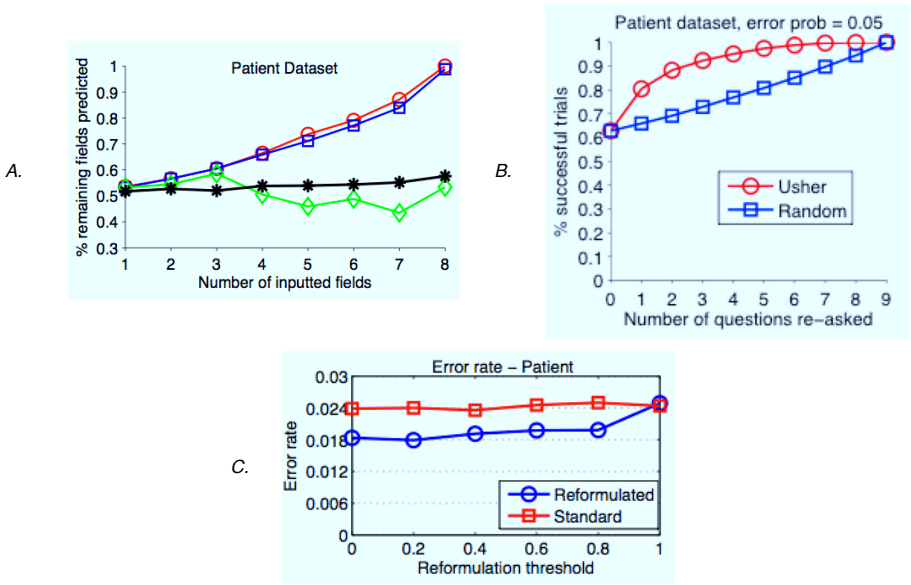

Figure 1: Results of the USHER simulation experiments for the *patient* data set. A. Ordering simulation experiment. The *x*-axis measures how many questions are filled before the submission is truncated. The *y*-axis plots the average proportion of remaining question whose responses are predicted correctly. B. Reformulation experiment. The *x*-axis shows the reformulation thresholds; when the threshold = 1, no question is reformulated. The chart shows the overall error rate between reformulated and standard entry. C. Re-asking simulation experiment. The *x*-axis measures how many questions we are allowed to re-ask, and the *y*-axis measures whether we correctly identify all erroneous questions within that number of re-asks. The error probability indicates the rate at which we simulate errors in the original data.

With USHER, we showed that a probabilistic approach can be used to design intelligent data entry forms that promote high data quality. USHER leverages data-driven insights to drive several disparate approaches to improving data quality for data entry. The three major components of the system — ordering, re-asking, and reformulation — can all be applied under various guises before, during, and after data entry. This suggests a principled roadmap for future research in data entry. For example, one combination we have not explored here is re-asking before entry. At first glance this may

appear strange, but in fact that is essentially the role that cross-validation questions in paper forms serve, as pre-emptive reformulated re-asked questions.

We can also extend this work by enriching the underlying probabilistic formalism. Our current probabilistic approach assumes that every question is discrete and takes on a series of unrelated values. Relaxing these assumptions would make for a potentially more accurate predictive model for many domains. Additionally, we would want to consider models that reflect temporal changes in the underlying data. Our present error model makes strong assumptions both about how errors are distributed and what errors look like. On that front, an interesting line of future work would be to learn a model of data entry errors and adapt our system to catch them.

Finally, our industry collaborators in survey research are interested in adapting this approach to the related problem of conducting online surveys. Here, we will have to deal explicitly with potential for user *bias* resulting from adaptive feedback. This concern is mitigated for *intermediated* entry, where the person doing the entry is typically not the same as who provides the data. They plan on exploring how USHER's predictive model can be used to detect problematic user behaviors, including detecting user fatigue and *satisficing*, where a respondent does just enough to satisfy form requirements, but nothing more [3].

