EXTRACTING DATA FROM FREE TEXT FIELDS: ASSURING DATA QUALITY FOR ERP IMPLEMENTATION

(Practice-Oriented Paper)

M. David Allen
Institute for Data Research
mda@idatar.com

Susan Carter
Institute for Data Research
scarter@idatar.com

Peter Aiken
Institute for Data Research
paiken@idatar.com

Mary Kay Cyrus
Defense Supply Center Richmond
MaryKayCyrus@dla.mil

Kathy Wade
Defense Supply Center Richmond
Kathy.W.Wade@dla.mil

Sid McCormac
Defense Supply Center Richmond
Sid.McCormac@dla.mil

Abstract: This experience paper describes a repeatable model developed to address a class of data quality problems encountered when converting text data to ERPs. Users often devise their own means of implementing system features not directly supported by the systems. Often they employ what are known as clear-text, free-text, or "comment" fields to support the desired features. Moving data from these fields to ERPs involves first extracting atomic data items. Unlike most data, free text is not subject to structural or practice-oriented data quality measures when it is created. This results in a range of data quality challenges ranging from typing errors to structural errors such as prime key mismatch, duplication, and other issues. In our experiences with one large government system, a number of challenges were encountered that contained enough internal differences to require the development of a more generic framework for addressing this type of problem. The specifics of the actual issues confronted are not as interesting as the lessons that can be learned from the general approach to problems of this type. The solution type developed demonstrated a positive return on investment to the government. We will discuss the challenges, the costs associated with continuing along the original path, the solution developed, and its applicability to other organizations and situations.

Key Words: Data Quality, Information Quality, unstructured text, extraction, regular expression, ERP migration

1. INTRODUCTION

The agency providing the environment for this research was a part of a larger organization whose task it was to ensure that the war fighter is adequately supplied for accomplishing their mission. Specifically, it

is tasked with managing a subset of the items that the US government purchases. Similar organizations exist that handle other aspects of the supply picture. Roughly $2.5 billion in goods and funding flows through the organization annually, with a purchasing base of approximately 1,050,000 items. The task of identifying which items are actively purchased and which are no longer needed, (such as parts for airplanes no longer used by the military) is itself quite a challenge. The existence of many outdated records tends to skew observations of how statistics should appear and how they actually appear.

The organization has been in the process of moving managed data towards an ERP. Preparing and moving the data has been a complex process as is frequently the case with new ERP systems. [1] For many data fields, the most current data was contained in a free text field ranging from one to 30 lines of text, with each line of text between 1 and 80 characters. This field contained English descriptions of the data item, as well as hints, notations, and other snippets suggesting certain values for other fields. The data came from manual entry by clerks directly into the system, and comes with all of the idiosyncrasies that one might expect from different personnel with different styles.

For this paper, the data dealt with was the NSN, or "National Stock Number" which is an identifier for items the government buys for supplying troops and ships. The NSN is a "smart key" that contains two portions – the FSC, and the NIIN. The FSC is the Federal Supply Code, and identifies the type of supply being purchased, such as packaged petroleum. The NIIN is the National Inventory Identification Number, which identifies a specific item. Two other fields frequently encountered were the CAGE and PART fields. CAGE codes identify specific vendors; to disambiguate purchasing of the same item from many different sources. PART numbers are the vendor's concept of a part number. While an NSN identifies an item for the government, (as in the fictional example "F-16 rearview mirror fuzzy dice") vendors may have a PART number for their product. There may be many CAGE/PART combinations associated with one NSN. Figure 1 shows the relationship between NSN, NIIN, FSC, CAGE, and PART numbers.

