Collaborative Discussion 1: Information System Failure

Initial Post

A classical information system usually provides three functions: collect real-time raw data from outside, like sensors, processing data to convert to meaningful internal information for the next step using or showing them to the user. Finally, based on this information, users can apply a pre-defined strategy or take some actions, e.g. control other system components.

A plane also can be considered as an information system. The system can have a massive impact on the flight, so that safety would be the highest priority in the whole system. However, Boeing 737 MAX had a critical flaw in the system that led to two crashings, hundreds of people died, and all 737 MAX had to ground around the world.

The 737 MAX had one sensor to collect flying angles to the system. The system would take the data as input to determine what position is proper in flying, and it is called Maneuvering Characteristic Augmentation System (MCAS). But the system had not to be considered or ignored in case one sensor could collect inaccurate data or be damaged and the system could send a command to adjust position into the wrong way. Pilots could fix the position with manual, but later the system reissued the instruction. This unstable process can last until the plane crashes.

We can find a series of information system engineering mistakes in the whole system about design, development, testing and training user.

1. The system designer ignored the redundancy principle for a critical-orient system. We can not take for granted that each system component can always be ideal. For 737 MAX, only one sensor to collect critical data is too dangerous. The system how to handle the sensor failure situation?

2. When there is a conflict between the user's command and system pre-defined command, there should be an obvious mechanism to resolve the dispute and not just let the system be uncontrolled status. Boeing had never told pilots there had an MCAS in the plane and never told how to disable it in the pilot training.

3. In the testing stage, Boeing software engineers had found that the issue over wrong data from one sensor could lead to a safety problem, but the situation had not been fixed and still went to the delivering product stage. Reality circumstances are usually worse than the testing environment, so it is like gambling to think that the issue will not happen.

Modern information systems can be more and more complex. Even people have complete confidence in their design, but a system would rely on many parts from outside, e.g., essential hardware, database, 3rd software module and operating system. Any of them into trouble could lead to a disaster. The worst-case consideration, ignoring related safety issues, and inappropriate user training can't be acceptable.

summary post

It is fair that modern society can no longer work without various information systems (IS), from traffic lights managed to planes on the fly, from banks to the government. There has countless IS at work to support human activity. And the high quality of IS is the crucial factor to being successful.

Before people try to build a high quality IS, they have to understand the quality of an IS. "Excellence in IS quality involves using state-of-the-art technology, following industry 'best practice' software standards, and delivering 'error-free' performance. The value of IS can be realized by improving profit margins for the firm, providing easy-to-use and useful applications, and designing easily maintainable software. " (Gorla, N., Somers, T.M. and Wong, B., 2010). We can see a high quality IS not only about the software itself from this definition.

When we review the failure IS examples from the discussion post, we can see many different parts that could lead a IS to be failed. Some were about incorrect requirement analysis, and some were about invalid data input. On the other hand, not only software parts could cause an IS not be working as expected. People may ask for more testing to discover these bugs that could cause disastrous results. However, "Thus, testing all possible paths within a complex program is an impossible task" (J. Glenn Brookshear and Dennis Brylow., 2019).

Software engineers borrow some ideas from the real-world construction industry that introduces software engineering, applying object-orient programming(OOP) to design and implement components in IS, using code statically analysis tools to detect bugs in the early, promoting open-source style to let more people involve a project and more bugs could be found out and adoption agile development flow to invite customers to the development cycle.

However, we should keep in mind that IS is considered a product of human ingenuity. It is also as will subject to limiting human thinking. Change another word, IS could be wrong in some situations for various reasons, and it is inevitable. We must analyze the most important modules of the information system and set up appropriate redundancy mechanisms to prevent the system from being brought out of control in the worst-case scenario. We also need the design to override the method in case of emergency and correct errors within the system.

Reference list:

Gorla, N., Somers, T.M. and Wong, B., 2010. Organizational impact of system quality, information quality, and service quality. *The Journal of Strategic Information Systems*, *19*(3), pp.207-228.

J. Glenn Brookshear and Dennis Brylow., 2019. Computer Science An Overview. *Thirteenth Edition,*pp.420.

Discuss With Roberto Cappiello

Hi Roberto. Thanks for your sharing.

The investigation about the Queensland Health payroll system's failure revealing a problem is widespread in the government/industry sections, "Queensland Government failed to communicate to IBM the business requirements for the workforce of Queensland Health and this brought IBM to fail in design an appropriate payroll system to accommodate the complexity of pay rules. ".

The waterfall software development style is still trendy when designing an extensive system for non-general consumption. People can figure out customers' real needs is the most crucial thing in the waterfall cycle. However, making a clean requirement for these projects sometimes is very hard. The things in the customer's brain may not be easy to describe into an immaculate concept; the software company may have not rich experience in these sections, just all about solutions for another area and it but may not be suitable. Now, some IT companies apply some new development ways to solve these issues. For example, the agile way could involve clients in the whole development stage, and it can find out the deflection in the early stage. But it still rares using in projects was related to governments.

However, in other government departments, such as medical and construction, government personnel generally have appropriate professional backgrounds. Generally speaking, when people with shared professional experiences discuss professional issues, they communicate much more efficiently. The chance of errors caused by miscommunication is also lower, so perhaps we can explore this aspect in these large government software projects (Niam Yaraghi , 2015) .

Reference

Niam Yaraghi (2015). *Doomed: Challenges and solutions to government IT projects*. [online] Brookings. Available at: https://www.brookings.edu/blog/techtank/2015/08/25/doomed-challenges-and-solutions-to-government-it-projects/.

Discuss With Rob Mennell

Hi Rob, thanks for your sharing.

We should express our gratitude to those who quickly discovered that this was disinformation. Your example illustrates an important issue. How we know or guarantee the accuracy and reliability of data when an important decision or calculation result of a system depends heavily on some external input data. "General, what you see on that board is not reality, it is a computer generated hallucination" (rr, K., 1998)

There should always be some way to examine data accuracy in a critical system. For example, it should never rely on a single data source to make some decisions that could get a severe consequences. There could be a mechanism to detect inconsistencies between data from different locations. In my initial

post, I also used Boeing's 737 MAX failure case to show that using a single data source to take some automation decisions led to tragedy.

Reference

Orr, K., 1998. Data quality and systems theory. *Communications of the ACM*, *41*(2), pp.66-71.