3 Enabling Adaptive Feedback

In order to study the bottom line effect of USHER on data quality, we built a data entry system, which uses USHER models to drive dynamic feedback of the user interface during data entry [2]. In this work, we designed a number of intelligent input adaptations, show in Figure 2:

- 1) Setting *defaults* corresponding to highly likely answers: this technique changes an entry task to a confirmation task, which we show has the potential to significantly improve accuracy and efficiency.
- 2) Dynamically re-ordering and highlighting likely options in the input *widgets*: these mechanisms *guides* the user towards more likely values, and away from unlikely ones.
- 3) Providing automatic *warnings* when the user has entered an unlikely value: by warning the user about particularly unlikely values, we approximate double entry at a fraction of the cost.

To evaluate these data-driven feedback mechanisms, we conducted a user study measuring improvements in accuracy and efficiency with professional data entry clerks working on real patient data from six clinics in rural Uganda.

We recall the high level results about the effect of adaptive feedback mechanisms on error rate in (Table 1). The *widget* and *warning* mechanisms improved quality by 52.2% and 56.1%, with marginal significance. The improvement by the *defaults* mechanism was not statistically significant. More detailed analysis showed that *radio button* widgets exhibit even better results: the error rates decreased by 54-78%. The impact of our adaptations on entry cost (time) varied between -14% to +6%.

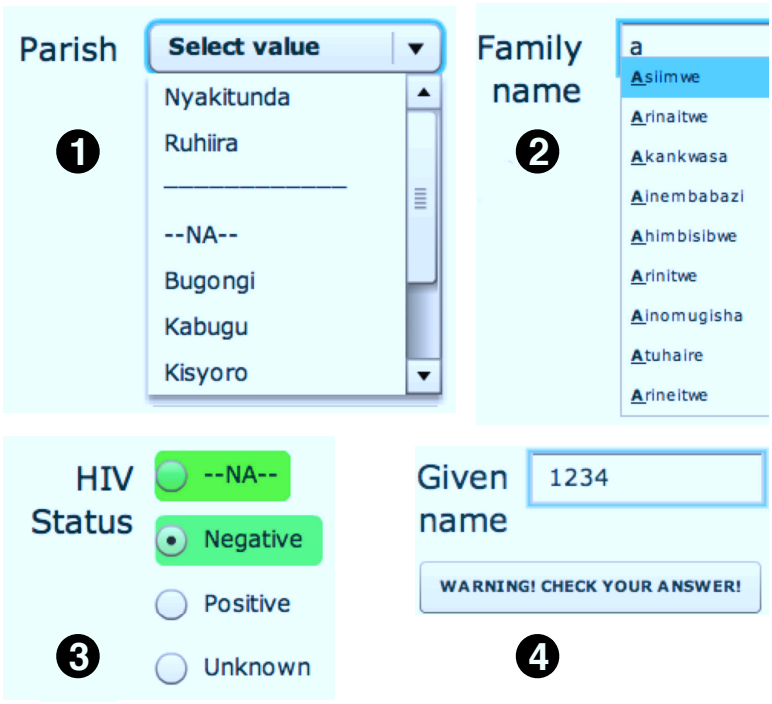

Figure 2: (1) drop down split-menu promotes the most likely items (2) text field ranks autocomplete suggestions by likelihood (3) radio buttons highlights promote most likely labels (4) warning message appears when an answer is a multivariate outlier.

Interesting future directions include developing a version of this system that can work with existing data collection software on mobile phones. We expect that a large number of ongoing mobile data collection projects in the developing world will benefit from this approach.

We believe USHER has wide applicability for improving data collection. In our work so far, we have applied this approach to data *quality* and the costs of quality assurance. In the next sections, we will discuss our plan to tackle the overall challenges of data entry *efficiency*.

4 Capture vs. Entry

For many organizations working in the developing world, the demand for data keeps increasing while the budgets do not. Indeed, the World Bank reports, "Prioritizing for monitoring and evaluation (M&E) has become a mantra that is widely accepted by governments and donors alike." [4]. These ever-increasing data requirements cause organizations to implement one-off, haphazard data collection systems to minimally meet reporting requirements. These systems require a large share of local resources,

Feedback type	Error rate	vs. *plain*	Adj. p-value
plain	1.04%		
defaults	0.82%	-21.0%	0.497
widgets	0.50%	-52.2%	0.097
warnings	0.45%	-56.1%	0.069

Table 1: Mean error rates for each feedback variation (across all widget types) and comparisons to the *plain* control-variation.

and too often, limit the effectiveness of these organizations. Ironically, a top down push for more data results in the de-prioritization of data for local service delivery and operational optimization. In our next phase of research, we aim to address these challenges with an approach that we believe can improve data input efficiency by an order of magnitude.

