A STUDY IN MAINFRAME DEVELOPMENT: WORKING FROM HOME IN 2020

The year 2020 will be remembered for many things and one of those is the largest exercise in employees working from home. With few exceptions across the world, most employers told their employees to start working from home in early March. Business Continuity plans were dusted off and implemented. Some plans may have been better than others; some may have played it by ear while others may have had a very detailed plan.

Regardless, this was very disruptive, especially for those companies that found the colocation of a development team to be a strategic advantage prior to 2020. Teams that sit together in a physical office communicate in ways that may be more difficult when everyone is sitting at home and trying to collaborate over the company's chat application (e.g. Microsoft Team, Skype, Slack, etc.). In the office, developers struggling with a particular problem or using a tool can sometimes show a body language that a more senior developer or manager can interpret and offer help without being asked. Hallway conversations, over-the-cube conversations or just walking over to someone's desk for advice do not happen as easily on Teams or Slack. This is not a problem central to development; it affects all aspects of a company.

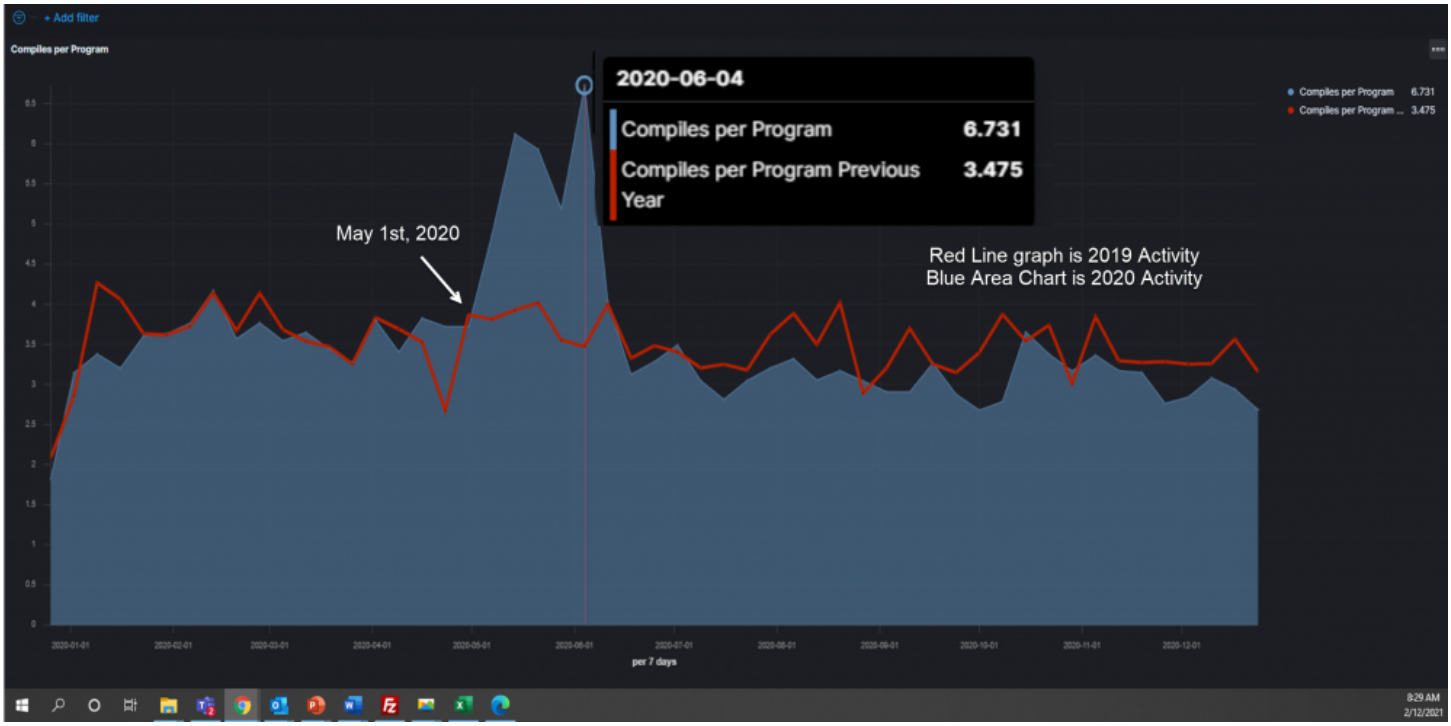
We're one year into the crisis and from the anecdotal evidence it looks like most companies have and will survive this period with little loss of productivity. At Compuware, a BMC Company, customers participate in a free service called zAdviser. zAdviser collects telemetry data from the development process when Compuware tools are used. A developer debugs a program, an abend occurs, code is promoted, code is checked out—all of these activities create data which makes its way to zAdviser, where the free service allows our customers to view this data and learn how to become better at mainframe development.