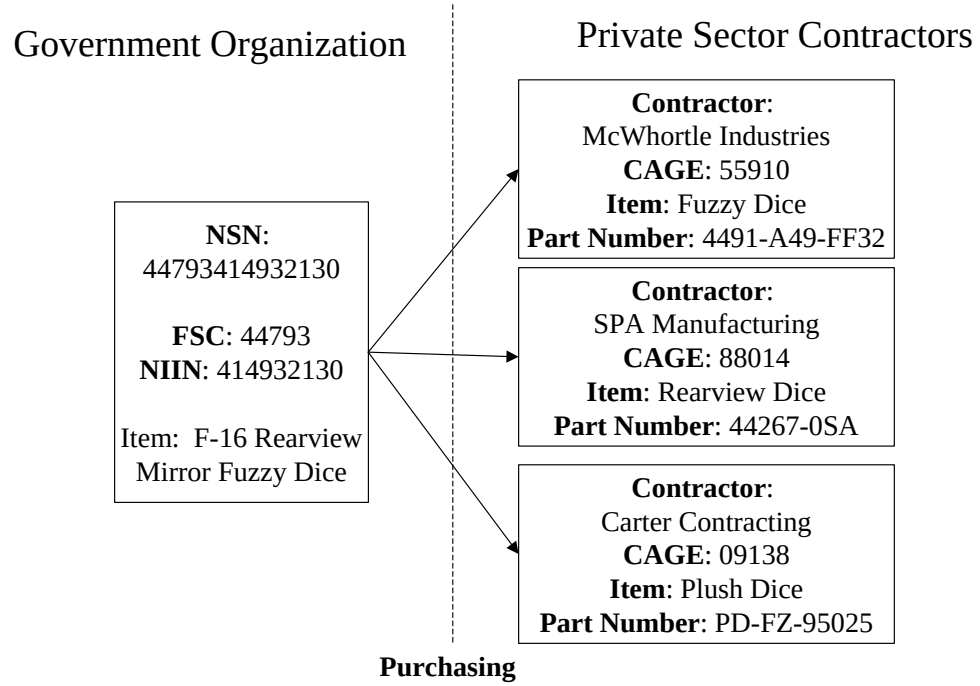

Figure 1: The relationship between various data elements in a sample problem space.

These data points form the core of the most important data stored by the organization. It supports the organization's key mission – to purchase and deliver goods.

2. THE PROBLEM AT HAND

Moving legacy data towards new ERP systems requires work to ensure the utility of the data that is being moved over. Since organizations must expend resources to move the data, it is vital to ensure that it is accurate so that the actual utility of the ERP can be maximized. [8] An ERP is no exception to the general rule for information systems – the system is only as good as the data that it stores and processes.

Part of the challenge in this situation was the flow of the data through the system. The database that was dealt with by the organization actually was completely refreshed on a quarterly basis from an upstream system. The system that was used would have modifications and changes made to it that could be negated with the next quarterly update unless the changes were communicated upstream. Each time the system was refreshed the data sources would begin to diverge, meaning that data ultimately extracted required validation against both systems – the local system and the upstream system to assess whether or not the correct information was being extracted.

2.1 The Importance of Fixing the Problem

Fixing this problem for the organization is critical because the acquisitions portion of the new ERP system can only function if it is given the correct information. Given the large volumes that the government may choose to purchase in, mistakes that are made in the acquisitions process can result in additional costs and increased time to delivery of the necessary goods. In the case of repair parts for jet fighters, delays and stocking problems can have very real and lethal implications in the field. There are a large number of pieces of equipment used by the government that have multiple critical parts, all of which must be in the right place at the right time to allow use of the equipment.

A good amount of data is stored in these systems pertaining to other aspects of the purchased items as well. For example, most NSNs have a Boolean field associated with them called FAT, or "First Article Testing". When an item is to be produced by a vendor, the first article testing Boolean flag indicates whether or not the vendor must submit their design along with a sample for testing to assure quality and adherence to specifications. Quality assurance and testing is key on many items bought by the government including engines, safety devices, and weapons. Personnel relying on poor quality or untested materials in the field might actually be worse than not having the equipment at all, since it might require troops to put trust in materials that should not be trusted.

In addition to all of the operational details and the need for accurate data, it is also critical that the data be moved out of the existing systems and into the implemented ERP because funding for the existing systems will be de-funded over a period of time. These deadlines have already been established and cannot be assumed to be malleable. As with many situations, if the data does not move into the ERP, it has nowhere else to go. When moving to the new ERP structure, it is also important that the organization be able to start with a clean slate of high quality data rather than beginning a new phase of data management with many of the same data quality challenges that they grappled with under the old system.

2.1 The Traditional Approach, Past Track Record and Results

As is typical in most organizations, the main portion of the earlier data quality effort was not specifically organized and executed – instead, as records were found that were erroneous, paper forms were filled out and filed with the keepers of the upstream data requesting a change to update the information. Systematic data quality effort was not recognized as a part of the organizational mission. As a part of the effort to migrate the data towards the new ERP system, a manual effort was underway to visually inspect each

record, (not just the ones that were encountered by chance) and compare the information gotten from those records with the various systems for purposes of validation. The manual approach is of course the slowest approach. Additionally, given the expertise and familiarity with the data and systems that was required to do this work, it tended to occupy otherwise productive employees who would better be used on a different portion of the conversion or data management effort.

Individual corrections made to the local and upstream data systems were ongoing within the organization and yielded better data in the instances of specific records. However, since there was no ongoing effort to address the reasons why the data became erroneous in the first place, correcting individual records failed to attack the overall problem in an effective way. The later organized manual effort was extremely effective in identifying and fixing erroneous records, but suffered from two problems. First, it still did not address the reason the errors were cropping up in the system, and second, the costs associated with it were fairly large considering the time investment required of the workers performing the inspection of records. Given the progress rate of the manual effort and the deadline for moving all legacy data over to the ERP, there was some question about whether or not the manual effort would succeed.