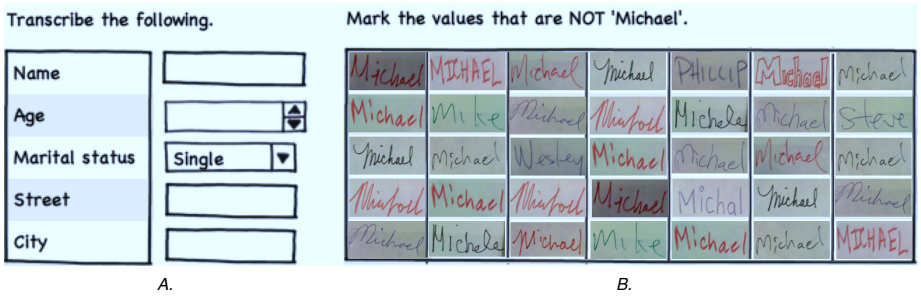

Figure 3: Two example interfaces for data entry.

We illustrate the key insight of our approach with a simple example. Suppose we have a large set of paper forms to transcribe. Each form has 5 questions: name, age, marital status, street, city. Figure 3 shows two input interfaces: A, on the left, is a traditional input interface for paper data entry; B, on the right, concerns only the 'name' question, but shows 35 scanned image fragments at once. Furthermore, candidate answers have been automatically preprocessed, such that, the system has cherry-picked the images with the highest likelihood of being "Michael". We posit that the batch-oriented interface on the right can have much higher throughput, and similar accuracy as traditional approaches.

Our example is illustrative of a fundamental conflict in the data entry workflow (Figure 4), between data *capture* and data *entry*. To understand this conflict, let us look from the perspective of information theory: the goal of the *capture* step is at odds with the goal of the *entry* step in term of information content. Forms by definition, seek to capture as much *unknown information* as possible from the respondent; in other words, they aim to maximize entropy. In contrast, entry involves human processors transforming *known information* from one format to into another; the aim is to maximize throughput, while preserving transmission fidelity.

From an information theoretic perspective, the data entry clerk is a data processor

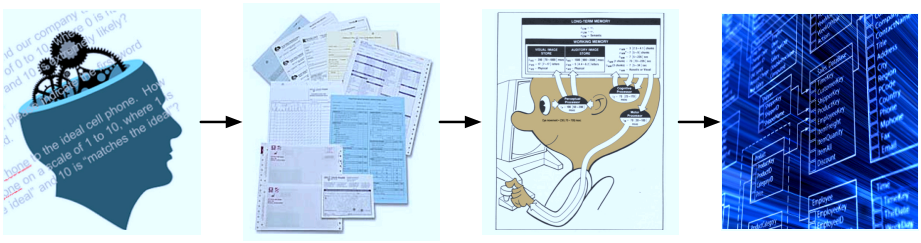

Figure 4: Data flow from respondent to database.

with certain performance characteristics, which are key for optimizing data throughput. Notably, a person's visual bandwidth and short-term memory are limited to a small number (7 ± 2) of distinct items [8, 9]. When this number is exceeded, the potential for error increases [7]. In essence, this is size of the human *cache*.

Consider the traditional "row"-oriented manner of entry (Figure 3 A.): entering the value for one question at a time across the database *row* that holds a form instance's answers. Depending on the domain size of the question being examined, the entry clerk must invalidate his cache between questions, whenever next question's domain differs from that of the previous. We posit that the cost of this human context switch, or *cache invalidation*, can be significantly reduced by *compression*.

We have witnessed *compression*, in action, during our user studies: when an entry clerk sees a sequence of questions with the same domain, for example, a sequence of *true/false* questions, she will remember and enter up to five values at once. Extending our computational metaphor: she is being cache-aware and opportunistically batching her input, reducing the cost of cognitive context switches.

