Data Survivability

Solutions for long-term data retention

CIV Michael R. Cirillo

The DOD is one of the largest producers of data. What would it take to retain any or all this data for more than a century? Is it even worth the effort? Except for command chronologies and similar official data, the Marine Corps has no obvious reason for retaining large pools of data. On the other hand, if we think strategically, we see that we already manage basic survivability for data that is created via email and other office automation software. Additionally, we also do this for defense business systems and national security systems. Thus, we need a strategy to manage the long-term survivability of our data.

The basic view of data retention addresses the near-term—a few months to a few years. The Marine Corps manages that well enough, but each organization faces different challenges. Through the years organizations evolve, leadership changes and people disappear, and the reasons for data collection, storage, and reuse change. An ever-increasing percentage of our organizational data begins gathering figurative dust. This dusty data ends up being left untouched or forgotten entirely. After a decade goes by, no one is thinking about the state of our organizational data and no one is looking into whether or not there is a need to keep data usable for this long.

What is the plan for the survivability of our data?

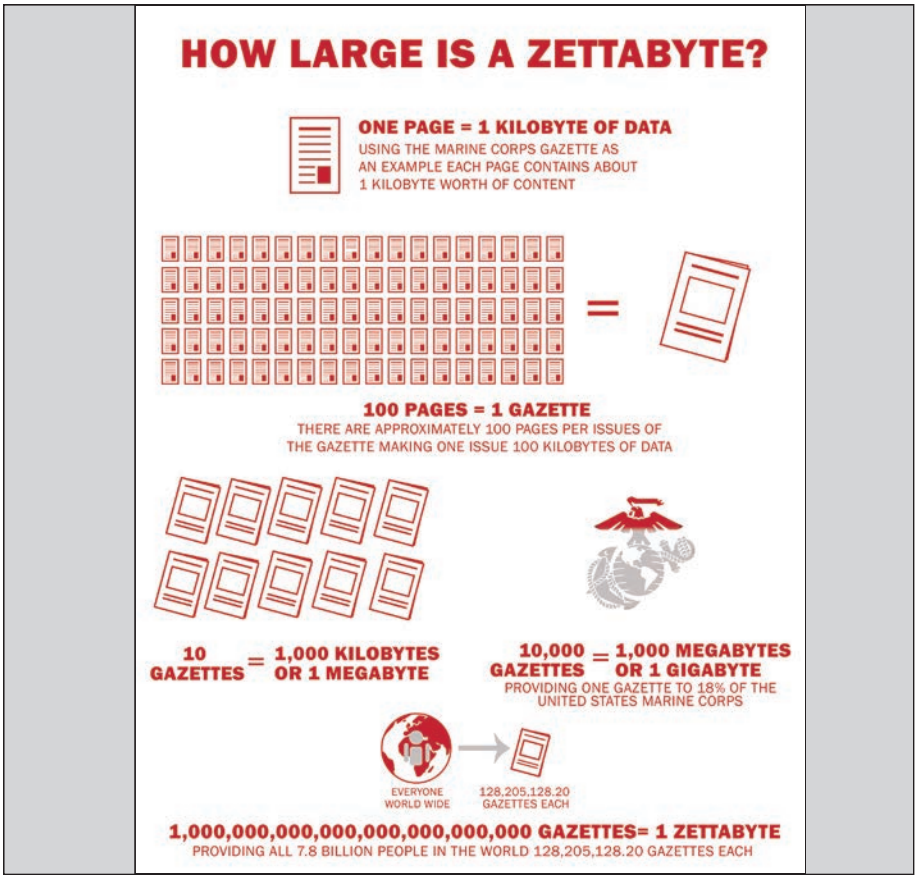
Our weapons systems and certainly our information technology (IT) will not last after nearly a century of use. However, if the systems do not last, how is the data supposed to outlive different iterations of these systems or remain useful for future systems? Is a planned end state simply assumed or entirely ignored? What about the relationship between Marine Corps data and other DOD data? Can we entirely disassociate the survivability of our data from all other data that may need our data?

