The Making of a Large-Scale Online Ad Server

Practical Lessons Building one of the World's Largest Online Ad Servers

Brendan Kitts
Applied AI Systems LLC
Seattle USA
bkitts@appliedaisystems.com

***Abstract*—Data Mining promises potential improvements in a variety of real-world problems. However large-scale implementations face an array of challenges between their inception and deployment, many of which are not technical in nature. This paper will describe the author's experience building one of the largest data mining systems in the world. The objective is to help orientate practitioners to the challenges involved in system implementation, and to suggest general strategies that practitioners can use to successfully navigate development.**

Keywords—data mining; information systems; large scale; implementation; deployment

I. INTRODUCTION

Microsoft adCenter's Marketplace Traffic Quality system is one of the largest of its type in the world, perhaps only bested in size and complexity by Google [39]. The scale of the system is incredible. Microsoft's system processes over $2 billion in revenue, and makes billing decisions on every click, impression and conversion. Microsoft processes approximately 17% of US search traffic – only Google is larger [41]. The system scores over 300 times the number of events as are scored in credit card transactions in the United States. Every hour the system scores nearly a billion events [26].

The story of how this kind of system was built is surprisingly complex. Development at Microsoft is challenging due to technical complexity, the need for 24-7 operation, and organizational challenges [29]. The present paper will recount the author's experience including what worked and what didn't work in the quest to build one or the largest automated data mining systems in the world.

II. THE END OF THE INTERNET ECONOMY?

In 2006 the spectacular success of Sponsored Search that had fueled the internet's meteoric rise was running headlong into its greatest threat [17]. Google generates most of its revenue from pay per click search. This is an ingenious system which works great as long as the clicker is a person. But there's just one fatal flaw. What if the person clicking on the ads isn't a person after all? For example a website displaying Google ads (eg. www.badpublisher.com) receives a bounty whenever a user clicks on an ad. The same publisher could start clicking on their own ads to produce more cash payouts. A sufficiently motivated publisher could treat an unprotected pay per click system like a "printing press" to print their own money. This problem started as a nuisance, but soon looked like it would become an epidemic and threaten the internet economy.

A range of major click fraud attacks were uncovered. ClickBotA was found to have over 100,000 infected computers to generate clicks on search results with a sophisticated command and control system to maintain fewer than 20 clicks from any one IP [10]. The Chameleon Botnet was estimated to have defrauded advertisers of $6.2 million dollars [38] by forcing infected browsers to redirect through payment sites. The WOW Botnet also used hundreds of thousands of IPs to attack advertisers on sponsored search listings. This time the botnet went after advertisers, draining their budgets in a rapid distributed attack, and allowing a conspirator advertiser to soak up high quality traffic for a thousandth of its market cost, which it then monetized [5], [8], [33], [12], [13].

Click Fraud Software was even being developed and sold by a large number of shady operations. This put the task of attacking search engines within easy reach of vast numbers of people. LOTE Clicking Agent, Smart HitBot, Robin Hood, Fakezilla and iFaker all automated fake clicks with statistical features to try to blend in the robotic attacks with human traffic.

Human clicking schemes were being widely reported also. Google successfully sued Auctions Expert Ltd, Houston in July 2005 for running a scheme in which dozens of employees clicked on ads [5]. Agloco took human clicking operations to a whole new level by creating a pyramid scheme which encouraged users to download the Agloco search bar. The company then would pay a bounty every time the user clicked on an ad. Even better, if you referred Agloco to a friend and they installed the search bar, you would receive a bounty for their clicks also.

The increasing severity of attacks led Google's Chief Financial Officer, George Reyes, to remark that "I think something has to be done about this really, really quickly, because I think, potentially, it threatens our business model," [9]. He also declared "Click fraud is the biggest threat to the Internet economy" [31]. Yahoo had similar concerns: "Anyone who says this is not a real challenge is kidding you", John Slade, Senior Director Product Management, Yahoo! Search Advertising [12].

Click Fraud was being covered on all of the major news outlets including CNN, Wall Street Journal, New York Times. Headlines appeared including "The Google Bomb" (Silicon Daily News, May 5, 2005), "Click Fraud: The Google Killer", WebProNews, May 65, 2005), "Click Fraud: Is it Happening to you?". The Times of India even ran a story titled "India's

Secret Army of Online Ad Clickers" which described armies of Indian housewives who were working part time to click on ads.