Another notable property of the human processor the exponential decay of short term memory. If the input interface displays one field at a time, the duration of entering those five batched values would take longer than if all five fields were shown at once. However, if the interface shows six values, then after filling the batched five, the remaining input field would compel the entry clerk to transfer only one value the next time – a loss of efficiency. In general, we propose the challenge of parameterizing the input interface to match the cognitive and physical effort of an entry task. For our human data processors, this is a brain *cache alignment* problem.

The controversial matrix-question type highlight the tension between capture and entry. Matrix questions are a set of questions with the same answer domain, for example: "Rate these items from *poor* to *excellent*)". A long record of research shows that matrix questions increase satisficing and survey abandonment [5], and are not recommended for *capturing* information. In contrast, a sequence of questions with the same *poor-excellent* answer domain presents the data clerk with an opportunity for more efficient *entry*.

In the following sections, we propose to exploit the gap between capture and entry. First we will discuss the design space of several optimization approaches. Next, we will present our plans for the SHREDDR data entry system.

5 Confirmation instead of Entry

Let us suppose that the SHREDDR system enables form contents to be split apart into fragments of values corresponding to each question. Many automatic techniques, such as optical character recognition (OCR) and USHER, exist for predicting the value in each fragment. However, no automatic prediction is perfect, and we assume that the final answer must be check by a human being.

> **Hypothesis 1.** A confirmation task is more efficient than the equivalent entry task.

Our first modest hypothesis is based on the fact that using automatic predictions decrease task entropy, and thus have the potential to increase throughput.

6 Column-oriented Entry

In going about optimizing the human processor, we are inspired by the column-oriented database literature [11]. The main argument for a column store is that operating on data from a single column allows better compression, enabling vectorized operators and block-tuple processing. The basis of better compression comes from the nature of entropy in rows versus columns. A row tuple inherently exhibits higher entropy because values are of different type and lie in different domains. In comparison, a column is of uniform type and domain, and thus having lower entropy.

> **Hypothesis 2.** A column-ordered entry approach is more efficient for data entry than a traditional row-ordered approach.

Our hypothesis is based on the intuition that column-ordered tasks have lower entropy and thus increase cache locality. Refer to Figure 5; we are interested in assessing the impact of row (A) versus column (B) ordered task streams.

Column-data enables additional techniques for compression not available to row data, such as run-length encoding. Furthermore, column stores have demonstrated an order of magnitude better compression [6] by *sorting* along column values. To further take advantage of column data compressibility, we would like to sort entry tasks by value. However, we do not know the actual values — what we can do is sort by predicted value. As discussed above, we can use USHER and OCR results to sort a set of tasks by their predicted answer.

> **Hypothesis 3.** A sorted-column-ordered approach is more efficient for data entry than a column-ordered approach.

The same intuition as above applies: a sorted-column approach has even lower entropy for a same size set of entry tasks (see Figure 5 C.).

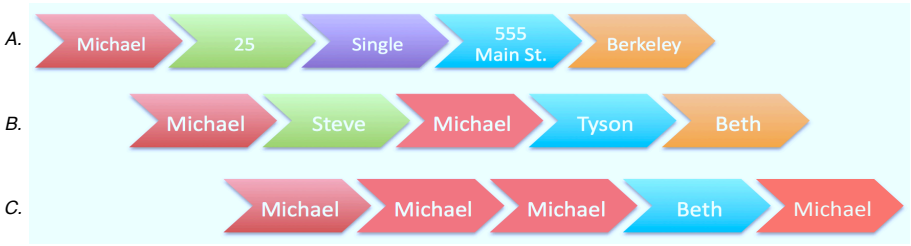

Figure 5: A. shows row-ordered sequence of questions. B. shows column-ordered sequence of questions. C. show sorted-column ordered sequence of questions.

7 Pipelined Entry

For data flow problems, opportunities for optimization are generally created by injecting levels of indirection and parallelism. Let us consider the data entry workflow like a data flow with human beings in the loop. Each optimization technique we have described concern automatic entropy reduction for the human computation steps of the data flow. Taking note of this meme, we observe that the mechanisms that we have so far proposed, fall in a pattern of *pipelined entropy reduction* for data entry:

- Shredding a form into fragments allows it to be worked on in parallel by multiple workers at once.
- USHER and OCR in conjunction, serve as the first stages of entropy reduction – turning entry tasks into binary verification tasks.
- Using column-ordering reduces entropy, allowing workers to work on a greater number of tasks at once.
- Sorting by prediction enables another order of magnitude in entropy reduction, further increasing parallelism.

