Building Software that Captures Better Data

Danny Casale

Dan Elbaum

Pacific Northwest Software Quality Conference 2014

Todays Focus: Data Requirements

- My goal: shed light on how to make decisions about:

    1. What data to collect
    2. How to store the data you collect
    3. How to manage changes to the data your application collects

- My approach: walk through scenarios we've encountered in current and prior roles on software development projects in order to point out lessons learned.

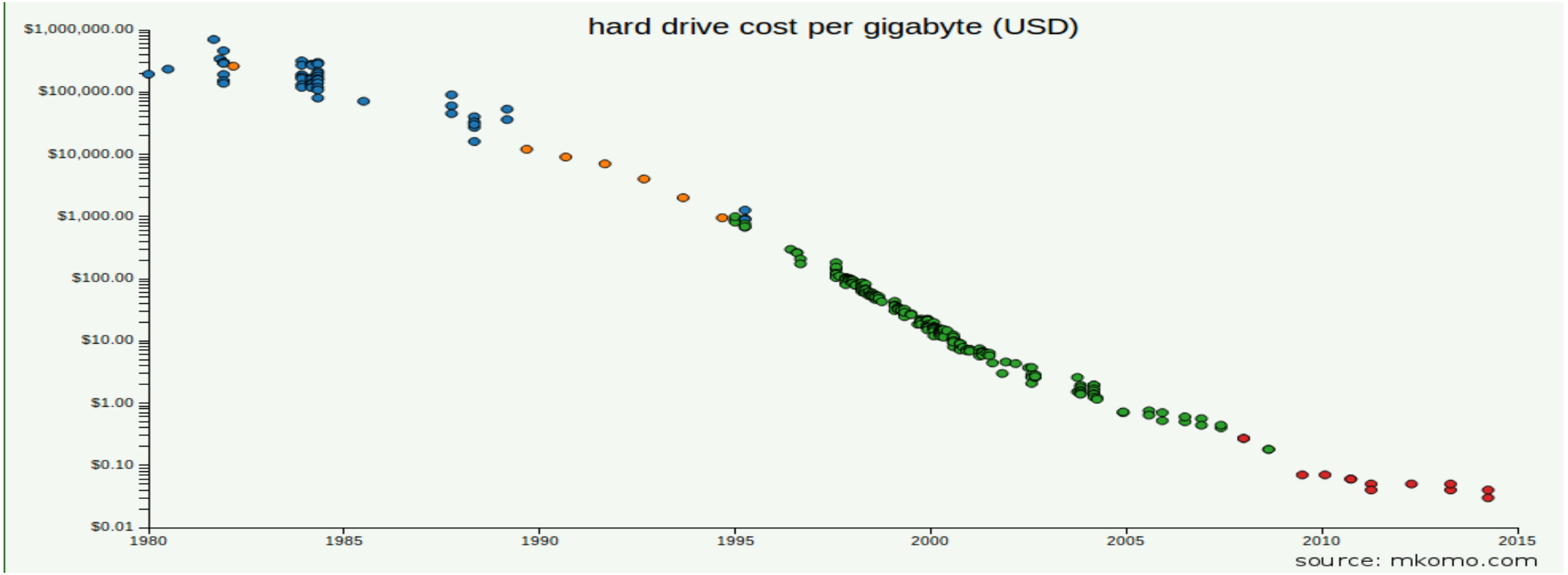
Motivation

Data Requirements are growing in importance because:

- Storage is now fast, high-capacity, and low-cost
- Processing power is growing with highly distributed cloud computing
- Analytic tools and techniques are maturing (e.g. predictive analytics, machine learning)

Cost of Storage is Declining

- Network Throughput, Disk Performance, Database Storage Models
- Most importantly, **COST.**



Processing Power and Cloud Computing

- Pay for what they use, no large infrastructure cost
- Near real-time analytics
- Analytics jobs can be optimized to run when and where power is cheap

Analytics Tools

- Visualization Tools
- Business Intelligence and Self-Serve
- Predictive Analytics
- Machine Learning

Challenges in Collecting Data Requirements

- Requirements elicitation has historically focused on delivering user-facing functionality, not collecting data from users. Analysts don’t know what questions to ask.

- Data requirements are difficult to elicit because business stakeholders lack the data-savvy to understand available options.

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.)

Example: Alerts

- The problem:
  - Many customers were unsubscribing from alerts and no one could figure out why.
- The data we had available to solve the problem:
  - How many customers subscribed / month
  - How many customers unsubscribed / month
  - How many alerts were sent / month
- The data we needed to solve the problem:
  - What alerts were sent, when, to which customers.

Example: Alerts

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

2. Determining How to Store Data

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

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

Who	Who triggered the event?
What	What did the customer click? What button did they press?
When	When did the customer trigger the event? This should be a date/time to enable event-sequencing across channels.
Where	Which channel was the customer in? 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?

Unstructured Data Example: Unsubscribe

- The problem: Customers were unsubscribing from email. Customers provided feedback about why they unsubscribed in a free-text entry field, so someone would have to manually read each customers response.

- The data we had available to solve the problem:
  - Free-form text description from customer.

- The data we needed to solve the problem:
  - A field that captured the high-level reason that the customer unsubscribed that could be analyzed and acted upon.

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

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

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