Exceptional Data Quality Using Intelligent Matching and Retrieval

*Clint Bidlack and
Michael P. Wellman*

■ *Recent advances in enterprise web-based software have created a need for sophisticated yet user-friendly data-quality solutions. A new category of data-quality solutions that fill this need using intelligent matching and retrieval algorithms is discussed. Solutions are focused on customer and sales data and include real-time inexact search, batch processing, and data migration. Users are empowered to maintain higher quality data resulting in more efficient sales and marketing operations. Sales managers spend more time with customers and less time managing data.*

Several business and technology drivers are disrupting the world of enterprise software, and that in turn is driving the need for more effective data-quality solutions. These drivers include business acceptance of the software-as-a-service (SaaS) model, wide adoption of the web as a platform, collapse of enterprise application silos, aggregation of data from disparate internal and external sources, the agile mindset, and the economic conditions driving agility.

SaaS is a software deployment model in which the provider licenses applications for use as service on demand, most often accessed through a browser. In 2007 SaaS clearly gained momentum, with the sector having three billion dollars in revenue; by 2013 revenue could reach 50 percent of all application software revenues (Ernst and Dunham 2006). Customer benefits from SaaS deployments include much quicker and easier implementations, relatively painless upgrades, global access through the browser, lower total cost of ownership, and software vendors sharing more of the risk (Friar et al. 2007). Related to SaaS is another significant shift, the move from proprietary platforms to the web as a platform. Again the user benefits because of less vendor lock-in, open standards, well documented and broadly accepted standards, and long-term commitment by the software industry as a whole. These all contribute to higher quality solutions across the industry as well as accelerated innovation and more choice and flexible solutions available to the user.

Over the past couple of decades enterprise software solutions tended to result in data in silos, for example, a deployed accounting system that is unable to share data with a customer-relationship management system. The potential business value, or benefit, from removing data silos has driven companies to pursue such efforts to completion. Collapse of the data silo is related to the larger phenomenon of overall data aggregation on the web. A growing pool of online structured data, better tools, and, again, a large economic driver are all pushing organizations to aggregate and use data from a multitude of sources.

Finally, one of the most significant shifts in the software industry is the explicit transition toward agile software development. Agile development includes iterative software development methodologies in which both requirements and solutions evolve during the development of the software. Agile methodologies are in contrast to waterfall methodologies, which imply that requirements are well known before development starts (Larman and Basili 2003). More effective agile processes are important not only for software development, but for organizations of all types and sizes. The necessity to keep organizations aligned with opportunities and external competitive forces is forcing this reality. Being agile allows an organization to adapt more rapidly to external forces, which in turn increases the chances of survival.

For companies, the above trends are resulting in more effective use of enterprise software as well as more efficient business operations. Effectiveness is driving adoption across the business landscape, across industries, and from very small companies up to the Global 2000. Efficiency is driving application acceptance and usage within the company. This combination of effectiveness and efficiency, driving adoption and usage, is fueling enormous growth of structured business data.

Large volumes of structured business data require significant effort to maintain the quality of the data. For instance, with customer-relationship management (CRM) systems (deployed under SaaS or traditional software installations), ActivePrime and its partners have found that data quality has become the number one issue that limits return on investment. As the volume of data grows, the pain experienced from poor quality data grows more acute. Data quality has been an ongoing issue in the IT industry for the past 30 years, and it continues and is expanding as an issue, fueling the growth of the data-quality industry to one billion dollars in 2008. It is also estimated that companies are losing 6 percent of sales because of poor management of customer data (Experian QAS 2006).

Several competing definitions of data quality exist. The pragmatic definition is considered here; specifically, if data effectively and efficiently supports an organization's analysis, planning, and operations, then that data is considered of high quality. In addition, data cleansing is defined as manual or automated processes that are expected to increase the quality of data.

Existing Solutions

Past solutions to data-quality problems were driven in part by the economics of the institutions having the problem. Traditionally, the demand for data-quality solutions was driven by very large organizations, such as Global 2000 corporations. They had the resources to deploy complex software system for gathering data, and they were the first to notice and suffer from the inevitable data-quality problems resulting from this complexity. Accordingly, the approaches developed by the information technology researchers pioneering the area of data quality (Lee et al. 2006) tended to emphasize statistical data assessment, business process engineering, and comprehensive organizational data-assurance policies. Given their size, the early data-quality customers had the resources to adopt these labor-intensive and therefore expensive solutions. The traditional data-quality solutions tended to rely heavily on manual operations, in two different respects.