2.3 Research Objectives

The effort at this organization is typical of many similar efforts in different areas – data quality needs to be assessed and improved before the data can physically be moved to the new platform. From [3], the areas of most interest for the organization traditionally have been presentation and application of data rather than collection and organization. The research that we have performed is in the area of finding new ways of attacking data quality assessment problems avoiding a manual approach wherever possible. Since a large part of the quality problem in many organizations can be at least assessed using tools that only examine data at the syntactic level, the possibilities for automating these processes are promising. There are a number of organizations that have free text data fields describing any number of phenomenon – customer contacts at a call center, free-form answers on survey data, manufacturing floor notes, and so on. The objectives of the research are to develop a framework for building repeatable processes that can extract usable, atomic data from free text fields.

2.4 What is New?

When addressing the issue of extracting data from text, the conceptual problem is able to be reduced to the question: "What patterns exist in the text, and what does the text tell us about what the values of certain data items should be?" Along the way, it is important to be able to assess the impact that the approach is having, and where there are spots of weakness or incompleteness. The new ideas in this research essentially encompass the framework that allows an information analyst to focus on the important parts of the problem that are specific to a given situation, rather than spending all of the time focusing on the "boilerplate" structure that must be built to support the core effort. Traditional approaches to dealing with free text fields have focused on specific solutions to specific data. By structuring the problem as described above, we can minimize future duplicated effort by gathering the similarities between efforts of this type, finding a central direction, and formalizing that direction and process flow.

Additionally, we put forward the use of a specific tool to aid in this process. Discreet finite state automata in the form of the regular expression language allow discovered text patterns to be formalized and codified in a way that is flexible. [2] Spelling mistakes in the text can be accounted for, as well as small variations in grammar and word ordering. Optional phrases or boundary identifiers can be used to point out critical areas in the text, and the relevant data extracted and potentially reformatted. [5] What is really needed in these situations is a formal language that allows an analyst to explain in detail the

patterns that they are seeing. These patterns must be communicate able to a machine which can then find instances of their occurrence within the source data – regular expressions are identified as the tool for the job as this description matches their purpose.

3. THE APPROACH

Given the circumstances, the problem dictated that we approach it in a semi-structured manner. Structured techniques are where the form of the problem guides the form of the solution. This solution was developed using a toolkit approach that iteratively broke the problem down into smaller sub-components, attacking each one individually – one of the classic techniques for managing complexity in projects. These components were then reassembled into a larger whole to produce a repeatable process suitable for later use on different datasets. This architectural approach attempts to maximize reusability while minimizing development time. The framework needed to be generalized so that the only required input for subsequent challenges would be information describing the specific differences of that situation. For example, the specific text patterns found in another data store would likely differ from the ones that were discovered in this particular data set, but the process and methodology would not.

The approach that was taken for this project is shown in the process flow diagram of Figure 2. This attempts to break the entire process down into various components, each of which are handled independently. The consolidated records represent the input data after some transformation to make manipulation easier. Throughout the process, statistics are captured about the data (input and result) to assess how effective the approach is. The iterative nature of the process is shown by the loops in the process flow diagram. The inside loop represents the work of finding patterns in the data, and codifying them as regular expressions in the program. The outer loop represents multiple runs of the program as it evolved, looping until a satisfactory percentage of the data has been verifiably correctly processed.

