Big Data Project Failures

Ch.12: J. J. Berman

February 6, 2018

Big Data Resources- Complex

- They are difficult to build and easy to break.
    - inappropriate selection and use of human resources
    - incorrect funding
    - legal snags
    - bad data
    - poor data security

Two general categories

- Failures due to poor design and operation flaws in Big Data resources and

- Failures due to improper analysis and interpretation of results.

Failure is Common

- database managers are not trained to deal with
    - the layers of complexity found in Big Data resources.
    - fundamental concepts discussed
        - identifier systems,
        - introspection,
        - metadata annotation,
        - immutability,
        - and data triples.

Failure rate- Unknown

- Difficult to determine the failure rate
  - Organizations herald their triumphs but hide their misadventures.
  - There is no registry of Big Data projects that can be followed to determine which projects fail over time.
  - no formal designation “Big Data project”
  - no definition for failure, as applied to Big Data.
- failure is positively correlated with the size and cost of the projects.
- Big Data projects are characterized by large size, high complexity, and novel technology

Failed Standards

- Most standards fail.
- Simply because a standard has been developed by experts, backed by the U.S. government, and approved by an international standards organization, there is no guarantee that it will be accepted by its intended users.
- The most successful standards are specifications that achieved popularity before they achieved the status of “standard.” The best of these filled a need, enjoyed broad use, had few or no barriers to implementation (e.g., free and easy to use), and had the bugs ironed out (i.e., did not require excessive modifications and version updates).
    - The PL/I standard, developed by IBM, was soon replaced by Fortran and COBOL.
    - The Algol 68 standard was replaced by Pascal.
    - Ada, promoted by the U.S. Department of Defense, was replaced by C. The OS/2 operating system produced by IBM was replaced by Windows.
- Instability of standards is always bad news for Big Data Resources

EXAMPLE

- Pathology Reports
- “The patient has an scc, and we see invasion to the subcutaneous tissue, all the way to the deep margins, but the lateral margins are clear.”
- The complex sentence could be rewritten as six declarative statements:
    - Diagnosis: squamous cell carcinoma.
    - Invasion is present.
    - Invasion extends to subcutaneous tissue.
    - Margin is involved.
    - Tumor extends to deep margin.
    - Tumor does not extend to lateral margins.

Even Better Approach

- expressing every statement as a triple consisting of an identifier, a metadata term, and a data value.
- For example, consider the following triples:
  - 2847302084 weight “25 pounds”
  - 2847302084 instance_of 8909851274
  - 8909851274 class_name “dog”
  - 8909851274 subclass_of 7590293847
  - 7590293847 class_name “canine”
- We can use triples to express any information we might ever collect, and
- we can relate every triple to other classes of triples, to produce a data model.
- The triples we collect can be converted into more complex ontology languages (such as RDF, OWL, or DAML/OIL) because all of our data objects are well specified.

Two general principles

- Data objects can be well specified, without a standard. You do not need to store your data in a format that is prescribed by a standard.

- Data standards are fungible. If you know the rules for standards, you can write a program that converts to the standard, as needed.

Complexity

- In complex systems, they can be very difficult to detect.

e.g. Toyota Lexus ES 350- Unintended Vehicle acceleration

  - 50 million
  - 9 million recalls
  - due to sticky pedals, driver error, or improperly placed floor mats;
  - most vexing problems in software engineering involve “sometimes” errors;

Big Data project should be designed for simplicity

- “Can we achieve this functionality with less complexity?”
- “Do we need this level of functionality?
- Might we achieve a reduced but adequate level of functionality with a less complex system?”
- Design team must analyze the consequences of the increased complexity.

When Does Redundancy Fail ?

- With redundancy, when one server fails, another takes up the slack;
- if a software system crashes, its duplicate takes over; and when one file is lost, it is replaced by its backup copy.
- The problem with redundancy is that it makes the system much more complex.
- the introduction of redundancies introduces a new set of interdependencies (i.e., how the parts of the system interact), and the consequences of interdependencies may be difficult to anticipate.
- redundant systems are often ineffective if they are susceptible to the same destructive events that caused failure in the primary systems.
- Nature takes a middle-of-the-road approach on redundancy.

Don’t Protect Harmless Information- Save Money!

- Big Data managers tend to be overprotective of the data held in their resources,
- when data is of a purely academic nature, contains no private information, and is generally accessible from alternate sources, there really is no reason to erect elaborate security barriers.
- Protection mechanisms can make a system more complex.
- The data in the system had been rendered harmless via deidentification and could be distributed without posing any risk to the data subjects or to the data providers.
- The value of most Big Data resource is closely tied to its popularity.

Simple Method to Render Data Harmless

- If your data set contains no unique records (i.e., if every record in the system can be matched with another record, from another individual, for which every data field is identical), then it is impossible to link any given record to an individual.
  - *Record ambiguation*
- When every piece of data is a source of profit, measures must be put into place to track how each piece of data is used and by whom. Such measures are often impractical and have great nuisance value for data managers and data users.
  - Avoid profit motive, when reasonable