First, data was often hand-cleansed by contracting with external staffing agencies. Business analysts would first identify what type of data-quality work needed to be performed and on which data. Large data sets would be broken up into reasonable sizes and put into spreadsheets. This data would be distributed to individuals along with instructions for cleansing. After the manual work on a spreadsheet was finished oftentimes the work could be cross-checked by another worker and any discrepancies investigated. Once the data was finished it was reassembled into the appropriate format for loading back into the source IT system. Such manual effort has clear drawbacks, including the time required to cycle through the entire process, the possibility for manual error, and the need to export and then import the final results. Exporting is usually fairly easy. The importing is almost always the bigger issue. Import logic typically needs to identify not only which specific fields and records to update, but also how to deal with deleted or merged data, and how to accomplish this all without introducing new errors. Another issue is that additional work is required to build and thoroughly test the import tools. Finally, the entire manual process has little room for increased return on investment. The customer has to pay for the manual work each time data is cleansed, meaning that the economic benefits of automation are not realized.

Second, earlier data-quality vendors provided

technological solutions to data-quality problems, and these required significant manual setup. The reasons for the manual setup included business analysis to understand data-quality needs of the organization, identification of the final data-quality work flow, and then the actual programming and configuration to put the data-quality solution in place. In other words, these companies were building custom data-quality solutions using data-quality vendor application programming interfaces (APIs). Once the solution was put in place, automation reduced the manual effort, so a longer horizon for return on investment was acceptable. These solutions worked fine for large companies that could afford both the problem, initial enterprise system that aggregates the data, and the solutions. Today, sophisticated business applications are being used by even the smallest organizations; hence, the manual effort associated with data quality must be in alignment with the resources of these smaller organizations. The data-quality solutions must leverage automation and, at the same time, provide the business user with intuitive access to the processing results and the ability to override the results.

Data-quality research has also seen significant progress with the first issue of the new *ACM Journal of Data and Information Quality* published in 2009. Frameworks for researching data quality have been introduced (Madnick 2009, Wang 1995) as well as specific mathematical models for addressing the record linkage problem (Fellegi and Sunter 1969). Recent research in record linkage includes the development and deployment of more intelligent linkage algorithms (Moustakides and Verykios 2009, Winkler 2006). Data linkage is a core issue in many data-cleansing operations and is the process of identifying whether two separate records refer to the same entity. Linkage can be used for both identifying duplicate records in a database as well as identifying similar records across disparate data sets.

Solutions

ActivePrime's initial products and services focus on increasing the quality of data in CRM systems. CRM systems store people-centered data at the core with other types of data, like product, transactional, or other forms of data, considered secondary in nature. Though the secondary data is essential as well, in a CRM system the user interactions predominantly revolve around managing and querying the people-centered data. The main entities involved are organizations and people. Organizations are typically referred to as *accounts* and people as *contacts*. As is expected, accounts and contacts have properties such as postal addresses, email addresses, phone numbers, and

such. The main focus has been on the effective and efficient management of the quality of account and contact data.

Before we describe the various solutions, it is important to understand why the quality of data in CRM systems is so poor in the first place, and also how data enters the system. One of the main reasons for poor quality data in CRM systems is that the data is initially entered by hand into some system, sometimes by users without appreciation for the importance of data quality. For instance, the user may initially enter the data into an email client for which the user is the sole consumer of the data. For this dedicated one-party use, the quality of the data may not be so important. If the company name is misspelled, or if the person's title is abbreviated in a unique way, there really is no problem. The issues arise if that data is then synchronized with a large volume of data in a CRM system with many other users, and the organization is depending on the data for operational purposes. Suddenly the misspellings, abbreviations, and missing information have significant, material impact on the ability of the organization to leverage the data.

There are five points of entry for the majority of data entering CRM systems: manual entry of data by sales, marketing, and support personnel; batch importing of data from external sources of information; data synchronization from other software and devices like email clients, mobile phones, and PDAs; web entry of data by a potential customer or other third party; and migration of data from legacy systems.

Successful CRM data-quality initiatives must account for managing the quality of data at all these points of entry. ActivePrime's three solutions, CleanEnter, CleanCRM, and CleanMove, address each of these areas.

Data Quality upon Manual Entry

Upon manual entry of data, either through the web or using the CRM data-entry user interface (UI), CleanEnter provides search functionality that notifies users if they are about to enter a duplicate record. Figure 1 displays the search UI on contact entities and shows that entering a new contact of "ron raegan" has a potential match with four records in the system. At this point the user can drill down on a record, navigate to a record in the CRM system, or choose to continue entering "Ron Raegan" as a new record.

Data Quality upon Batch Entry

Customer data is often batch-entered after attendance at a trade show or after purchasing a list of customer data from a third party. Upon batch entry of data, or when cleansing large volumes of data that have just been synchronized into the

Figure 1. Results of Real-Time Inexact Matching.

