Building Software that Captures Better Data

Danny Casale
Dan Elbaum
Pacific Northwest Software Quality Conference 2014
The purpose of this presentation is to introduce some of the decisions that should be made about what data to collect, how to store that data, and how to manage changes to the data. Some of these decisions will be familiar to you, but in my experience many programmers and business analysts are only aware of a couple of these decisions.

For this presentation, we will walk through these three topics using examples from projects we have been on. We tried to use examples that are relevant in many industries, not just utilities and transportation logistics.

We will also describe some substantial shifts in technology that have increased the potential value of the data our applications collect.
In the past, one of the top priorities of all systems was to limit the amount of data that was stored by the application. This made sense because storage was extremely expensive, disk capacities were limited, and database performance was a primary concern.

With improvements to storage, processing power, and analytic tools, we need to rethink our minimalist data strategy.
Disk storage has made incremental improvements over the last 10 years with faster interfaces, better caching, lower read/write times and higher capacities. Databases have also improved their storage technologies with distributed file systems, in-memory databases, hybrid databases and database engine optimizations.

Telecommunications equipment has also continued to improve to support the movement of large volumes of data.

While all these improvements were integral, the cost of storage is the pivotal change in the market that is causing the explosion of data analytics.

This is a graph of hard drive cost/GB over the past 35 years. While this looks like a nice linear trend, the scale is logarithmic.
Cloud computing has allowed companies to only pay for processing power when they need it.

This flexible pricing model can allow companies to answer very challenging questions using terabytes of data without having to invest in large infrastructure projects.

Cloud computing and the ability to engage 100’s of processors for JIT analysis is a key enabler of near real-time analytics.
With technology being capable of storing and processing large quantities of data at a relatively low cost, we have seen substantial investment in analytic tools.

The performance increase has created a market for Visualization tools that enable analysts and business users to quickly explore large volumes of data from disparate sources to look for anomalies, trends or to gain better insights into their organization and its activities.

Business Intelligence tools have matured to a point where end-users can sometimes pull their own data (i.e., self-serve). BI tools have also begun to incorporate more advanced analytical capabilities like linear and logistic regression, clustering, classification, association rules, decision trees, etc.

Predictive analytics describe a set of statistical techniques that predict the likelihood that something will happen. Some of these techniques require that someone build and refresh the models manually while other techniques, often referred to as machine learning, have the capability to learn and improve themselves from new data.

One of the issues we will discuss next is that business analysts and SMEs are often
unaware of the capabilities of predictive analytics and machine learning and fail to collect the data required to support those techniques.
Historically, software applications focused almost exclusively on the functional requirements of the system. Data requirements were often reduced to legal requirements, regulatory requirements, and data needed for basic reporting.

In our experience, data requirements are often incomplete due to a lack of understanding from the SMEs and business analysts of what questions can be answered with certain data.

Ideally, during requirements elicitation, the business analyst would consider the data the application handles and the different types of questions the data could be used to answer. The analyst could then work with the applications stakeholders and SMEs to determine if and how valuable answering those questions would be to the business.

We will see some real-world examples of the types of questions that data can answer when it is captured, stored and managed appropriately.
1. Identifying What Data to Collect

- From Wikipedia...
  - In computing, an event is an action or occurrence detected by the program that may be handled by the program.

- Who
- What
- When
- Where
- Why
- How

*(Corr and Stagnitto define the 7Ws analysis, we only discuss six.)*

From Wikipedia...
In computing, an event is an action or occurrence detected by the program that may be handled by the program.

The reason I am providing this definition is that the remainder of the presentation will focus on data produced by events.

What are all the dimensions of an event?

In the book Agile Data Warehouse Design, Corr and Stagnitto leverage the 7Ws Who, What, When, Where, Why, How, and How Many as part of a process to elicit Business Intelligence requirements using an Agile methodology.

We borrow from them but don’t use the How Many question because it is slightly more complex and not always relevant.
Businesses sometimes send “alerts” to notify customers of upcoming bills, mobile data usage, electricity consumption, etc. The goal of alerts is to help customers manage their usage of a product.