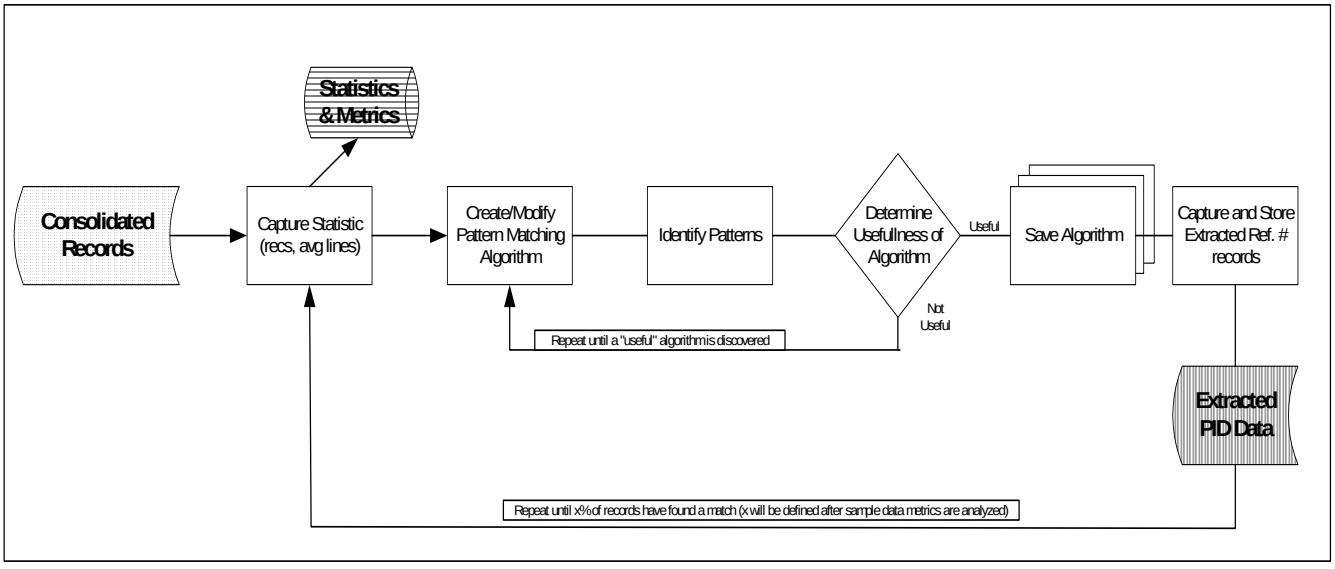

Figure 2: Process flow for iterative improvement of the extraction algorithm

The entirety of the process is automated by three components – the first prepares the source data, the second extracts the targeted data fields from the source data, and the third gathers statistics and information about the process. In Figure 2, the "Identify Patterns" and "Determine Usefuleness of Algorithm" are the only stages that require human intervention. Once the algorithm has been created, the

process runs entirely without intervention.

During the development of this approach, the outer loop was run a total of eighteen times, while capturing iteration improvement statistics along the way. These statistics are presented later, showing the gradual improvement of the algorithm over time. The effectiveness of the inner loop was the greatest at the beginning of the effort, and tapered off towards the end, as the patterns that were found in the data comprised fewer and fewer total records. The stop point on the outer loop is when a target percentage of records have been successfully matched. This number is informed by what patterns can be located in the data. When newly discovered patterns hold true only for a handful of records is the point of diminishing returns.

3.1 Implementation

The time required for implementation of this effort from start to finish can be seen in Table 1. This time includes planning, analysis, development, summary, and reporting.

Role	**Time Spent (Hours)**
Strategist	48
Implementation Director and Project Manager	300
Implementation and Technical Work	360
Administrative Support	80

Table 1: Roles and project time expenditures

The role of the Strategist was to guide the overall effort, give suggestions and input to the process, identify potential areas of improvement and to encourage them. The Implementation Director and Project Manager acted as an interface between the organization and the implementation team, gathered requirements, elicited feedback, and helped to develop the set of "stop measures". Since the project was an iterative process, the set of "stop measures" were measurements and milestones that were identified as indicative of project success. "Stop measures" included measurement of the average number of successful matches for a given pattern, and monitoring the total percentage of matched records. The number of successful matches for a pattern informed where the point of diminishing returns was as described above, and the total percentage of matched records identified the overall utility of the effort to the organization. The role of Implementation and Technical Work was tasked with turning the requirements, strategy, and specific knowledge about the problem into a viable, working piece of software that could be reused. Finally, the role of Administrative Support lent support to the effort, documented progress, and reported on successes and obstacles encountered along the way.

3.2 What are the new ideas?

The new idea surrounds testing the cost effectiveness of a parser driven architecture in this type of data environment. Given the flow of the data through systems, packaging the entire approach up into a repeatable process was not only an important idea, but it was absolutely necessary since it was known from the start that the entire dataset was being refreshed quarterly from an up-stream source.

The new technical ideas involved with this research are:

- The use of regular expressions based on discreet finite state automata against unstructured English text rather than structured text
- A method of pruning the total record set based on characteristics of the data to be extracted.
- The method of prioritizing those expressions in a way that maximizes their utility while minimizing their misclassification rate (false matches).

This "divide and conquer" approach looked at the data in three segments – the first segment contained records that had no extractable data at all. The second segment were the records that contained data to be extracted, and the third segment were undetermined records that must be looked at manually, but cannot be assumed to have no extractable data. The goal was to maximize the size of the second segment, minimize the size of the third segment, and minimize any possibilities for misclassification of data between segments.