CRM system, sales or marketing support staff will need to identify areas in the data that need to be standardized or find and merge duplicate data. Figure 2 shows the standardization screen in CleanCRM, with each pair of lines representing the record before and after standardization. For example, Rows 97 and 98 illustrate two fixes: the address convention standardized to short form—Dept instead of Department, and the country name normalized from America to USA. The user has control over which standardizations can be applied by selecting controls on a UI (not shown) that include turning all standardizations on or off and setting naming conventions for corporate names, business designations, addresses, secondary address formatting, city spelling corrections, and state names as short form or long form.

The second type of cleansing applied by CleanCRM is identification and merging of duplicate data. Duplicates can be identified based on exact or inexact matching on any field. Duplicates can then be merged together according to some predefined merge rule or a custom merge rule. Merge rules specify how records are rolled up, which record is the master record in the system, and which field data are selected in the case of conflicting data. Figure 3 shows the merge screen with two duplicate sets. Note that the first set contains two records, 8.1

and 8.2 with differences in account name and first name. Even given spelling errors and nicknames the duplication has been found.

A combination of domain knowledge and inexact matching allows for finding these records. The nickname ontology is deployed and in this instance finds that Becky and Rebecca refer to the same person. Note that if either Becky or Rebecca had been misspelled, CleanCRM still would have identified them as being duplicate because of inexact matching. The three duplicates records flagged as 7.1, 7.2, and 7.3 show how the robust inexact matching works, including the handling of non-ASCII character sets. The first names include Ývonne and Yvonne. In both duplicate sets a final record is shown that displays to the user what the final record will look like in the CRM system.

The grid in figure 3 also enables the user to edit the final record manually after standardization and merging. Any record that has been standardized shows up in the grid, highlighted accordingly, and the user can toggle the record to see the values before and after standardization. At this point any standardization values can be discarded. Because the software performed most of the difficult work of identifying duplicates and applying appropriate standardizations, the user is left with simply reviewing and overriding as necessary. If more specific standardizations or merging is needed, then plug-ins can be provided for an extra level of automation. Specific business logic can be applied resulting in a very streamlined and automated data-cleansing process with the user being able to adjust the results accordingly. A balance is provided between full automation and appropriate manual intervention.

Data Quality upon Migration

CleanMove is a data-quality solution for assisting the migration of customer data from a previous, legacy CRM system to a new system. It expedites the data-migration process by applying a series of data-cleansing rules to customer data that verify the quality, integrity, and format of the data. Rules include verifying unique data in database key fields; referential integrity checking; phone, address, and date formatting; finding and merging duplicates; and other processing. Some CRM systems, like Oracle CRM On Demand, put strict controls upon the format of data before it can be imported. If data like phone, addresses, and picklist values are not formatted properly, the database will reject the data. In other words, if the data is not clean then the customer cannot even import the data. When millions of records are being brought into such systems, the value provided by CleanMove is very significant. Several months of either manual work or custom coding can be saved by using CleanMove.

	CleanCRM.Contact Address 1	CleanCRM.Contact Address 2	CleanCRM.Contact City	.Contact Stat	CRM.Contact Zip/	CleanCRM.Contact Count
93						
94	333 West 7th St	Suite 202	Richmond	VA	23233	USA
95	333 West 7th St	Ste 202	Richmond	VA	23233	USA
96						
97	333 W 1st St	Department C	Oak Brook	IL	60523	America
98	333 W 1st St	Dept C	Oak Brook	IL	60523	USA
99						
100	333 W First St	Suite 307	Oak Brook	IL	60523	USA
101	333 W First St	Ste 307	Oak Brook	IL	60523	USA
102						
103	201 King of Prussia Rd					USA
104	201 King Of Prussia Rd					USA
105						
106	6007 Oak Court		JEFFERSONVLE	GA	31044-9753	USA
107	6007 Oak Ct		Jeffersonville	GA	31044-9753	USA
108						
109	6007 Oak Court		JEFFERSONVLE	GA	31044-9753	USA
110	6007 Oak Ct		Jeffersonville	GA	31044-9753	USA
111						
112	6007 Oak Court		JEFFERSONVLE	GA	31044-9753	USA
113	6007 Oak Ct		Jeffersonville	GA	31044-9753	USA

Figure 2. Results of Standardization of Address Data during Batch Processing.

