COMP47480 Seminar 2: Facebook

Sukrat Kashyap (14200092)

6 March 2018

1 Speaker

Richard Sheehan (Production Engineer) and Mike Elkin (Site Reliability Engineer) at Facebook.

2 Report

The speakers started with a statement which said that "Failure is not an option but it is inevitable". They meant to focus on the fact that Failure is a part of Software Development. It is something we definitely do not want but at the same time it is inevitable and cannot be deferred. The main idea behind was to accept this nature of Development and the team should be able to learn from them and make sure it never happens again.

They showcased the vast Facebook's data centre all over the world. Each datacenter was built in order to manage massive traffic coming from not only Facebook but from Instagram, Whatsapp etc. They also showed the switches that they build for the data centres which they chose over purchasing from the vendors because of the cost. Each data centre was supposed to have a massive infrastructure to handle large traffic from the network. Now with such big infrastructure possibility of error is vast. Identifying and resolving them is another big challenge. They gave us an example of how one of the errors that occurred shut down the entire Facebook data centre. Performance is also important and with such large infrastructure, it becomes harder to optimize and increase performance. They showcased their software development methodology by giving an example of improving their performance of data transfer between networks to users.

Their methodology does not include agile planning or software development. It consists of abstract project development method which is as follows:

1. Investigation
2. Analysis
3. Project Retrospect
    - Set clear project goals
    - If you cannot measure it, it did not happen
    - Get the right stakeholders
    - Requirements gathering
    - Career progression is not linear
    - Sunk cost fallacy (if not working move on)

The overall structure is quite similar to IBM's Vision, Plan and Develop. The words involved are quite and different and so is their idea behind it. IBM focuses more on software as building a machine whereas Facebook's development is in terms of numbers if a project tends to increase the performance they ask themselves by how much. Both have teams working on projects but whereas IBM follows their project lifecycle. Facebook leaves everything to the team. Facebook has more of an "ownership"

way of development where they give the project to a team and then the project is owned by the team in a sense that they can decide whatever it seems fit.

Now when it comes to errors, they need robust methodology to tackle the errors occurred and a plan so that it doesn't ever happen again. Their immediate step was the following:

- Reboot servers
- Disable automation
- Check rest of the site's status
- look for config changes or deployment

These steps are simple enough to tackle errors occurred followed by small hack or fix to keep the site running for the time being. After this software engineers gather together in "War room". Where they try to do the following:

- Try to keep the rest of the site running
- Reproduce failures on a small scale
- Look for code/config changes
- Dig into and analyse data

Now, this is the immediate measures taken by the engineers at Facebook. But since we need something to fix the problem such so that it does not happen again. For this they do the following:

- Focus on the root cause (Deep dive and Not blame)
- Find how to prevent recurrence
- Do not rely on best intentions
- Always remember humans are desperately unreliable

They showed with example errors that actually occurred on Facebook and how they followed the above steps and guidelines to fix the memory leak and overloading of traffic on servers problems.

To prevent failures unit testing, integration testing and load testing are one of the ways. But testing would not catch all the failure nodes and one can't test an entire distributed system at scale. Hence, they try to:

- Roll out stuff slowly
- Monitor KPIs
- Fail fast than timeouts
- Defensive servers

Since contribution, changes, code and bugs are all proportional to each other. Hence, small changes help them to figure out what broke when. The error tackling process is very different from IBM's as most of the errors are caught before the development pipeline and if not immediate fix is created but proper fix is done in the later stages of the cycle of development whereas in Facebook they do not of have cycle of development, they release as they build and if error occurs they rollback and fix and then keep developing.

The code rolling out is in small chunks but every day. Whereas in IBM's agile process rolling out of new code is every 2-4 weeks. Every team in IBM follows a proper agile planning whereas in Facebook it is on ownership basis. Testing and code reviews are done by both the companies in the same way but when it comes to testing IBM have their own pipeline of testing where it goes through a number of nodes (Testing on local, Testing on a server, UAT testing) before declaring as pass. Whereas in Facebook, test are usually against what they develop and only those test and their dependencies are run.

I feel such varied in the development process is due to the sheer number of computing and data. If Facebook had to use IBM's way it would fail because rolling out such big change is risky. Rolling out to such large data centres is another impossible task. With such large code base testing, each of them would take ages. And because of this sheer number they do something call reactive programming where they automate most of the tasks and wait for the signal from the system if something is wrong. They both have automated tools for detecting software vulnerabilities, code smell etc. Facebook uses their own whereas IBM uses SonarQube.

If the same is reversed where IBM uses Facebook way of development it might not do as well as IBM has way more employees than Facebook and managing and tracking would become a difficult task. Also, most of the Facebook's apps run on their servers, hence they provide services. Whereas IBM does both they provide services by running software on their machines and they also provide software to be run and used by the buyer. Hence rolling out would not be possible every now and then. Also, rigorous testing is required as rolling back or updating a software running on someone else's machine is difficult and inconvenient. So, for making sure that their rollout is stable their software has to take an agile process of development.

3 Q & A

Some question that were asked at the end of seminar were:

1. How do they manage the test as their code base is huge?
   Answer: They focus more on unit testing as they are quite fast. They have tools which run tests only on the new code and their dependencies.

2. Development methodologies used at Facebook?
   Answer: Development process is chosen by the team. They choose the best model according to the project.

3. How is code quality maintained in such a large codebase?
   Answer: They have automated tools to find code smells and memory leaks. Also code review is a must before going into production.