Lawsuits began to pile up for damages due to alleged click fraud attacks, including Yahoo vs Checkmate Strategic Group 2005, Google vs Lanes Gifts in 2006, Samuel Lassoff vs Google in 2006 and Crafts by Veronica vs Yahoo in 2008.

III. adCenter Launches

I arrived at Microsoft in January 2006[1]. Microsoft adCenter had just gone live in September of 2005[2], and had coded in a rudimentary "invalid click" system. This system had been baked into the Business Intelligence (BI) / Reporting data processing pipelines with different logic spread throughout the pipeline. The method of distributing logic meant that any kind of change to the model meant a full re-coding of the pipeline with enormous implications across all systems and the potential for introducing bugs.

IV. Quick Progress

By February I had also identified what looked to be a "quick hit" in another part of adCenter's ad serving system. This was an opportunity for Microsoft to improve user matching quality and yields at the same time. Microsoft adCenter is a Sponsored search system that shows advertising in response to search queries typed by users. If a user types in "shoes" then ads for shoes will be shown. However, what about if the user typed in "sneakers"? Mis-spellings, synonyms, acronyms, and semantically related words, all are important for matching the user's intent to the ads that are returned.

It looked that we could build out a lookup table of these keyword expansions – i.e. if the user types in shoes then expand to sneakers, and code the expansion table with an algorithmID so that we could track the performance of the different algorithms. The expanded keyword was simply added to the user's search query. We built a "flighting system" so that the algorithms could be randomly rotated. I created a working code prototype in Perl and a functional specification. I also created a default expansion algorithm.

This expansion idea caught on. Development teams and the General Manager all supported it and believed there was revenue potential. We put it on the development roadmap to deploy in July 2006. The idea also captured the imagination of research teams. Microsoft maintains a very large contingent of research teams including Microsoft Research, Live Labs, and adCenter Labs. Many teams wanted to contribute algorithms and have the chance of their algorithm creating value at very large scale.

The beauty of this project was in the data representation – all that the teams had to do was to simply provide a text file in the following format:

(keyword,expansion,algorithmID)

There was no code, and each team could use any data, and any data mining algorithm that they wanted.

Because of the lookup table, any team, or individual could contribute an algorithm and have it deployed to production. Because each expansion was a physical row, we could even score the performance of specific line-item expansions. We were also able to mine the algorithms and their click through and conversion rates, and produce "meta algorithms" that combined the best expansions from all of the algorithms, based on how users responded to them.

The system was coded into adCenter's Delivery Engine and released in July 2006. We then experimented with algorithms as we attempted to get the best result in terms of revenue (value to adCenter), Clickthrough rate (value to users interacting with the system), and advertiser conversion rate (value to advertisers).

As we ran tests, awareness of the system grew, and more and more algorithms rolled in [32]. GMs (General Managers) started complaining that all of their researchers were no longer working on their projects – they were now all working on Smartmatch algorithms!

Eventually we would test 56 algorithms from 9 different teams, all in a very short space of time. The best algorithm was eventually sent in from a talented team from Microsoft Research Asia, and lifted revenue, CTR, and conversion rate, so it was a positive for all users of the system. The algorithm is described in [40].

We reviewed the algorithm with Microsoft's revenue governing board, and got the okay to increasing flighting to 98%, with control groups maintained.

The net result of this algorithm was an increase of revenue to adCenter of around $30 million per year, plus improvement in user clickthrough rate and conversion rate.

After deploying the system, we continued to make refinements and produce about 30 million additional revenue per year over the next 3 years. Afterwards we handed the system to the Revenue and Relevance team within Microsoft who continued to build and refine the system. I wrote a KDD paper on the system in 2008 [42].

V. Meanwhile Click Fraud Enters an Infinite Loop

The success of Smartmatch contrasted with efforts around Click fraud. The Legacy Click Fraud system was proving difficult to replace.

I had a small group of developers working on implementing a system which would extract features from

[1] Prior to Microsoft I had been working on automated bidding systems for online search auctions [22]. Because we had built our own click and conversion web servers and routed our Google and Yahoo ad traffic through them, we had observed a discrepancy between our server counts and the traffic being removed from search engines – specifically the search engines were under-billing about 15% of the traffic. I reported my observations in [23] and speculated that this appeared to be due to their clickfraud filtration systems. Little did I know that I would soon be on the front lines of the battle against this problem.