Dup?	Master Loc	Filter	CleanCRM.Account Name	CleanCRM.Contact First Name	CleanCRM.Contact Last Name	CleanCR
Dup?	Master Loc	Filter	CleanCRM.Account Name	CleanCRM.Contact First Name	CleanCRM.Contact Last Name	CleanCR
▣ 5.1	▣ CSV	▣ allow	COMMCAST COACHING	▣ BECKY	▣ SHELTON	▣ 300 W
▣ 5.2	☐ CSV	▣ allow	Comcast Coaching Inc.	☐ Rebecca	☐ Shelton	☐ 300 W
Final			COMMCAST COACHING	BECKY	SHELTON	300 W Al
Dup?	Master Loc	Filter	CleanCRM.Account Name	CleanCRM.Contact First Name	CleanCRM.Contact Last Name	CleanCR
▣ 6.1	☐ CSV	▣ allow	Claus Productions	☐ BOB	☐ Beache	☐ 333 W
▣ 6.2	▣ CSV	▣ allow	Klaus Productions	▣ Robert	▣ Beach	▣ 333 W
Final			Klaus Productions	Robert	Beach	333 West
Dup?	Master Loc	Filter	CleanCRM.Account Name	CleanCRM.Contact First Name	CleanCRM.Contact Last Name	CleanCR
▣ 7.1	☐ CSV	▣ allow	Sports-Power	☐ Ývonne	Dwyer	☐ 333 W
▣ 7.2	▣ CSV	▣ allow	Sports Power	▣ Yvonne	Dwyer	▣ 333 W
▣ 7.3	☐ CSV	▣ allow	The Sports Power	☐ YVONNE	Dwyer	☐ 333 W
Final			Sports Power	Yvonne	Dwyer	333 W 1s
Dup?	Master Loc	Filter	CleanCRM.Account Name	CleanCRM.Contact First Name	CleanCRM.Contact Last Name	CleanCR
▣ 8.1	▣ CSV	▣ allow	Hughes Instruments	Steve	Jonson	110 Mass
▣ 8.2	☐ CSV	▣ allow	Húghes Instruments	Steve	Jonson	110 Mass
Final			Hughes Instruments	Steve	Jonson	110 Mass

Figure 3. Results of Duplicate Identification during Batch Processing.

A Data-Quality Platform

All three solutions are built on a platform for rapidly building data-quality applications: an integrated set of APIs, algorithms, 64-bit Linux servers, ontologies, specialized databases, and third-party data sets. For performance reasons low-level algorithms are coded in C or Cython; business logic, application logic, and ontologies are encoded in Python. User interfaces are coded in standards-based, cross-browser HTML and JavaScript, with data being transferred as XML, JSON, or YAML.

Intelligent Matching and Retrieval

ActivePrime leverages AI-related techniques in three broad categories: lightweight ontologies, search-space reduction (SSR), and query optimiza-

Articles

tion. The use of lightweight ontologies has been described elsewhere (Bidlack 2009). In summary, lightweight ontologies are deployed as modules and classes in the Python programming language, enabling rapid, iterative development of ontologies using a popular scripting language. The ontologies also benefit from the large repository of built-in Python operators. Sophisticated operations on ontologies can be performed with just a few lines of code.

SSR techniques are utilized when performing inexact matching on larger volumes of data, when record counts grow into the many thousands and millions. Query optimization techniques allow for real-time detection of duplicate records when matching one record to a large remote data base.

Search Space Reduction

Robust inexact matching algorithms, comparing the closeness of two strings, have been in circulation for at least four decades (Levenshtein 1965) and remain an active area of research (Navarro 2001, Chattaraj and Parida 2005). The challenge today is how to perform such matching on larger volumes of data, very quickly. Inexact matching is important in many data-cleansing operations including normalization, searching for duplicates, and performing record linkage (Dunn 1946). In such situations the computational cost of comparing each record in the large data set to every other record is prohibitive, especially given that each inexact string comparison operation itself is expensive, being $O(n\ m)$ to compare strings of length n and m. Thus, a brute-force comparison of all pairs among K strings, each of length n, would take $O(K^2\ n^2)$. Clearly this complexity is unacceptable in many data-cleansing operations where there are millions of records, that is, $K > 10^6$. In addition, if matching is being performed on z fields on each record, then the complexity is $O(K^2\ n^2\ z)$. Processing time with modern hardware would most likely be measured in years.

To provide solutions with acceptable computational complexity, four types of SSR techniques are leveraged. First, many data-quality operations involve matching strings, where matches are defined as those strings within some acceptable threshold of one another. Therefore, candidates outside the threshold can be pruned without computing an exact edit distance. The histogram-based pruning technique does this by quickly computing a bound on the edit distance between two strings. Second, search-based pruning uses the acceptable distance threshold to terminate the edit-distance computation as soon as the threshold is reached. Third, inexact indexing facilitates rapid inexact matching on large volumes of data. Finally, for conjunctive queries over multiple fields, choosing

an intelligent query-field ordering can produce dramatic performance improvements.