It is commonly understood that the survivability of anything eventually goes to zero, including data. Knowing and managing that destiny for our data is both a personal and organizational responsibility. Failing to address data survivability today delays the inevitability of being forced to address it in the future. By delaying or ignoring data survivability, we will very likely arrive at a point in time in which some or much of our data is unusable. By preparing our data and putting appropriate processes in place, we can avoid this unproductive end state.

>CIV Cirillo is an IT Specialist and acquisition professional employed by Marine Corps Systems Command, Quantico, VA, as the Strategic Initiatives Lead for Task Force Aquila in the Chief Engineer's office.

Background

Writ large, we do not understand data. Data is something that we assume is fact and thus something that we use


HOW LARGE IS A ZETTABYTE?
ONE PAGE = 1 KILOBYTE OF DATA
USING THE MARINE CORPS GAZETTE AS AN EXAMPLE EACH PAGE CONTAINS ABOUT 1 KILOBYTE WORTH OF CONTENT
=
100 PAGES = 1 GAZETTE
THERE ARE APPROXIMATELY 100 PAGES PER ISSUES OF THE GAZETTE MAKING ONE ISSUE 100 KILOBYTES OF DATA
10 GAZETTES = 1,000 KILOBYTES OR 1 MEGABYTE
10,000 GAZETTES = 1,000 MEGABYTES OR 1 GIGABYTE
PROVIDING ONE GAZETTE TO 18% OF THE UNITED STATES MARINE CORPS
EVERYONE WORLD WIDE
128,205,128.20 GAZETTES EACH
1,000,000,000,000,000,000,000,000 GAZETTES= 1 ZETTABYTE
PROVIDING ALL 7.8 BILLION PEOPLE IN THE WORLD 128,205,128.20 GAZETTES EACH

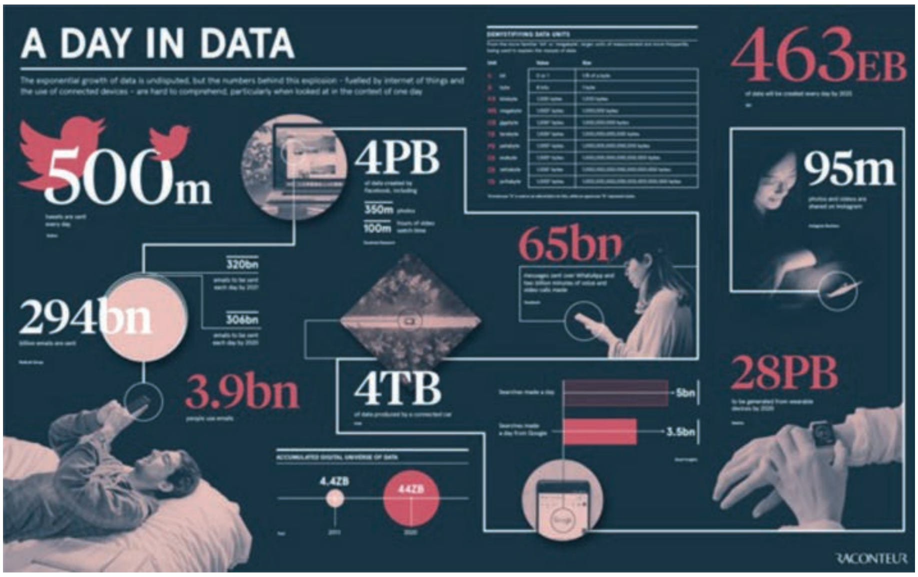
Zettabyte. *(Marine Corps graphic by Cassandra Merchant.)*

to make decisions We struggle to grasp what exactly we each mean when using the term data. One person may refer to the data on Microsoft PowerPoint slides, text in emails, or the emails themselves. Another person may speak of an assay of chemical conditions of metallic rust as data. Yet another person may think that data is what machines use when communicating with each other across a network. Still another person may think of budgets as being comprised of fiscal data and digital money. Let us include all the above because the bottom line is that we need a strategy to manage the long-term survivability of our data.