[2] Microsoft had originally pursued Overture as an acquisition target, as it was already using its technology to monetize its search engine as it began to compete seriously against Google. adCenter was originally conceived as "Plan B" – a contingency plan in the unlikely event that the acquisition fell through. As it turned out, Yahoo beat out Microsoft to acquire Overture, leading to the directive at Microsoft to "build it". With a very small number of developers, the first version of adCenter went live on September 2005 [35], [36].

adCenter weblogs, and then pass new weblog requests through a C4.5-like decision tree algorithm to create a prediction as to whether the impression was bot-generated [37], [21]. We started up a Click Fraud Investigation team, and enlisted their help to comb through weblogs and identify click fraud attacks, and then provide those as labeled cases for our system which we called "Clickfraud Bot Signatures" [28]. This was a "classic" machine learning approach in which a set of labeled cases are used to train the system to recognize attacks.

In order to test the system, we re-played weblogs against the system so that we could measure how many True Positives versus False Positives were being identified [2], [3].

One of our key discoveries at this time was that we could predict the conversion rate of traffic with a fairly high degree of accuracy. We would use this capability to help predict the commercial intent of the traffic, in addition to the likelihood of the traffic being fraudulent or bot generated.

However re-playing weblogs was as far as we were getting. Although the system was performing well, we were having trouble getting our code into production.

In Microsoft, Program Managers have a separate organizational tree from Developers, which eventually meet up under a General Manager. These developers reported to a Development Manager who was responsible for generating reports for advertisers as well as internal reports for Business teams.

However, for some reason, the Development Manager had no interest in deploying the code. We undertook endless code reviews, architecture reviews, performance reviews. I became frustrated by the lack of progress and the strange infinite loop of roadblocks.

After the space of years, I've now come to understand better what was happening here. The Development Manager was being pressured to produce reports in an operational system – that was his pain point. We were offering no solution to his number one problem. At the same time, he was concerned that we were asking him to deploy new, complex code, which he would have to allocate resources to fix and maintain over time. This was a losing proposition for the Development Manager. From his perspective, he had an incentive to allow the work to continue in "incubation" stage, but not to deploy it and assume the considerably higher costs of maintaining a completely new system.

VI. Fraud Explodes

While this was going on, we experienced our first taste of what fraud could do to a live revenue-generating system. A lot of advertisers had discovered a security flaw and were engaged in siphoning off revenue, and created side-effects on auction dynamics including impact to relevance.

Soon we were looking at a graph showing (roughly) 1%, 2%, 4%, 8%, 16% of revenue being lost to fraud. Everyone gasped with the realization that it was exponential.

Overnight fraud became the top – in fact the only - priority. Our mission was literally to save adCenter from large numbers of fraudsters who were stealing large sums of money and wrecking our system while they were at it.

Within days we had deployed adjustments into Delivery Engine to slow down the speed of attack –"stemming the bleeding". In less than a week we had built from the ground up the first version of Fraud Workbench, a major new system for automatically detecting account fraud using rules, with links into Delivery Engine for suspending accounts, slowing delivery of suspect accounts, and surfacing suspects to human reviewers for final determination.

Attackers adapt, and many of our temporary fixes were quickly overcome. However after releasing Fraud Workbench the impact was dramatic. Within two weeks we had paused most of the fraudulent accounts and they were being picked off by the human fraud team. Despite attempts to continue to attack, fraud was now proving ineffective and its impact on revenue dropped to undetectable levels.

The systems we built during this attack were foundational and were upgraded and expanded over the next several years. This was a great lesson in what would happen without an investment in protection.

VII. The "Bank Shot"

Despite success deploying Smartmatch and responding to fraud outbreaks, our new Click Fraud systems were still stymied.

In 2007 a new Developer lead joined the Delivery Engine team. I had earlier deployed Smartmatch through the Delivery Engine with enormous success.

As a result of this work, I had good relationships with the Delivery Engine teams, and a track record and reputation for deploying good, high performance, working Delivery Engine code. This would now help me to achieve an important deployment.

The New Dev Lead was tasked with re-architecting the Delivery Engine architecture to reach unlimited scale. This was an ambitious undertaking. However, before biting off complete re-architecture of critical operational systems, he was looking for a smaller application that he could productize through his architecture, which would be used as a "technology demonstrator" in order to work out the bugs in the new system.

We had already been working on code for a machine-learning based quality scorer, had been running simulations against weblogs, and had even been running performance load testing on the system. But we'd been blocked trying to deploy it to BI / Reporting.