Histogram-Based Pruning

Histogram-based pruning is a technique to rapidly identify whether two strings are beyond an acceptable edit-distance limit. Pruning of inexact string comparisons can reduce run time, sometimes by well over 70 percent, by decreasing the number of exact edit-distance operations being performed. Exact edit-distance calculations are rather expensive (quadratic time), and histogram-based pruning cheaply identifies many instances where such operations are unnecessary. The histogram computation is itself on average much better than $O(n + m)$.

Histogram-based pruning works by computing a global histogram-based metric on the strings being compared. The metric provides a lower bound on the distance between two strings. If the metric is greater than T, the acceptable difference threshold between the two strings, then there's no need to perform the computationally expensive edit distance.

An alphabet histogram H is computed by making a one-time scan over the two strings being compared, $S1$ and $S2$. H contains N cells, where N is the number of letters in the alphabet B and B has all possible characters in the alphabet. For instance, $N = 36$ for an alphabet of case-insensitive, alphanumeric ASCII characters. For each character $C1$ in $S1$, increment $H[C1]$ by one, and for each character $C2$ in $S2$, decrement $H[C2]$ by one. If a character C appears equally often in both strings, then $H[C] = 0$. Define $H+$ as the sum of all positive numbers in H, with $H-$ the sum of all negative numbers in H. The magnitudes of $H+$ and $H-$ each provide a lower bound on the edit distance between the two strings. Thus if the absolute value of either $H+$ or $H-$ is greater than T, then the strings cannot be similar to one another.

Search-Based Pruning

The edit distance between two strings can be defined as the shortest path in a graph of strings, where edges represent edit operations. Standard edit-distance algorithms calculate the distance by finding the shortest path. Since we do not care about the exact distance if the match is unacceptable, we can often terminate the search early, once we have ruled out all paths of length T or less.

Note that just as T can be used to terminate edit distance computations, T can also be used to terminate histogramming. In many instances the algorithm can terminate early such that $S1$, $S2$, and H are never fully scanned. The computation is beyond the acceptable distance, hence the search for a solution is terminated. Finally, H is most often a sparse array, hence with proper accounting, while

computing $H+$ and $H-$. only nonzero elements in H need to be accessed, providing a further practical speedup of the algorithm.

Inexact Indexing

Indexing can enable speedups in processing of several orders of magnitude. Indexing is viewed as a way to reduce processing by grouping the records into a large number of buckets, and then any specific string comparison searches only over the data in a small number of relevant buckets, effectively pruning the search space. The creation of buckets is possible by mapping strings (keys) such that all potentially similar strings map to the same bucket. Then when performing a search, a string is matched against only the records in the appropriate buckets.

A main engineering challenge with inexact indexing is developing the appropriate mapping. The ActivePrime matchers use several different mappings. Two of the most common are length-based mapping and starts-with mapping.

Length-based mapping simply puts each string into a bucket based on its length. For example, if 400,000 records are to be indexed and the string lengths range from 3 to 43 characters, then on average there will be 10,000 strings per bucket. If T (the acceptable threshold) is on average four edits, then on average the number of buckets to search in will be four and the computations can be reduced by a factor of 10.

Starts-with mapping can potentially segment data into a very large number of buckets, depending on how many characters are used for mapping. All strings that start with the same N characters are put into the same buckets. Given a case-insensitive, alphanumeric ASCII alphabet, for $N = 2$, then the number of buckets is 1296 (36 squared). However, this mapping will result in missing some inexact matches: those with discrepancies within the first N characters. One way to significantly reduce incidence of such misses is to include an ends-with index and then search on both starts-with and ends-with. On average, for $N = 2$, the search space is reduced by two orders of magnitude.

Querying Field Ordering

A second way to leverage inexact indexing is to take advantage of matching on multiple fields where the user expects to conjoin (AND) the results together, which is the case for most data-quality applications. Data-quality applications almost always require matching on multiple fields, and search times vary greatly on different fields, depending on the data itself as well as the match threshold being applied to the field. Because of varying search times, the order of querying can be adjusted to take advantage of these differences by dynamically detecting the optimal field search order. This ordering effectively tiers the searching of indexes.