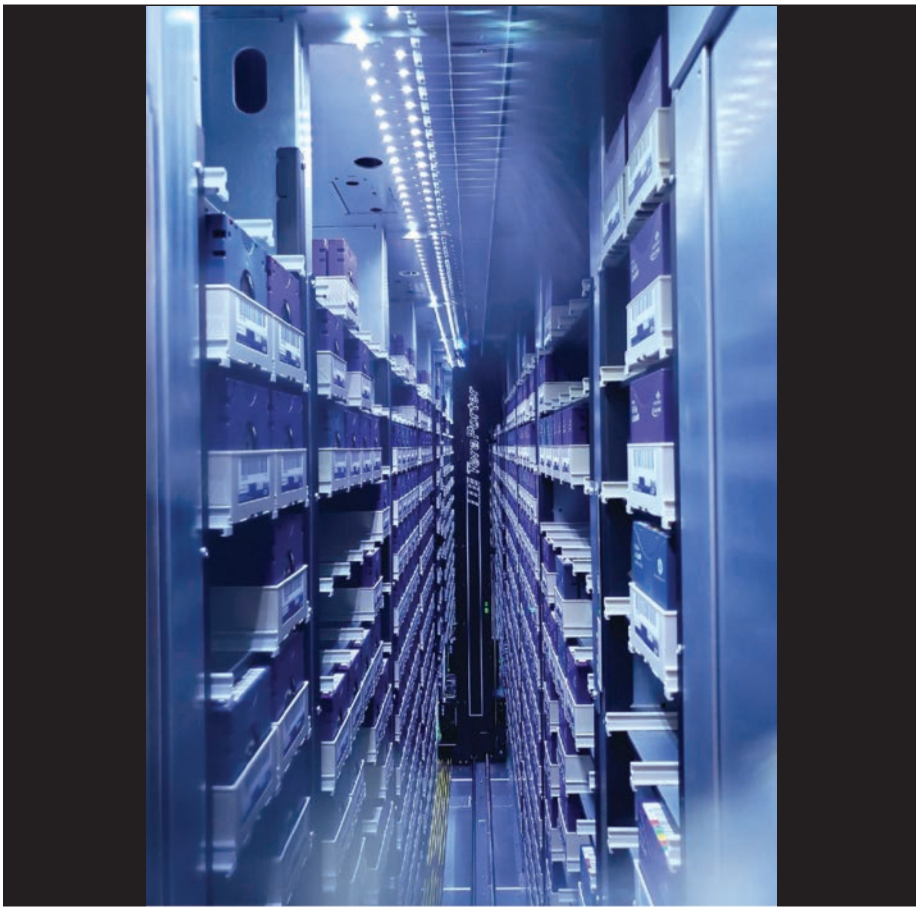
A prime example of not planning long-term for data survivability was Y2K. The Y2K—or Year 2000—problem originated from our inability to effectively address data survivability. Software programming or coding has always had a non-fiscal economic driver. The less code needed and the less coding required, the faster a program is ready for use, and the faster it functions. Shorter code is less likely to have errors or bugs, and fewer problems are likely to appear throughout its use. In the early 1990s, programming languages like COBOL were economic because coding required sharing time on mainframe computers with many other coders. Back then, a year's worth of mainframe time was about equal to five minutes on today's smartphone, so coding time was precious.

Part of this data economization included truncating the format of dates from "1990" to "90." That 50 percent savings in date text reduced processing time; in 1990, we did consider the future implications of the year "2000." The survivability of our program data was near-term: weeks, months, or a year or two. Our myopic view of date truncation did not account for the global impact of all computers changing four-digit dates from the 1000s to the 2000s. If not planned for its survivability, data usefulness atrophies as its survivability goes to zero.