3.3 What is the new technical Approach?

The new approach that this project brings to the table is the use of a prioritized matrix of regular expressions, each of which is associated with an extraction area – when a particular piece of context data is matched within the plain text, a subset of the matched data is identified as the extractable data and the relevant fields are populated. The technical approach also allows for iterative statistics gathering along the way that provides constant feedback about the flow of the data through the system, pointing out potential problems as they arise rather than at the end of the pipeline where their consequences are seen. The information gathering along the way enabled us to keep a close watch on progress and improve the process as it went along.

Typically, regular expressions in the past have been used only to identify data syntax; [2] that is to say, is a particular piece of data comprised of the right combination of characters, numbers, special marking, and so on. The ability of regular expressions to do "fuzzy matching" though allows analysts to cast the net a bit wider and develop patterns that take into account variations in spelling, wording, and in some cases even grammar. Since a regular expression is essentially nothing more than a formal language describing a particular discreet finite state automata, [6] regular expressions are capable of parsing human language text to the extent that a formalized step-oriented model can be developed for understanding language. The lessons learned from the development of so-called "artificially intelligent" chatter bots indicate that limited understanding of human language is possible through these means. [7]

The use of a prioritized matrix of regular expressions refers to ranking regular expressions according to two separate measures. The first measure is to rank the patterns based on how frequently they appear in the source data. The most frequent patterns tend to occur earlier, to minimize the time the program must spend searching for a match. The second measure is by probability of misclassifying data. For example, if one searches for the word "no", occasionally it will match the actual word "no". It will also incorrectly match "know" and "now". For this reason, the regular expressions were also ranked based on probability of misclassification of data. How likely a regular expression is to misclassify data depends on how much context the pattern includes. If a pattern is very specific in what it is looking for, the chances of misclassification are low, while if the pattern is very general, the chances are higher that it will misclassify some data. Using the example above, the pattern might have been made more selective and less prone to misclassification if the pattern had specified that the characters before and after "no" should not be alphanumeric characters. This would ensure that "no" was only detected as a freestanding word, (rather than a small piece of a larger word) or as part of a quotation, and so on. [2]

Identifying and Classifying Records With No Extractable Data

Another new aspect of the technical approach was work that was put into identifying which records had extractable data and which did not. While the majority of the records had information which could be extracted, some did not due to the fact that the text fields were malformed, completely empty, or fragmented. In these cases, it was important to develop a set of heuristics for deciding which records were not even worth considering. This saved both software time and human time - software was saved time since it could quickly identify whether or not a particular record should have patterns applied to it to extract data. Given that the program's accuracy rate was not 100%, a human effort was also inevitably needed to work on the records that did not match any patterns at all. Using a heuristic to detect which

records did not have any potentially extractable data saved humans the time it would take to identify all of those records by hand and reject them.

To identify records with no extractable data one must first eliminate all data from a text string that has no bearing on the values to be extracted. The goal of eliminating useless data from the string is to minimize the length of the string and the complexity that must be dealt with. This approach is essentially asserts that the useful context data is everything in the string that cannot be proven to be uninteresting. For example, if the goal is to extract the values "x" and "y" from the sentence "The values of x and y are 5 and 7", the words "The", "values", and "of" can safely be removed from the string without damaging the actual data – what is left is context and actual data. As human languages are often quite redundant in their construction, many words (such as prepositions, interjections, conjunctions, spacing, punctuation, and so on) can be eliminated from the text string without losing the data that the text is referring to. Additionally, due to the specifics of the data in this case, a number of additional constructs in the text field could be eliminated. For example, frequent references to the name of the data entry person were present that were later removed to minimize the length of the overall data. Given local conventions, this will likely be the case in many projects of this type.

After removing unnecessary data, the resulting string is compared with a well defined "threshold length". The threshold length is a minimum length for which strings can be considered to have data. For example, when attempting to extract a 10-digit identifier from a text string, the correct threshold is 10 characters, since one cannot extract a 10-digit item from less than 10 characters. This is a simple case – the threshold length for this research was a function of two variables. These variables are the sum minimum length of all data items to be extracted, and the sum minimum length of the needed context data. Adding the two variables yields an effective threshold length. The sum minimum length of all items to be extracted is straightforward. If two data items need to be extracted, whose minimum lengths are five alphanumeric characters and thirty characters, then the sum minimum length of the data items is 35. Finding the sum minimum length of the context data is a bit more involved. If the variables to be extracted are alphanumeric values, contextual information must be used to avoid having the word "hello" misclassified as a vendor identifier just because it is a 5 character alphanumeric as the identifiers are expected to be. After inspecting all candidate patterns for matching data, the sum minimum context data can be derived by adding the lengths of the smallest pieces of context data that identify surrounding data as an extractable field. Given patterns that accurately reflect the state of the data, this approach will provide a reasonable minimum threshold length that can be used to disqualify records from consideration.