Let us suppose that SHREDDR aggregates tasks at a central controller and enjoys an unlimited number of workers from an elastic labor pool. In such a scenario, we can enjoy a sufficient number of workers and tasks to create an economy of scale and task specialization.

8 Entropy-aware Input Interfaces

In addition to the above techniques, we can further pipeline the entry task by subdividing the process of verification among new input interfaces that sift the values into increasingly lower entropy bins.

> **Hypothesis 4.** The effort for a group of entry tasks T depends on the entropy of its values: effort $E = f(|T|, H(T), u)$, where $u \in U$, a set of input interfaces.

Recall the above discussion about human *cache alignment*. Given a set of entry tasks, predicted values and entropy, we want to appropriately parameterize the input interface. Since the design of input interfaces can vary widely in design choices, we limit our discussion to the *fan-out* factor, explained in the following example.

In Figure 6, the domain for a sequence of questions is A, B, C, D, E. Interface A filters out any value that is equal to a chose value — in this case 'A'. Interface B requires the worker choose between the two most likely values or 'Other'. Note that interface A only tells whether the answer is A – its *fan-out* factor $= 1$. Were we to only use this interface, for a domain of size N, we would need to iterate N times. Note that interface B disambiguates two values at a time (*fan-out* $= 2$), but can show fewer values at once than interface A, if we were to hold *effort* constant. For a domain of size N, similarly, we would need to use interface B $N/2$ times.

A. Click cells not equal 'Michael':

B. Click the best answer or 'Other'

Figure 6: A. shows an interface, in which the fan-out is 1. B. shows an interface, in which the fan-out is 2

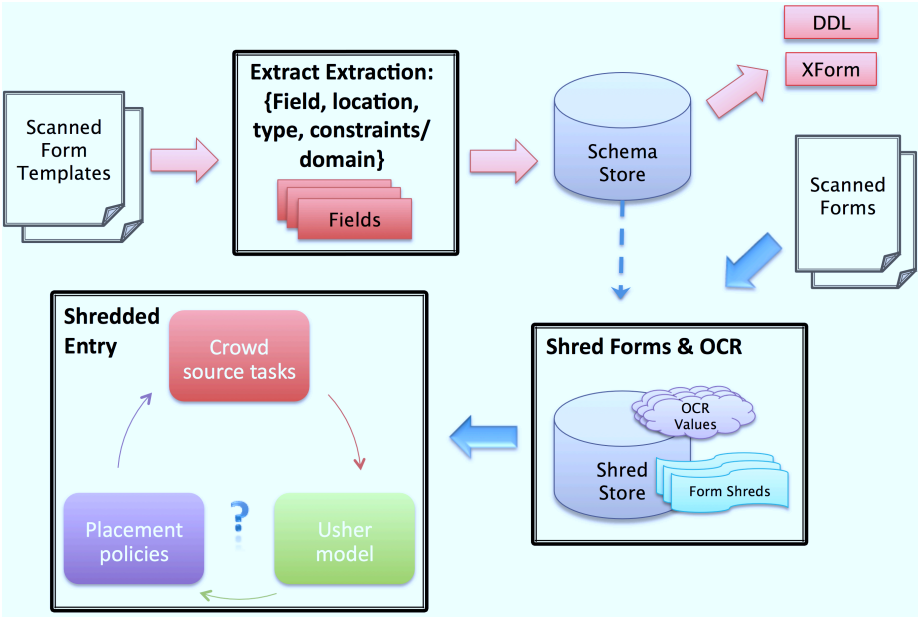

Figure 7: The Shredder system dataflow and components

This example illustrate a set of tradeoffs in interface design, and suggest that for a given workload, determined by a set of entry tasks, predicted values and entropy, there is a optimal interface. We propose to study interface efficiency with users, by varying the entropy level and number of tasks shown as experimental effects. We aim to derive a set of design guidelines for entropy-aware input interfaces.