If there was a data survivability plan in play, it did not work to prevent Y2K. In 1999, we were still using COBOL and the truncated date problem still existed. Although Y2K did not erupt into the global disaster many thought possible, substantial funds and a massive international national exertion of programming resources took place to fix or mitigate the Y2K problem. All this effort was avoidable had data survivability been effectively considered.



Twenty four hours of data usage by the numbers. (Courtesy Visual Capitalist.)



NASA Ames Pleiades super computer. (Photo provided by author, personal files.)

Discussion

Within a data survivability strategy, data volume and data quantity are considerations. The sheer, if not exponential, volume and quantity of data created, manipulated, and stored is truly astonishing. Knowing this trend likely will never decrease, the following 2020 data statistics are staggering:[1]

- Every day, people send 300 billion emails and send 500 million Tweets.
- Every second during, every person created ~2 megabytes of data.
- Every day, humans are producing 2.5 quintillion bytes of data.
- Every day on Instagram, we share 95 million photos and videos.
- Since 2019, we created 90 percent of the world's data.[2]
- By the year 2025, humans will generate ~500 exabytes of data.
- By the end of 2020, 44 zettabytes will make up the entire digital universe.

To understand a zettabyte, for simplicity's sake, let us say that one page of text in the *Marine Corps Gazette* contains one kilobyte of data (it actually contains about two). If one *Gazette* magazine has 100 pages, it contains 100 kilobytes of data. Ten *Gazettes* is about a 1,000 kilobytes or 1 megabyte; 1,000 megabytes is one gigabyte or 10,000 *Gazettes.* As we extrapolate this factoring to a zettabyte, which is expressed by a "10" followed by 21 zeroes, this gives us approximately 1,000,000,000,000,000,000,000,000 *Gazettes*.

The Library of Congress contains more than 150 million books and other archived items like maps, microforms, and it has approximately five petabytes of data storage. Therefore, by the end of 2020, the digital universe of data was equal to 500,000 Libraries of Congress.[3] Although the Marine Corps' data survivability challenge is substantially under this number, we still make a lot of data. In order to craft an effective data survivability strategy, here are some basic questions we should consider:

- How much data does the Marine Corps possess, and where is it stored?
- What kinds of data do we have and is it stored, used, protected by categories?
- How much of our data is stored organically by Marines?
- Can we describe the security and protections used for all of our data?
- How old is our data, when was it last accessed or modified, and is it still needed?
- How much of our data is unreachable or unreadable?
- What certainty do we have that data activities support warfighting investments?
- What is the most feasible way to organize future data and (re)organize existing data?
- What is a potential data governance model and potential organizational structure?
- What are the primary steps for implementing a data survivability strategy?

Marines managing data. ***(Photo provided by author, unnamed source.)***

Federal, state, and local governments already store data older than a century. Some of that storage is tangible as paper or plastic products. The Library of Congress spends a substantial portion of its entire budget devoted to this very problem (e.g., our Constitution). Children born today are provided an initial data point—their Social Security Number—which they will need until death and perhaps even after death for surviving family members or aspects of society and government. For example, upon the owner's death, Social Security Numbers are "retired." So, the numbers are not reused, and this data is stored forever.[4] As Americans, we hope the Social Security Administration is executing a robust plan for data survivability.

Long before the data ends its usefulness, technological advances cause electronic media to die or to become obsolete. Individual, organizational, and institutional memories eventually fail us. We forget where the data is stored, how to use it, what its purpose was, or find the data inaccessible/unusable. For example, still in 2021, many files created in the mid-1990s can convert to modern

Error 2000!

Millennium starts 2001.

Try again next year!

Y2K graphic. ***(Photo provided by author, Marine Corps History Division.)***

formats, but some files are lost forever. These seemingly "ancient" files can provide a rich historical context for Marine work and accomplishments from more than two decades ago. This data can provide meaningful connective tissue for the time and circumstances of their creation. This data demonstrates how our ability to access that data changes.