The process described above has a few interesting characteristics. It will produce no false positives; in other words, given that minimum values are used for calculating the threshold, the threshold will be as small as possible and no record will be identified as having no extractable data when it in fact does have extractable data. The other side of this issue is that since the threshold was minimized, some records that do not have extractable data will still be considered candidates for extraction. These cases arise when the total length of the string is above the threshold value, but contains no useable data. This research was conducted using a common approach to problems of this type; false positives are considered more damaging than false negatives. Specifically, false positives mean that the process is missing the chance to extract valid data, while a false negative only implies additional effort to be sure of the status of a particular record, which would have had to be performed if no record culling was done in the first place.

Assessing this entire process, this record identification can significantly decrease the amount of work that is done both by programs and by humans. It is also safe and conservative, meaning that since no false positives are allowed, it will only eliminate records of no use – it will never incorrectly eliminate records that could have been used.

4. WHERE ARE WE NOW?

The effort described here has resulted in a number of interesting facts that have been discovered about the nature of the data, and effective methods of attacking the problem. Aside from aiding the task at hand, the analysis that went into the creation of the repeatable method described here ultimately shed a lot of light on the specific characteristics of this data set. The next step for the organization is to take a look at the processes that go into creating the free text and that feed numerous other systems to determine where exactly data quality problems are creeping in. It is important to address several different areas where data quality problems can occur (such as collection, organization, presentation, and application) in order to improve the data evolutional life cycle [3].

4.1 The Iterative Process

Figure 3 shows the iterative improvement in the process over time. With most iterations, the overall statistics on the effort improve, with the exception of a few iterations in which different extraction patterns were tested for their effectiveness. The total number of records was approximately 1,050,000 – at the end of the effort, roughly 92.6% of the records had been accounted for, either identified as having no extractable data (the "ignoreable" items) or records where data was actually extracted ("items matched"). Since each text field could possibly have more than one extractable set of data values, an average number of extracted items per record were also included as a measure. While the average number extracted was about 1.2, the majority of records contained one set of extractable data, while a minority had as many as 15 sets. As more patterns were identified and added to the process, the total number of unmatched items fell dramatically. On the first revision, 31.47% of records had no extractable data, but also could not be classified as having no data. As the process improved, that number fell to 7.46%.

	Total Records	1048323									
Rev#	Unmatched Items	Unmatched Items (% Change)	Unmatched Items (% Total)	Ignoreable NSNs	Ignoreable Items (% Total)	Items Matched	Avg Extracted Per Item	Items Matched (% Change)	Items Matched (% Total)	Items Extracted
1	329948	0.00%	31.47%	14034	1.34%	N/A	N/A	N/A	N/A	264703
2	222474	32.57%	21.22%	73069	6.97%	N/A	N/A	N/A	N/A	286675
3	216552	2.66%	20.66%	78520	7.49%	N/A	N/A	N/A	N/A	287196
4	340514	-57.24%	32.48%	125708	11.99%	582101	1.100022161	0.00%	55.53%	640324
5	300476	11.76%	28.66%	132882	12.68%	614965	1.106002781	5.65%	58.66%	680153
6	296628	1.28%	28.30%	136699	13.04%	614996	1.106015324	0.01%	58.66%	680195
7	258818	12.75%	24.69%	146944	14.02%	642561	1.121207169	4.48%	61.29%	720444
8	175758	32.09%	16.77%	230002	21.94%	642563	1.12120368	0.00%	61.29%	720444
9	118178	32.76%	11.27%	230002	21.94%	700143	1.118208709	8.96%	66.79%	782906
10	116813	1.16%	11.14%	230084	21.95%	701426	1.118331798	0.18%	66.91%	784427
11	114642	1.86%	10.94%	230084	21.95%	703597	1.120758048	0.31%	67.12%	788562
12	102233	10.82%	9.75%	236867	22.59%	709224	1.12156808	0.80%	67.65%	795443
13	93825	8.22%	8.95%	237113	22.62%	717385	1.121552583	1.15%	68.43%	804585
14	94542	-0.76%	9.02%	237113	22.62%	716668	1.114291415	-0.10%	68.36%	798577
15	94929	-0.41%	9.06%	237118	22.62%	716276	1.113928151	-0.05%	68.33%	797880
16	99890	-5.23%	9.53%	237128	22.62%	711305	1.11530075	-0.69%	67.85%	793319
17	99591	0.30%	9.50%	237128	22.62%	711604	1.115439205	0.04%	67.88%	793751
18	78213	21.47%	7.46%	237130	22.62%	732980	1.207281236	3.00%	69.92%	884913

Figure 3: Iterative improvements in the data extraction process

4.2 Extraction Results

The data that was extracted from the free text field was then compared to data in two other systems for the purposes of validation. The first system, which will be referred to as System 1, was the main system storing the master data file. System 2 contained a subset of System 1's data that was updated on a

quarterly basis as described earlier. Taking several data items that were extracted from the free text field, the object of the validation phase was to compare the generated results with actual data in the target systems to assess how often there was overlap.