After indexes are created for each field, a small random selection of queries is applied to each field's index to rank the field indexes from fastest to slowest. Then all future queries are applied to the field indexes in the selected order. The edit-distance computation is being applied to a progressively smaller number of the total records (progressive reduction). Each field index value stores the unique record identifiers. Identifiers are the row numbers of the records in the original database. For instance, a database with 1000 records will have unique identifiers from 0 to 999. Progressive reduction is possible because matching is an AND operator (intersection of records). For instance, assume a database of 1000 records, and with two fields being queried, fields $F1$ and $F2$ with the $F1$ field index being faster and thus always being queried ahead of the $F2$ field index. When comparing strings $S1$ and $S2$ to $F1$ and $F2$, $S1$ is queried in $F1$, and the subset of records returned is $U1$ and is 1000 records or less, and most often much less than 1000. Then $S2$ only needs to be compared to the subset of records in $U1$ on the $F2$ field index. Therefore, if $U1$ reduces 1000 to 20, the total number of edit-distance computations is 1020, 1000 on the fastest field index $F1$, and 20 on the slower field index $F2$. Statistical evidence shows that with three fields and with the large variation in the field index performance, the reduction in processing can often be between one to two orders of magnitude.

Query Optimization

Query optimization is deployed to enable real-time matching of a record to a remote database. For instance, with CleanEnter, while entering a new record into the CRM system, the user first searches to identify whether the new record already exists. The user is comparing the one new record to the entire CRM database, searching for anything that looks similar.

The user's query is analyzed based on the context of the fields, like company name or state name, and appropriate domain knowledge is referenced for expanding the query. For instance, the state Massachusetts may have MA and Mass as synonyms and the query is expanded appropriately. Besides expansion through domain knowledge, query expansion occurs using phonetic rules as well as heuristics around transposition and removal of characters. The query optimizer effectively constructs a query that has a high probability of finding potential inexact matches while only retrieving a very small subset of the remote database. The subset of records is then analyzed using SSR techniques to compute actual inexact matches.
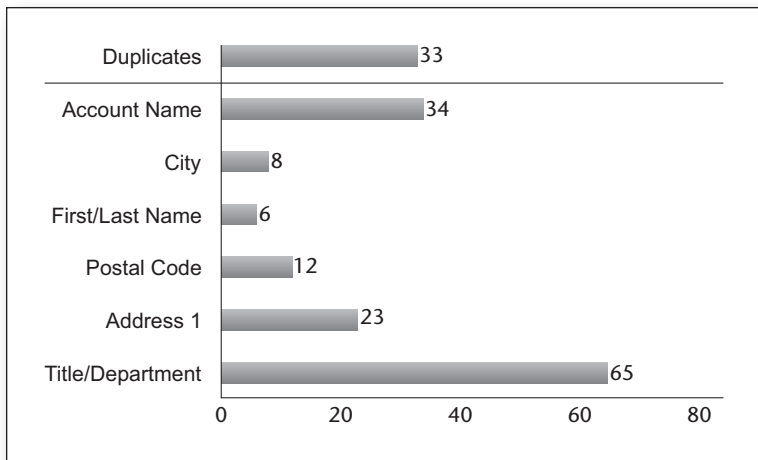
Figure 4.

Statistics showing the average percentage of records and fields that are updated during typical data-cleansing operations. Operations include identifying duplicates (the first entry) and field standardization (the bottom six entries).

Results and Impact

Over the past several years, experience shows that the number of duplicate records in corporate CRM systems range from the low end of 10 percent to a staggering 70 percent on the high end. In addition it is common for at least 50 percent of CRM records to have faulty data, for instance, misspelled customer names or address data.

More specifically, Figure 4 shows average results for the percentage of errors found in data sets from eight operational (in production) data sets. The x-axis represents the percentage of records that are updated during various data-cleansing operations. The operations are de-duplication and normalization of the specified field names. For instance, on average 8 percent of all data in the city fields was normalized in some fashion. The six fields in figure 4 are often candidates for data cleansing in a CRM system. There are however many more fields that get cleansed.

The Title/Dept field in figure 4 corresponds to fields in the CRM system that represent the title of the person and the associated department. In many instances standardization of data in these fields is important for customer messaging. There's a direct correlation between the quality of the normalization and the ability to customize the messaging for a specific person, and that is incredibly important and valuable from a marketing perspective. Context-aware messaging always produces better results. How a company communicates value to a chief technology officer, for instance, is much different from how communications are provided to a financial analyst. Such targeted messaging is possible only with high-quality data. The title or department of the person must be well defined (normalized) to perform precise messaging.

Using a combination of approaches, cleansing data upon initial migration and then ongoing cleansing in real-time and in batch processes, duplicates can be drastically reduced. Typical deployments reach below 1 percent duplicates and have a fix rate of up to 90 percent of faulty data.

Quantification of the business impact includes customers who have saved their sales representatives up to four hours per week. These savings are due to less time spent looking for the correct data and determining where the data should reside because there are fewer duplicates, as well as improvement in communication with customers due to higher-quality account and contact data. For large companies, the savings of four hours per week per sales representative results in very large financial savings, and more important, more efficient use of the sales force, which directly results in more revenue. The combination of savings and extra revenue can quickly reach into the hundreds of thousands and sometimes millions of dollars.