Now came "the bank shot". The most obvious way to develop a click scoring system would be to build it as a standard weblog mining system.

But what if we could take this same system that was running on weblogs, and have it score live web traffic. After all, the weblogs were just a record of the HTTP requests that were sent to the ad server – why not expose the system to the real thing – the actual HTTP requests as they came in?

The idea wasn't as crazy as it sounded. The vast scale of web traffic being logged (eventually 17% of US Search traffic – a large chunk of the internet's traffic) was being handled by the Delivery Engine in real-time as it served ads from geographically distributed, co-located servers. Processing this traffic again offline would mean saving and re-loading the same data again, and issuing results for that same traffic in just thirty minutes (adCenter has to state reports every hour because advertisers use automated biddings systems to maximize their ROI on the auction also, and so need almost real-time information on conversion performance). Because of the speed requirements, most storage would have distributed and in-memory in either architecture. Why not cut out the store and re-load and deploy it inline as it was happening?

There were even reasons why this would work better than an offline system. By invalidating traffic during ad-serving, we could render advertisers immune from otherwise devastating real-time attacks[3] [24], [26]. This design would also give adCenter protection against Distributed Denial Of Service (DDOS) attacks that might be designed to try to take down the entire system.

Implementation in Delivery Engine was still not easy. The performance requirements for operation in Delivery Engine were the most extreme of any adCenter system. Scoring would need to execute within miliseconds on tens of thousands of requests per second. The system would need to be modified to utilize the distributed memory online-store that the Dev Lead had architected. On the plus side, our system had been exhaustively tested including performance testing – this was one of the benefits of getting held up in a testing infinite loop!

Work began around 2007, and on July 2007 we rolled out the new code and had a tense period in which we validated that it was in a good state. Some days later as part of a planned, staggered deployment, we then switched on the system.

It worked.

We had a new machine learning system autonomously detecting fraudulent traffic and telling downstream systems not to bill and not to deduct budget.

The main factor which led to success was the opportunity created by a new Dev Lead who needed to demonstrate the success of his new architecture, and so his incentives were firmly aligned to create a successful deployment with a "test" application such as ours. In turn we would help them to work out the bugs of their architecture, and we would be the "guinea pigs" for the new architecture.

The Dev Lead's architecture later became the blueprint for all of DE, and all of DE systems were eventually switched over to it. The Fraud system remained the canonical Online Store application that the developers used when testing and teaching new developers about how the system worked.

[3] The WOW Botnet [11], [12], [14] exploits latencies to create revenue. There are many other examples in which delays updating the auction creates room for an attacker to start generating revenue.

VIII. The Conversion MarketPlace

In parallel there were some other efforts underway on the problem that went beyond technology.

Firstly there was an industry-wide effort to address the problem. Starting in 2006, the IAB (Internet Advertising Bureau) in conjunction with the MRC[4] (Media Ratings Council) initiated secret meetings with Look, Ask, Microsoft, Google, and Yahoo to discuss the development of click measurement standards[5]. I was privileged to be the Microsoft representative in these talks and on the development of the click measurement standards that followed.

Also in adCenter we attempted to address the problem in a new way that hadn't been done before. We believed that there was a fundamental problem of incentive in click counting. The problem is this:

If the ad engine has the responsibility of both counting clicks, and billing clicks, then on the surface it appears to have an incentive to count too many clicks, ie. count too few True Positives – or make too many Type II errors [31].

But at the same time, a host of companies had sprung up to work on behalf of advertisers to look for fraudulent traffic and ask for their money back, in return for a bounty. Their incentive, unfortunately, was to flag everything as fraudulent to maximize their potential bounty; ie. count too many False Positives or make too many Type I errors, regardless of the smallness of the probability.

The advertiser was caught in the middle of this, and none of the proposals for Click Fraud seemed to deal with the incentive alignment problem.

We had an idea which might get everyone working on the same team, and effectively put incentives in place to reduce Type I and II errors. The solution was to "open it up" to a market. We proposed having a system score the quality of traffic, and then inform the advertisers what the quality of the traffic was. The advertisers could then load their bids on the different levels of quality. For example, traffic that was formerly "low quality" and was given to the advertiser for free, would undoubtedly contain some human traffic. The proposal was to by default opt advertisers out of the low quality traffic, but if an advertiser wanted to bid on the traffic, they could specify their price (including potentially zero). Bidding on low quality traffic was similar to trading on junk bonds – higher risk but workable for a low enough bid.