	System 1		System 2		Both		Neither	
Data	Count	% of Total	Count	% of Total	Count	% of Total	Count	% of Total
NSN	592454	86.83%	674613	98.87%	587214	86.06%	2498	0.37%
CAGE	617233	90.46%	623327	91.35%	616327	90.32%	58118	8.52%
CPN	529645	77.62%	504611	73.95%	496359	72.74%	144454	21.17%
CPN/NSN	524617	76.88%	504314	73.91%	493952	72.39%	147372	21.60%
CPN/NSN Together	523413	76.71%	501377	73.48%	493145	72.27%	150706	22.09%
CAGE/NSN	550591	80.69%	616510	90.35%	545067	79.88%	60317	8.84%
CAGE/NSN Together	534815	78.38%	547295	80.21%	506734	74.26%	106975	15.68%
RNCC Found	529645	77.62%	504611	73.95%	496359	72.74%	144454	21.17%
RNVC Found	529645	77.62%	504611	73.95%	496359	72.74%	144454	21.17%
CPN/NSN/RNVC/RNCC Match					490936	71.95%		

Other Data	Count	% of Total
RNVC Match	494292	99.58%
RNCC Match	494067	99.54%
RNVC/RNCC Match	493711	99.47%

Total Results Records:	682351
Total System 1 Records:	973661
Total System 2 Records:	2172860

Figure 4: Binning results in the validation phase

In Figure 4, the results of the binning phase are displayed. Counts refer to how frequently the extracted data overlaps with a particular system. In the "Both" column, counts refer to how frequently the extracted data matches both systems simultaneously, and the "Neither" column refers to the cases when extracted data could not be matched in either system. NSNs refer to record identifiers taken from the free text records. The CAGE and PART numbers, (together referred to as "CPN") are the data items that were actually extracted.

RNCC and RNVC numbers were taken from a separate source and used as indicators of how internally consistent the system data is. These variables stand for "Reference Number Category Code" and "Reference Number Variation Code". Together, they typically identify various procurement characteristics of the item in question, such as if it is bought from one or many sources, and so on. The RNCC and RNVC fields were used as a control variable – the data was easily available and given the system structure, should have matched 100% between systems. As can be seen here, this was not the case. Overall, taking the extracted data into consideration, there was a 71.95% accuracy rate for this extraction effort. Part of the difference between a 100% success rate and the actual success rate can be attributed to the fact that System 1 and System 2 do not perfectly match – as evidenced by the statistic that RNCC and RNVC match between both systems 99.47% of the time. As discussed earlier, System 2 is refreshed on a quarterly basis from System 1's data, and independent changes are made to both over the quarter. These changes, not all of which are communicated and synchronized, can also be identified as a source of some of the mismatch.

4.3 Consequences of Success

Given the data from Systems 1 and 2 to compare against, it was possible for this effort to identify the success rate, and also to get an idea of the percentage of records for which no decision was possible. The accuracy rate is quantifiable, as well as the costs associated with inaccurate data. With regard to the manual effort of inspecting each record, the accuracy rate is generally considered to be between 98-99%, but it is more difficult to quantify the costs associated with those errors, since the errors cannot be

discovered but by re-checking the data, or experiencing the consequences of using the bad data.

The cost savings realized by the performance of this process are substantial. The manual effort at data cleansing generally accomplished the needed work on 5,000 records per person/month of work. By correctly extracting data from 69.92%, and further classifying more than 22.62% of records as not containing any extractable data, the number of records requiring human intervention was reduced to a quarter of its original number. 69.92% of records had extractable data, which had an accuracy rate of 71.95%, meaning that a total of (0.6992 * 0.7195 + 0.2262) * 100 = 72.93% of all data was dealt with programmatically. Based on 5,000 records per person/month and 1,050,000 total records, this amounts to (1,050,000 * 0.7293)/5,000 = 153 months of manual effort, or 12.75 person years. With a per-full time employee cost to the organization well exceeding $60,000 per year, the savings are enormous.