9 SHREDDR

Many organizations desire electronic data collection, but the inertia of existing paper systems and the capital expenses of "starting over" can often prevent adoption of new technologies. In order for an organization with paper-based workflows to benefit from digital data collection, it needs to digitize a significant portion of its archive (a *backlog* problem), and create electronic versions of existing forms (a *migration* problem).

In addition to research contributions, we aim to build tools for organizations on the ground. As such, we propose SHREDDR (Figure 7), a system that addresses these specific entry efficiency challenges.

With a pile of paper forms and a scanner, SHREDDR enables *schema extraction* to address the migration problem: First, SHREDDR provides a simple web interface that guides a user to draw boxes on a scanned form image. Aided by optical character recognition (OCR), SHREDDR semi-automatically extracts both the form's data schema and data locations. Next, SHREDDR uses the extracted schema, coupled with a wizard interface, to semi-automatically generate an electronic form (Xform), as well

as its underlying database (SQL). With the form and database, an organization is bootstrapped to start electronic data collection with mobile devices, as well as transcription by workers.

After schema extraction, SHREDDR begins to digitize the backlog as follows: First, SHREDDR enables batch scanning of existing paper forms, shredding them according to extracted data locations into image fragments. SHREDDR uses OCR to automatically extract a fuzzy value from the shredded image fragments. It stores the value, confidence score and original image fragment in the database. After shredding, SHREDDR builds an USHER model using the OCR-extracted values. We use this model to optimize the final steps of digitalization:

1. With approximate values from OCR, we turn entry-tasks into lower-entropy *confirmation*-tasks.

2. By shredding forms into image fragments, we are free to reorder entry tasks; presenting workers with column-ordered tasks.

3. With the USHER model and OCR, SHREDDR predict and sorts entry tasks by the mostly likely answer, further reducing entropy in task sets.

4. Using crowd sourcing, SHREDDR enjoys sufficient number of simultaneous workers, to whom we can dynamically queue tasks to both maximize the USHER model's predictive ability, and while employing all of the above optimizations.

5. Tasks can be grouped by entropy and worked on iteratively – response values are checked and re-checked by workers using different input interfaces, which dynamically adapt to the entropy of the task set.

References

[1] K. Chen, H. Chen, N. Conway, T. S. Parikh, and J. M. Hellerstein. Usher: Improving data quality with dynamic forms. In *Proceedings of the International Conference on Data Engineering*, 2010.

[2] K. Chen, T. Parikh, and J. M. Hellerstein. Designing adaptive forms for improving data entry accuracy. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 2010.

[3] R. M. Groves, F. J. Fowler, M. P. Couper, J. M. Lepkowski, E. Singer, and R. Tourangeau. *Survey Methodology*. Wiley-Interscience, 2004.

[4] I. E. G. (IEG). *Monitoring and Evaluation: Some Tools, Methods and Approaches*. World Bank, Washington, DC, 2004.

[5] J. A. Krosnick. The threat of satisficing in surveys: the shortcuts respondents take in answering questions. Survey Methods Centre Newsletter, 2000.

[6] D. Lemire, O. Kaser, and K. Aouiche. Sorting improves word-aligned bitmap indexes. *Data and Knowledge Engineering*, 69(1), 2010.

[7] J. Martin. *Design of Man-Computer Dialogues*. Prentice-Hall, Inc., 1973.

[8] G. A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for information processing. *Psychological Review*, 63(2), 1956.

[9] K. Mullet and D. Sano. *Designing Visual Interfaces: Communication Oriented Techniques*. Prentice Hall, 1995.

[10] S. Patnaik, E. Brunskill, and W. Thies. Evaluating the accuracy of data collection on mobile phones: A study of forms, sms, and voice. In *ICTD*, 2009.

[11] M. Stonebraker, D. Abadi, A. Batkin, X. Chen, M. Cherniack, M. Ferreira, E. Lau, A. Lin, S. Madden, E. O'Neil, et al. C-store: a column-oriented DBMS. In *Proceedings of the 31st international conference on Very large data bases*, 2005.