We have seen an implementation of alerts that collected insufficient data for analysis, and precluded continuous improvement of the system. The system stored a user’s enrollment status in alerts (subscribed/unsubscribed), but did not maintain a log of all alerts (text and email) that had been sent. It simply kept a monthly count of how many alerts were sent out to the entire customer population, how many customers subscribed during the month, and how many cancelled their subscription during the month.

This implementation was problematic because a large volume of customers were unsubscribing from alerts, and the business didn’t have data to help them decide what approach they could adopt in order to reduce the attrition rate. They had some hunches that the alerts may’ve been delivered too frequently, or at not the best times, but they didn’t have any data to verify those hypotheses.
Example: Alerts

<table>
<thead>
<tr>
<th>Who</th>
<th>Which customer received the alert?</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td>Which alert did the customer receive?</td>
</tr>
<tr>
<td>When</td>
<td>At what date and local time did the customer receive the alert?</td>
</tr>
<tr>
<td>Where</td>
<td>Where was the customer when they received the alert?</td>
</tr>
<tr>
<td>Why</td>
<td>Why was the alert triggered?</td>
</tr>
<tr>
<td>How</td>
<td>Which channel/device did the customer receive the alert through/on? (e.g. text, email, phone call, etc.)</td>
</tr>
</tbody>
</table>

Had we known who was unsubscribing, we could have profiled those customers and looked at their attributes to identify any particular customer segments that are unhappy with the product.

Had we known which alerts went to whom when, we could see if a particular alert was angering customers and causing them to unenroll.
2. Determining How to Store Data

- Resolution – the granularity of measurement at which a value is stored
- Unstructured Data – free form text

Resolution is the granularity of measurement at which a value is stored. Sometimes, categorically new insights can be extracted from granular measurements which are not feasible to obtain from coarse measurements.

This section reviews the importance of collecting data in a structured format rather than through free-form text input fields. When collecting data from users, it can be tempting to take the easy path and allow free-text input instead of doing additional work to figure out a set of pre-determined options for users to select from. While free-from text input fields may be less restrictive, and easier to technically implement, they create data that is more difficult to aggregate and analyze in bulk. The better option from an analytics standpoint is to anticipate what type of feedback to expect from users, and then make users select from one of many exclusive choices so that their feedback can be aggregated and analyzed more easily.
Companies often engage with their customers through a variety of channels such as phone, Interactive Voice Response (IVR), web, storefronts, mobile, etc. Certain channels which do not require human labor to operate are classified as “self-serve”, and cost substantially less to maintain than other channels. The cost of self-serve channels such as web and automated phone systems typically cost on the order of a few cents per customer engagement, whereas phone or storefronts can range in cost from $10-30 per customer engagement (Voxeo, 2013). In cross-channel analytics, organizations look at the channel that each customer starts in, and tracks the customer to see if they fall out of the self-service channel at some point, and into a higher-cost channel.

One mistake that can prohibit cross-channel analytics is storing the timing of customer interactions at too coarse of a resolution. When timestamps are captured at day-level resolution rather than second-level resolution, it’s not possible to determine the sequence in which same-day events occurred. So suppose we know that a customer has engaged the company both through phone and web on a single day to find information – the company would have no way of knowing whether the customer began their interaction with a call, and then visited the website, or vice-versa.

**Resolution Example: Cross-Channel Analytics**

- The problem: Customers were using high cost channels like storefronts and talking on the phone with CSRs instead of low-cost channels like Web and IVR (Interactive Voice Response).

- The data we had available to solve the problem:
  - Which customer did what on which channel on a particular date

- The data we needed to solve the problem:
  - Which customer did what on which channel at a particular date and time
### Resolution Example: Cross-Channel Analytics

<table>
<thead>
<tr>
<th>Who</th>
<th>Who triggered the event?</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td>What did the customer click? What button did he/she press?</td>
</tr>
<tr>
<td>When</td>
<td>When did the customer trigger the event? This should be a date/time to enable event-sequencing across channels.</td>
</tr>
<tr>
<td>Where</td>
<td>Which channel was the customer in? Do we want to record the IP address for geospatial analytics?</td>
</tr>
<tr>
<td>Why</td>
<td>Why was the customer using the channel? What was their goal?</td>
</tr>
<tr>
<td>How</td>
<td>Which device and operating system/browser did they use?</td>
</tr>
</tbody>
</table>

**Who:** Who triggered the event?

**What:** What did the customer click? What button did he/she press?

**When:** When did the customer trigger the event? This should be a date/time to enable event-sequencing across channels.