Another significant customer benefit is enhanced reporting and analytics. For large public companies accurate reporting is very important due to the increase in securities compliances such as those required by the Sarbanes-Oxley Act. Failing to properly report on metrics such as customer counts and revenues has potential financial penalties that can be rather severe.

Benefits to Data Mining

Data-mining uses have clearly expanded in reach the past decade, broadening out from the corporate business analytics system to now becoming a common component of data-intensive, web-based consumer and business systems (Segaran 2007). When performing data mining there is often a requirement to perform analysis on numeric data with a dependency on associated textual data. For instance, when deriving analytics or performing pattern analysis on sales information in the 50 USA states, a significant issue that arises is simply normalizing the state names so that a small set of queries can be generated to obtain and prepare the data for analysis. If the data contains many different and unknown means of representing the state names, then the queries themselves become very complex. Worse than that, the result of the queries and hence the results of the analysis become suspect.

Robust data-cleansing operations as described in this article can have a significant benefit in data-

mining applications. Not only can the analysis be more automated and hence operationally efficient, but the results themselves can be more readily accepted as valid results.

Conclusions

The described products connect to three of the leading CRM vendor solutions and are in use by more than 150 customers, in more than 40 countries, representing more than 30 different languages. Companies as large as Fidelity and Oracle and as small as two-person consulting firms use the products. Important lessons have been learned over the past few years regarding how to build high-performance data-quality solutions on distributed, SaaS systems.

Integration of domain knowledge can drastically improve the quality of results (Bidlack 2009). A key challenge with this integration is that the search space is expanded because now more matches need to be considered. The domain knowledge expands single terms out to multiple terms. For instance, Massachusetts may be expanded to include MA and Mass. Now there are three strings to consider when searching instead of just one. This search expansion adds one more reason for focusing on high-performance inexact matching algorithms.

AI techniques discussed in this article, various search space reduction techniques and query optimizations, drastically improve the performance of inexact matching of textual data on large volumes. With corporate databases for even midsize companies now growing into the millions of records, and the desire for better results as enabled by domain knowledge integration, high-performance matching is becoming critical to user adoption of any data-quality solution. SaaS-based systems and other industry dynamics are also changing user expectations. Users are expecting great results with less effort on their part. It is now clear that AI-based systems will continue to play an important role in matching the users' expectations on the data-quality front. This article outlined a few ways that intelligent algorithms have been leveraged, and it is

expected that future data-quality solutions will continue down this path.

Acknowledgments

References

Bidlack, C. 2009. Enabling Data Quality with Lightweight Ontologies. In *Proceedings of the Twenty-First Innovative Applications of Artificial Intelligence Conference*, 2–8. Menlo Park, CA: AAAI Press.

Chattaraj, A., and Parida, L. 2005. An Inexact-Suffix-Tree-Based Algorithm for Detecting Extensible Patterns. *Theoretical Computer Science* 335(1): 3–14

Dunn, H. L. 1946. Record Linkage. *American Journal of Public Health* 36(12): 1412–1416.

Ernst, T., and Dunham, G. 2006. Software-as-a-Service—Opening Eyes in '07; Half the Market in '13 (FITT Research). Frankfurt, Germany: Deutsche Bank AG.

Experian QAS. 2006. U.S. Business Losing Revenue through Poorly Managed Customer Data (Experian QAS Briefing 3/28). Boston, MA: Experian QAS.

Fellegi, I., and Sunter, A. 1969. A Theory for Record Linkage. *Journal of the American Statistical Association* 64(328): 1183–1210.

Friar, S.; Zorovic, S.; Grieb, F; and Isenstein, M. 2007. *Getting SaaS Savvy—Successful Investing in On Demand.* New York: Goldman, Sachs & Company.

Larman, C., and Basili, V. 2003. Iterative and Incremental Development: A Brief History. *IEEE Computer* 36(6): 47–56.

Lee, Y. W.; Pipino, L. L.; Funk, J. D.; and Wang, R. Y. 2006. *Journey to Data Quality.* Cambridge, MA: MIT Press.

Levenshtein, V. I. 1965. Binary Codes Capable of Correcting Spurious Insertions and Deletions of Ones. *Problems of Information Transmission* 1(1): 8–17.

Madnick, S. 2009. Overview and Framework for Data and Information Quality Research. A*CM Journal of Data and Information Quality Research* 1(1).

Moustakides, G. V., and Verykios, V. S. 2009. Optimal Stopping: A Record-Linkage

Approach. *ACM Journal of Data and Information Quality* 1(2).

Navarro, G. 2001. A Guided Tour to Approximate String Matching. *ACM Computing Surveys* 33(1): 31–88.