Regarding weapons systems acquisition and sustainment, we understand that artillery concepts and practices do not change overly much. However, artillery technology changes and not just regarding metallurgy and ammunition. Artillery uses IT to manage data today in ways inconceivable decades ago. What is the useful life of artillery data? What is the conscious decision regarding artillery data and its survivability?

> ***… we can understand that, for our data to survive some planned length of life, we need a data strategy. Such a strategy stands in stark contrast to how we … address data survivability …***

A more IT-centric example is how data is vulnerable to software and hardware incompatibilities. Hardware evolves, too, and there is a tandem hardware/software risk to data survivability that requires both to remain up-to-date. A hardware failure can jeopardize the software's ability to recover the data if too much time has passed between the currency of each. In the end, the data is lost not when the software becomes obsolete but when the hardware fails. The two must remain synchronized as part of a plan for data to survive.

For weapons systems data to survive a century, our existing approach requires progressively evolving the data format to ever-newer IT software and hardware multiple times over its useful life. Within this approach likely lies a sincere hope that we are able to do so. A good example of this is how music has changed over the last century. Music data has changed storage formats from phonographs and gramophones to 78 and 33 1/3 speed records, to 8-tracks and cassettes, to CDs, and to an all-digital format. It is very unlikely that this change in format caused an existential problem for music lovers. Music data shows us how the hardware and software change, but the need to preserve the data for future use constantly presents us with a survivability challenge.

For data, time increases the opportunity for error. Incorrect data entry due to human error, data and operating system incompatibilities, and hardware incompatibility are just some of the kinds of errors data survivability faces. Given that we will likely use weapons systems for much more than a decade, are we planning for the inevitable hardware and software changes advancing technology will brings us? Even if a system vendor is still going in 30, 50, or 100 years, the application and its architecture will almost certainly have technologically shifted sufficiently enough to be incompatible with 30-year-old data, much less 100-year-old data.

Conclusion

In the end, we can understand that for our data to survive some planned length of life, we need a data strategy. Such a strategy stands in stark contrast to how we generally and habitually address data survivability, which is tactically. Our current approach has us grinding away at incompatibility problems as they arise. Stored data has hardware and software dependencies, each of which can move quickly into obsolescence. We do not recognize that data survivability is a problem for our successors.

We need a strategic and data-centric approach for data survivability. We need to appreciate the need for our data to survive—perhaps a century or more of storage and utility. We should intentionally plan for data survivability and document the reasons why we did or did not do something for our data to survive. Current hardware and software have hidden dependencies that are exposed over time. We need to invert our thinking from hardware and software centricity to data centricity. Data survivability needs prioritization over hardware and software.

Returning to our Y2K example, the current fix for Y2K will lead inevitably to a crisis in the year 10,000 when programs will again manifest as having been designed to fail.[5] The most common fix for the Y2K problem was to switch to 4-digit years. This fix covers roughly the next 8,000 years (until the year 9999). It seems to be commonly understood that all current programs will have been retired by then. This is exactly the faulty logic and lazy programming practice that led to the Y2K problem. Programmers and designers tend to assume that their code will eventually disappear, but history suggests that code and programs are often used well past their intended circumstances.

Notes

[1] Jacquelyn Bulao, "How Much Data Is Created Every Day in 2020?" *Tech Jury*, (January 2021), available at https://techjury.net.

[2] Jeff Desjardins, "How Much Data is Generated Each Day?" *Visual Capitalist*, (April 2019), available at https://www.visualcapitalist.com.

[3] Staff, "The Zettabyte Era Officially Begins How Much is That?" Cisco Corporation, (September 2016), available at https://blogs.cisco.com.

[4] Melissa, "What Happens to Your Social Security Number When You Die?" *Gizmodo*, (October 2014), available at https://gizmodo.com.

[5] S. Glassman, M. Manasse, and J. Mogul, "Y10K and Beyond," (Palo Alto, CA: Compaq Computer Corporation, April 1999).