Data that is not addressable by automated and semi-automated means must ultimately be addressed manually. Manual intervention requires human inspection of each individual field, followed by manual extraction and cataloguing of that data. This is accomplished using whichever tool is most efficient for the analyst, and may come down to simple use of "copy" and "paste". Given that this is slow and tedious, one of goals of this work is to minimize the need for it in the first place. In this case, the manual intervention would apply to the records that were not correctly extracted or identified by the automated process. 69.92% of records had data extracted, and 22.62% of records could be proved to contain no data. The remaining portion, 7.46% of records, would require manual intervention. Selective manual intervention on the portion of extracted records that were incorrect would also be appropriate. Given an extraction rate of 69.92%, and an accuracy rate in that extraction of 71.95%, manual intervention would be needed on 28.05% of the extracted records, or an additional 28.05% * 69.92% = 19.61%. Adding 7.46% and 19.61% indicates that 27.07% of the data must be dealt with manually. These numbers apply only for this specific instance of the process being used. In other situations, the exact amount of manual intervention will be a function of the accuracy of the patterns that are used as input to the process.

4.4 Research Advances

The approach that was used for this project can be effectively reused on a number of different projects requiring the extraction of meaningful information from free-text fields. For example, call logs from customer centers contain important information about customer interaction, CRM, hand-entered data from shop and warehouse floors and so on is often useful for determining trends in an organization but must first be formalized in order to make automated processing of the data possible. The parsing techniques described in this article are also applicable to less structured formats such as constellations of documents related to corporate house holding research and efforts. [4] The particular items of interest and relationships between business units may be piece of information that can be gleaned from automated parsing of unstructured documents. This is not to say that these parsing techniques are applicable when what is really needed is semantic analysis of the data, but they are quite helpful for identifying specific quanta of data that act as further hints and tips in the right direction. For example, a document that contains a description of task hierarchies, various personnel assigned to those tasks and their relationship between one another might be an excellent candidate for the creation of an approach similar to the one described in this paper.

4.5 Research Theory

The parser driven approach used as a part of this research effort seems to be a viable approach to complimenting existing data quality approaches. The effect of the addition of this approach to the overall information quality toolset is that now a repeatable method exists for dealing with some aspects of data quality in unstructured text fields. In some situations those text fields come from human data entry clerks, while in others they come from catalog product descriptions, internal documents, emails, and so on. The area of information research has long had a powerful set of tools for dealing with data and its

quality implications for structured, organized data, but has traditionally been somewhat weaker in dealing with amorphous data such as that found in unstructured text. While data managers consistently move towards structured storage of data wherever possible, free text fields will likely exist as long as data must be collected on phenomenon for which no comprehensive questionnaire or form can be developed, such as customer interaction.

5. CONCLUSION

Overall, a repeatable process was created for extracting data out of free text fields. The process developed approached the data with a prioritized set of regular expressions, a pruning strategy, and tools needed to assess the effectiveness of the work. The result was substantial time and monetary savings over a complete manual approach. The process is reusable for different types of unstructured text, making it possible to extend this research to other related fields. The use of an automated process more than paid for its own development in savings and reduced the manual workload to less than one tenth of its original size.

Regular expressions have proved themselves an interesting tool for data quality in several different settings. This research provides further evidence of their utility in the area of unstructured text. Since known and inferred patterns can be written as regular expressions, they may also be used for testing hypotheses about data, (to determine the existence or non-existence of patterns) and to extract particular items depending on contextual clues.

6. REFERENCES

[1] Aiken, P. H., Ngwenyama, O. K., and Broome, L., *Reverse Engineering New Systems for Smooth Implementation*. IEEE Software. March/April 1999 pp. 36-43.

[2] Friedl, J., Mastering Regular Expressions, O'Reilly and Associates, 1997, pp 4-17, 139-177.

[3] Liu, L., & Chi, L., *Evolutional Data Quality: A Theory-Specific View* in Proceedings of the Seventh International Conference on Information Quality (ICIQ-02). 2002. Cambridge, MA.

[4] Madnick, S., Wang, R., & Zhang, W., *A Framework for Corporate Householding* in Proceedings of the Seventh International Conference on Information Quality (ICIQ-02). 2002. Cambridge, MA.

[5] Miclet, L., *Structural Methods in Pattern Recognition*. Springer-Verlag. 1986.

[6] Savchenko, S., *Practical Regular Expression Mining And Its Information Quality Applications.* Proceedings of the Seventh International Conference on Information Quality (ICIQ-02). 2002. Cambridge, MA.

[7] Weizenbaum, J., *Computer Power and Human Reason*. W.H. Freeman, 1976.

[8] Yoon, Y., Aiken, P., Guimaraes, T. *Managing Organizational Data Resources: Quality Dimensions* Information Resources Management Journal 13(3) July-September 2000 pp. 5-13.