B. V. Patel Institute of Computer Science/Shrimad Rajchandra Institute of Management and Computer Application

Programmes: BCA/IMCA **Semester:** 5th

Course Code & Title: 030010517/060060511- CC12 Software Testing Techniques

Integral Evaluation: Case Analysis and Concept Mapping -2

Date: 3rd December'2020 **1:00 p.m. to 22:00 p.m.**

Mode: Online (Moodle LMS) **Total Marks: 30**

SET - A

Case Study 2010´s – Stock Market Mistake (KCG)

An American company, called Knight Capital Group, which operates in global financial market. More specific electronic execution and institutional sales and trading. In the year of 2012 KCG was the biggest trader in US equities, market share of around 17%. It´s trading division Knight´s Electronic Trading Group, handled daily more than 3.3 billion trades, which converted to money equal over 21 billion dollars. The New York Stock Exchange (NYSE) was planning to take into action a new Retail Liquidity Program. This program was meant to offer improved pricing to retail investors with the aid of retail brokers and the launch date was set to August 1, 2012. To prepare for this operation, KCG updated their high-speed, automated algorithmic router. This router was designed to send orders for execution into the market. This trading algorithm was called SMARS and one of it core function was to receive orders from KCG´s other systems which uses the same trading platform. It was coded to function in the way, that when it receives big orders from the trading platform, it simply divided them into smaller orders. Idea was to find buyer or seller that would match for the number of same shares.

So basically, the bigger the "parent" order, the more "child" orders it would have created. The reason for the SMARS update came from need to replace the old system. This old and unused code was called "Power Peg" and in this point it has not been used in 8-years. When the new code was updated it has a function that should recreate an old and unused "flag" that was formerly used to activate functionality of Power Peg. The new code had completed the tests and seemed to work reliably and correct.

Root Causes

This new software was manually deployed between July 27 and July 31 in 2012. One by one the servers received the new update, in the end eight servers had been updated. This is what the SEC report indicated about the deployment process. "During the deployment of the new code, however, one of Knight's technicians did not copy the new code to one of the eight SMARS computer servers. Knight did not have a second technician review this deployment and no one at Knight realized that the Power Peg code had not been removed from the eighth server, nor the new RLP code added.

Knight had no written procedures that required such a review."

(SEC Filing, 2013).

On Wednesday 1st of August in 2012, the markets opened in the morning. This day started like an average Wednesday. KCG´s trading platform started to process orders from broker-dealers and handle them for the new Retail Liquidity Program. Even in this point everything went smoothly. The problems started when the seven correctly working servers had operated their orders and it was eighth server turn. When the orders arrived at this server the new code activated the old "flag" even it was meant to overrate the old purpose, but it went the other way around and it woke up the old Power Peg to alive. To understand why this was so catastrophic, the purpose of the old code needs to take into deeper review. Old code counted the shares which was sold and bought against a parent orders every

time child orders were completed. So, this means that Power Peg kept a record of child orders and stop them as soon as the parent order was ready. When the eighth server “flag” was activated the operation started to route child orders but was not keeping track about the number of shares of parent orders, like a never-ending loop. In the morning people started to notice that something was wrong. In the Wall Street people were holding their breath for a couple of minutes and started to quickly wonder how this could be possible and why it was still continuing. In the fast trading world these minutes welt like hours. Within the disaster KCG´s trading executions created more than half of the total daily trading amount. Due to that, some stocks rise over their value and due to that other stocks dropped in their value to scale the error.

Whole disaster took 45 minutes. During that time KCG tried to stop these trades several times. Sadly, there were no kill-switch or any documented info about how to react in this kind of situation. This led to a situation where they had to diagnose and solve the issue in live production environment and at the same time almost 10 million shares were traded in every minute. After a while they noticed that they can´t solve the issue what was affecting the false orders. What happened next only made the situation worse, they started to uninstall the new code from the servers that was actually deployed correctly. So, in the panic they erased the working code and left the damaged code into servers. Now instead of one server all the eight servers were activating the old Power Peg function.

Finally, after 45-minutes of trading they managed to stop the system. Harm had already done, during the time servers were online they processed 212 parent orders and these orders SMARS converted to millions of child orders. As a result, these child orders converted into 4 million transactions which equals almost 400 million shares. When all was over, company suffered $460 million losses and almost had to close the doors for good.

What Was Learned

What happened in August 1, 2012 is a book example for every development team about how not to operate. Even the software has been writing and tested well it is not enough if it is not delivered properly to market. Not only one person could be blamed for this case. Company had not thought about the deployment process in the level required for the risk there were. Basically, process was only built counting on human actions and without any safety backups. Mistake could happen so easily in this kind of process, it could be just misunderstanding of instructions which cause wrong execution. To prevent these kinds of disasters, couple of points should take into consideration.

• An automated deployment system

• Better configuration and test automation

When deploying software, it should be a reliable and repeatable process. Also, consider that automate as much as is moderate and as much as the situation allows.

Do as directed:

1. **What is the major technical problem of focus in the case?**
2. **Enlist three testing types that have been poorly identified or done.**
3. **Discuss any one testing type with reference to the above case and explain how its test plan could have been prepared in a better way to avoid failure.**
4. **Is the system of stock user acceptable? Justify your answer with any three valid technical reasons.**
5. **Is the system testing done efficiently? Justify your answer by listing any two parameters that have been satisfied and not satisfied.**
6. **Which two deployment testing concepts are mentioned in the case? Give one keyword each that match the concepts.**