Segaran, T. 2007. *Programming Collective Intelligence.* Sebastopol, CA: O'Reilly Media, Inc.

Wang, R. Y.; Storey, V. C.; and Firth, C. P. 1995. A Framework for Analysis of Data Quality Research. *IEEE Transactions on Knowledge and Data Engineering* 7(4): 623–640.

Winkler, W. 2006. *Data Quality: Automated Edit/Imputation and Record Linkage* (U.S. Census Bureau, Research Report Series, Statistics 2006–7). Washington, DC: U.S. Government Printing Office.

Clint Bidlack is president, chief technical officer, and cofounder of ActivePrime. Prior to ActivePrime, he was the founding chief executive officer of Viveca, a venture-capital-backed company that built innovative electronic catalog software. Viveca was sold in 2001. Bidlack has been building data-intensive software systems for almost 20 years, first in the automotive industry, then in research roles at the University of Tennessee in conjunction with Oak Ridge National Laboratory, and then at the University of Michigan. In 1992 Bidlack was part of the team that won the first AAAI robot competition.

Michael P. Wellman is a professor of computer science and engineering at the University of Michigan and a technical adviser to ActivePrime. He received a Ph.D. from the Massachusetts Institute of Technology in 1988 for his work in qualitative probabilistic reasoning and decision-theoretic planning. From 1988 to 1992, Wellman conducted research in these areas at the U.S. Air Force's Wright Laboratory. At the University of Michigan, his research has focused on computational market mechanisms for distributed decision making and electronic commerce. As chief market technologist for TradingDynamics, Inc. (now part of Ariba), he designed configurable auction technology for dynamic business-to-business commerce. Wellman previously served as chair of the ACM Special Interest Group on Electronic Commerce (SIGecom) and as executive editor of the *Journal of Artificial Intelligence Research*. He is a Fellow of the Association for the Advancement of Artificial Intelligence and the Association for Computing Machinery.

WORKSHOPS AT THE TWENTY-FOURTH AAAI CONFERENCE ON ARTIFICIAL INTELLIGENCE (AAAI-10)

Sponsored by the Association for the Advancement of Artificial Intelligence

AAAI is pleased to present the AAAI-10 Workshop program. Workshops will be held Sunday and Monday, July 11–12, 2010 at the Westin Peachtree Plaza in Atlanta, Georgia. The AAAI-10 workshop program includes 13 workshops covering a wide range of topics in artificial intelligence. Each workshop is limited to approximately 25 to 65 participants, and participation is usually by invitation from the workshop organizers. However, most workshops also allow general registration by other interested individuals. There is a separate fee for attendance at a workshop, and is discounted for AAAI-10 technical registrants. Registration information will be mailed directly to all invited participants. All workshop participants must preregister, and must indicate which workshop(s) they will be attending. Workshop reports are included in the workshop registration fee, and will be distributed onsite during the workshop. In most cases, reports will also be available after the conference as part of the AAAI Press technical report series.

THE WORKSHOPS

W1 – AI and Fun
W2 – Bridging the Gap between Task and Motion Planning
W3 – Collaboratively-Built Knowledge Sources and Artificial Intelligence
W4 – Goal-Directed Autonomy
W5 – Intelligent Security
W6 – Interactive Decision Theory and Game Theory
W7 – Metacognition for Robust Social Systems
W8 – Model Checking and Artificial Intelligence
W9 – Neural-Symbolic Learning and Reasoning
W10 – PAIR: Plan, Activity, and Intent Recognition 2010
W11 – StarAI — Statistical Relational AI
W12 – Visual Representations and Reasoning
W13 – Workshop on Abstraction, Reformulation, and Approximation

DEADLINES

March 29: Submissions due (unless noted otherwise)
April 15: Notification of acceptance
May 11: Camera-ready copy due
July 11–12: AAAI-10 Workshop Program
AAAI Formatting Guidelines: www.aaai.org/Publications/Author/author.php

SUBMISSION AND FORMAT REQUIREMENTS

Submission requirements vary for each workshop, but most key deadlines are uniform, unless otherwise noted. Submissions are due to the organizers on March 29, 2010, except where noted. Workshop organizers will notify submitters of acceptance by April 15, 2010. Accepted camera-ready copy is due on May 4, 2010. Please mail your submissions directly to the chair of the individual workshop according to their directions. Do not mail submissions to AAAI. For further information about a workshop, please contact the chair of that workshop. AAAI two-column format is often required for workshop submissions, and is always required for all final accepted submissions. Links to styles, macros, and guidelines for this format are located in the publications area of the AAAI website

AAAI Workshop Cochairs

Michael Beetz and Giuseppe De Giacomo

www.aaai.org/Workshops/ws10.php