This was originally an exercise to see if there was any fatigue evident as employees were spending more time at home. In analyzing the data from zAdviser, data from 2020 showed three distinct anomalies when compared to the same periods in 2019.

Below is the analysis of these three periods and how they compared to when developers started working from home.

Period 1: Compiles per Program

From April 2020 to June 2020, there were significantly more Compiles per program than we saw in 2019. In the visualization below, the red line represents the Compilations per program weekly (defined as # of Compilations / # Unique Programs over a one-week period) and the blue area represents activity from 2020. For most of 2020, the metric matches 2019, with 2019 being slightly higher in the July to December timeframe. From May to Mid-June there was significant increase in the number of Compiles per program.



What is going on here? Are developers adding display statements into the code, compiling, executing the code and checking the output to see what happened? There are times where you make one simple change to the code, compile and promote to testing. Developers should be utilizing Xpediter (source code debugger) and Topaz Program Analysis to ensure they understand the application and what the ramifications of the change are. Do the developers feel confident in using the tools and if they do not, are they confident they can reach out asking questions of those that do? Communication in an office where developers are colocated happens very easily, “Hey Joe, how do you set a watch on a variable again?” Picking up a phone or texting someone through Microsoft Teams may be more intimidating for some.

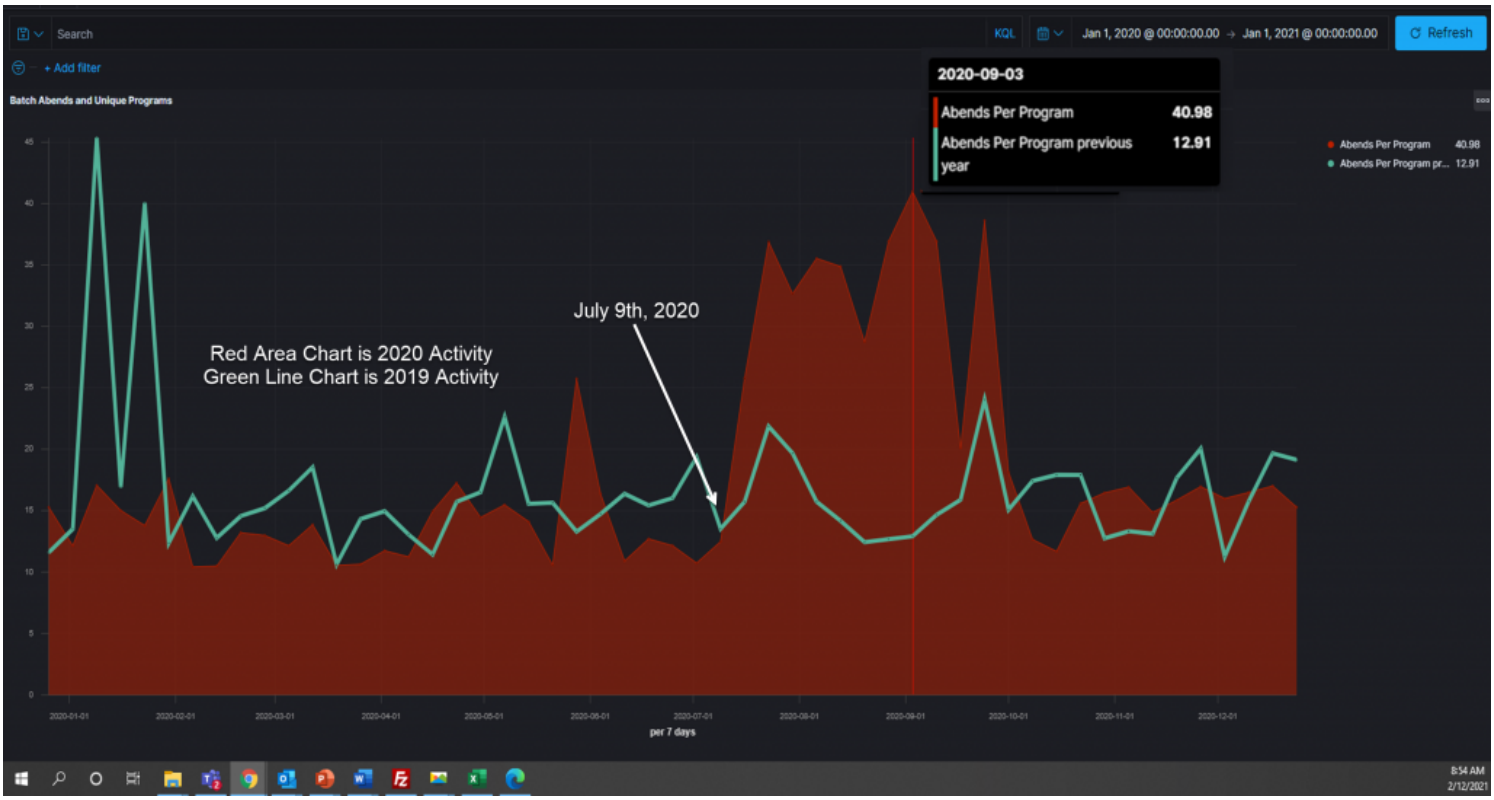
Period 2: Batch Abends per Program

This metric looks at how often a program is abending on a weekly basis. The definition is defined as sum of batch abends / number of unique Programs

For those not familiar with the mainframe term, “abend” means “abnormal end” or simply the

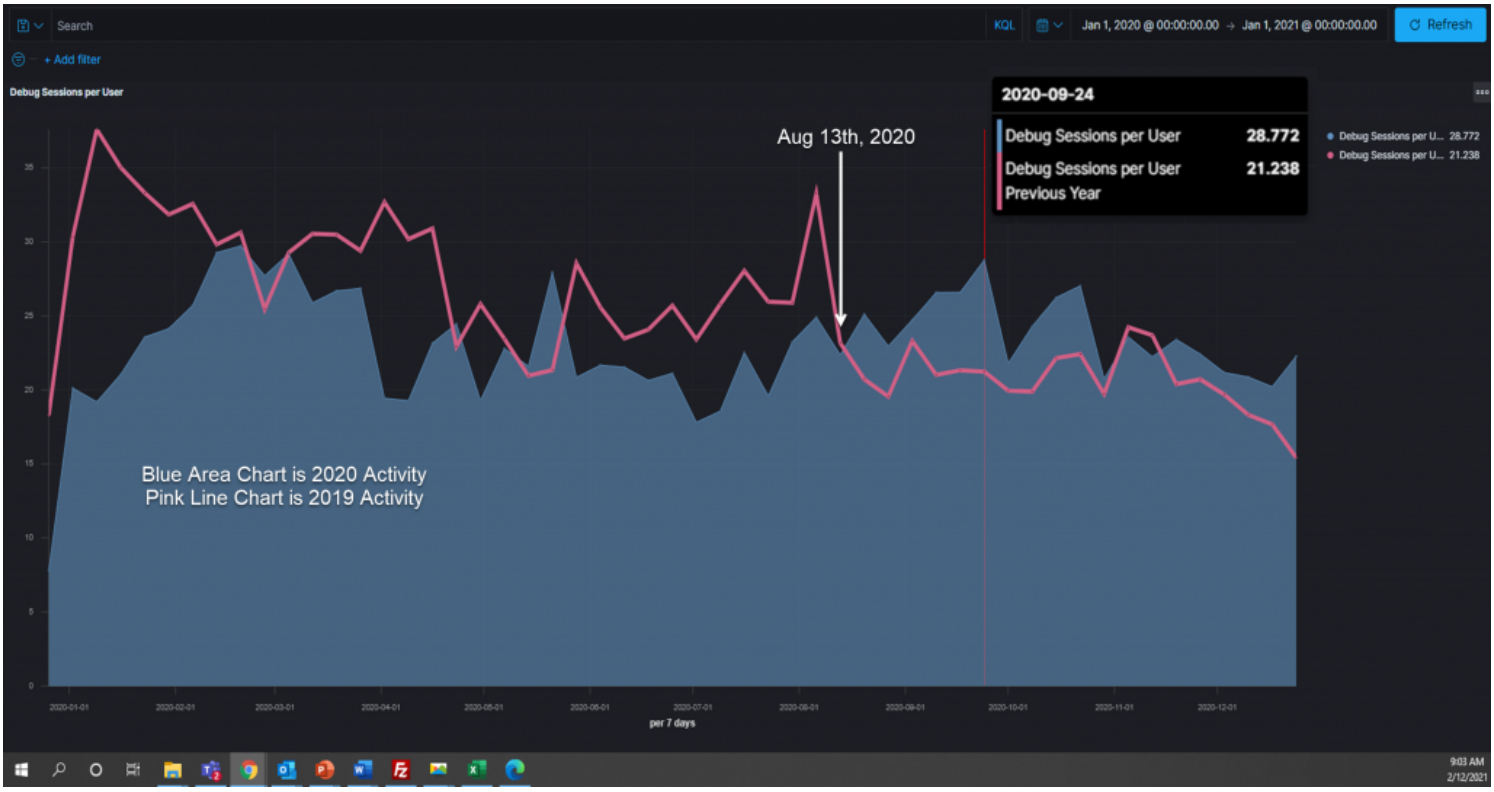
program terminating due to some condition or “exception”. It may have tried to read a file that was not available, reading a string into a numerically defined variable, or simply maybe it tried to divide by zero. Whatever happened is generally “not good” and often requires a person to investigate what happened to try to get the program running again. This can be done by regressing the code to a previous version, making sure the file trying to be read is available, removing the offending records that are causing the problem, or something else. This elevation in batch abends per program continues from mid-July until October 1st when the metric returns to a curve like 2019.