**Where:** Where was the customer engaging us from? Do we want to record the IP address for geospatial analytics?

**Why:** Why was the customer using the channel? What was their goal?

**How:** Which device and operating system/browser did they use?
Consider the data that companies attempt to collect when people unsubscribe from their emailing lists. All promotional emails are legally obligated to offer an option to unsubscribe under the CAN-SPAM act (FindLaw, 2014). Often, companies use an unsubscription confirmation page to collect customer feedback and learn how to improve their communications. Companies can collect this feedback with a free-text form, a list of mutually exclusive reasons, a list of non-mutually exclusive reasons and others offer a list of reasons and a free-text form.

One common error we’ve seen on unsubscribe pages is when companies ask for free-text commentary about the reason people have unsubscribed.

It is advantageous to offer a pre-determined list of reasons that map to actions that the business can take to rectify the problem the user identifies. For example, if the user says that the email is too frequent, the business can adjust the frequency or allow the user to select the desired frequency of emails. If many users are providing feedback to the effect that content is not relevant, perhaps the business should consider offering segment-specific messaging or more targeted offerings. By offering a list of issues, businesses can help users to identify correctable issues. On the other hand, offering a free-text field will help capture issues the business did not think of, but that also may
not be actionable.

As a note, we recognize that the availability of natural language processing software is increasing, but most companies do not have the software and required skills to easily perform this type of semi-automated text analysis.
3. Handling Changes to Stored Data

- From Wikipedia...
  - In computing, an event is an action or occurrence detected by the program that may be handled by the program.

- Change Data Capture
  - Similar to auditing features

From my favorite website Wikipedia:
In computing, an event is an action or occurrence detected by the program that may be handled by the program.

For this presentation, I would add to this definition (although it may be inferred) that an event happens at a point in time.

Change Data Capture is a technique to capture event information and manage changes to data modified by an event.
Financials were rolled up using the current relationship structure, not the structure that existed when the transaction took place.

Example: Company Relationships

• The problem: Rollup reports could not be produced accurately for historical data because prior company relationships were not maintained.

• The data we had available to solve the problem:
  • Current business to business relationship information

• The data we needed to solve the problem:
  • Effective date, expiry date, and relationship type for all company relationships past and present.
## Example: Company Relationships

<table>
<thead>
<tr>
<th>Who</th>
<th>Which companies participated in the relationship?</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td>What type of relationship was created between the companies?</td>
</tr>
<tr>
<td>When</td>
<td>When did the relationship begin and end? Do we need just date or date and time. Do we need to allow for back and future dating for accuracy reasons?</td>
</tr>
<tr>
<td>Where</td>
<td>Where is the relationship valid or where did the relationship event occur. Perhaps the relationship is only valid in certain countries or regions.</td>
</tr>
<tr>
<td>Why</td>
<td>Why was the relationship formed?</td>
</tr>
<tr>
<td>How</td>
<td>How was the relationship created? For example, was it through a merger or acquisition?</td>
</tr>
</tbody>
</table>

Who: Which companies participated in the relationship?
What: What type of relationship was created between the companies?
When: When did the relationship begin and end? Do we need just date or date and time. Do we need to allow for back and future dating for accuracy reasons?
Where: Where is the relationship valid or where did the relationship event occur. Perhaps the relationship is only valid in certain countries or regions.
Why: Why was the relationship formed?
How: How was the relationship created? For example, was it through a merger or acquisition?
Conclusions

- Leverage the data your application handles by making conscious decisions around:

  1. What data to collect
  2. How to store it
  3. How to manage changes to it

In conclusion, more powerful analytics tools and decreased storage costs have increased the value of your data.

Leverage the data your application handles by making intentional decisions around:

What Data to collect, how to store it, and how to manage changes to it.

Questions?