Is this the result of what happened in Period 1 and of not using Xpediter and Topaz Program Analysis? Based on the two time periods 1 & 2, it appears there is a cause and effect. Ineffective development and testing processes may very well be the instigator to a higher than normal number of abends. This is significant, especially when you look at what 2019 looked like.



Period 3: Debug Sessions per User

This metric looks at how often a user is debugging a program. The definition of the metric is # of times programs are debugged by # of unique users or Sum of Programs Debugged / Sum of Unique Users.



This is the metric and visualization that pulls the story together. Starting in March 2020 the metric shows the amount of debugging being done is falling compared to what was seen in 2019. This corresponds to the time when companies were forced to close their physical offices. If you recall the first metric that was discussed in Period 1, the hypothesis was that there was less use of Xpediter and Topaz Program Analysis. There is some recovery towards mid-March before the metric dives sharply starting with the third week.

Contrast that to the 2019 data when the number of debug sessions per User increased. Starting in May, the metric begins to recover before dropping again in June and lagging the 2019 data until mid-August. In Mid-August through November the metric exceeds the 2019 activity significantly. Most companies have system freezes of new code starting in the fall, is this the rush to beat that cutoff? Have companies also become better at communicating between remote workers? These are questions that are difficult to answer without more analysis and conducting interviews with those that supplied the data.

Summary

While each of the individual periods is interesting on its own when compared to 2019, if they are looked at together, patterns emerge. After developers started working from home in March 2020, they debugged less and started adding code to applications, compiling and re-running the program to see if what they did worked. This is not the ideal approach to developing code. This code was ultimately deployed to production and because of the lack of sufficient testing and debugging, there were significantly more abends during the mid-July to October 2020 timeframe (Period 2).

At this point, since PLAN A of compiling and running the program (Period 1) to see if a new feature or bug was fixed didn't work, developers started to fall back to the tools that they should have been using to test and fix the bugs. After six months of working from home, communication improved, asking for help became easier, managers became better at spotting developers needing help, and the bugs introduced when the code was deployed were fixed prior to the end of year system freezes.

About the Data and Analysis

This data represents two years of data collected from the Compuware customers who participate in zAdviser. zAdviser is a free service for all customers which collects telemetry data from Compuware products; in addition, customers supply data from their SCM and ITSM toolsets. This data is used to form KPIs so customers can understand how well they are developing mainframe code. Machine Learning algorithms analyze the data and determine prescriptive analytics to help developers become better by suggesting small nudges such as learning a function that they may not be using in one of the Compuware tools to become better developers and deliver code faster.

Different permutations of the data were studied, looking for patterns. The dataset was normalized by user and by program where appropriate. A great deal of time went into studying different metrics and different data sets, looking for patterns and discerning what happened in 2020.

This represents over 1 million data points across all customers. The data does not include any Compuware data, which *is* captured and stored in zAdviser, but many of those datapoints are from internal testing of the tools. Adding this data to the dataset would impact analysis.

This post originally appeared on LinkedIn.