This was a real-time traffic quality or conversion rate market. Advertisers could decide what level of quality that they

[4] The Media Ratings Council (MRC) is a Congress mandated not-for-profit organization that governs the development of ad ratings and traces its lineage to the 1960s. They were founded in the 1960s because of alleged attempts to influence ratings so as to inappropriately change TV spot prices. Ratings companies such as Nielsen are now required to issue a public Description of Methodology document that describes the counting methodology, and they are audited each year by accounting firms, which audit them against the MRC industry standards. If they pass the audit they retain their MRC certification.

[5] I was also privileged to able to author some sections of the Click Measurement Guidelines. For example "refractory period" is one of my creations (apologies!).

wanted, but the proviso was that if they opted out of lower quality traffic, they wouldn't get it for free and other advertisers could bid on it.

An added twist was to allow third party companies (the same ones who were flagging everything as fraud) to supply the click quality engines via our Delivery Engine API, taking the ad network out of the business of assessing click quality, and opening up a business for third parties to plug in scoring modules that would provide information about the traffic and be paid by advertisers who could independently contract with the engines to help them identify the best traffic. This was a market-based solution to the problem of click fraud that would create incentive alignment with all parties.

Throughout 2006 and 2007 I promoted this idea. However, despite support from engineers, the business model sent adCenter into unknown territory. No other ad network used such a system or had ever tested such a system. Concerns were raised about complexity for advertisers (which we addressed by simplifying the system as much as possible, resulting in three quality bands), the possibility of sophisticated advertisers being able to use bidding to obtain better traffic, with less sophisticated advertisers getting worse composition of traffic.

The Click Quality Scoring System package we deployed in 2007 classified the traffic into conversion rate bands and we also loaded bid capabilities into Delivery Engine & were able to control bidding based on quality.

However, attempts to get the GUI team to expose the control proved futile. Several people in the Business team lined up against exposing the feature because of concerns above. In October 2008 my final attempts to expose the GUI were not prioritized based on input from certain business representatives and the initiative killed. Nearly three years of work trying to deploy a different take on the click fraud problem came to an unfortunate end, and our hopes for changing the click fraud problem fundamentally disappeared – it was one of those good ideas, right up there with flying cars.

Today real-time exchanges are common-place [1]. However opening up conversion levels or commercial intent to advertisers remains an area of untapped potential, and is a capability for which a sophisticated company with a lot of traffic and visibility into conversion events, like Microsoft or Google, would be ideally positioned to offer to advertisers [24].

IX. MINERVA DEPLOYS

Although we had a setback with Conversion Marketplace, the functionality in the online fraud system had been a huge success, and we finally had sophisticated fraud detection operating at a massive scale.

However the BI system which was the heart of weblog processing, was still operating a Legacy pipeline and really a mess of logic. It would be ideal if we could get BI overhauled and working in concert with the Delivery Engine system, and bring grid computing resources to bear to find the most sophisticated attacks. Low frequency, distributed IP attacks in particular needed really vast tables of data and was ideal for being analyzed via a distributed grid architecture. We finally got our chance.

The former BI Development Manager who had blocked work was re-assigned, and a new Development Manager took his place. The new Development Manager's job was to re-platform adCenter's data processing pipelines and make them scale for the large expansion of Microsoft traffic due to the acquisition of Yahoo by Microsoft.

The new Dev Manager first of all needed to develop the computing infrastructure necessary to process all of the data. After this there were two major "products" of BI. The first which was termed "Content Fact Repository" and was a set of joined, cleaned, enriched data, which would form that source for all downstream reports. The second "product" was the myriad of Content Reports that were required for advertisers, publishers, and internal teams. Traffic quality and the decision to "bill" or "not bill" would ultimately need to be a part of the Content Fact Repository, so that all downstream Content Reports would have a consistent picture of what was billable or non-billable.

We recognized an opportunity. Up until now, the fraud systems had been "just another thing to build". But if we could take ownership of the construction of the CFR itself, we could put some serious resources against traffic quality. This was a win for the Dev Manager who needed to build the CFR. As a result, we increased our scope significantly, and took ownership of this component.

This created a sea change in our area. No longer were we just another party asking for features. We were now critical for achieving BI's success – everything we did directly affected the success of billing, revenue recognition, reports and so on.

The increase in scope also made sense technically. Most filtration problems were due to instrumentation problems, dimension problems or other technical issues that was causing the traffic to be unbillable. We had extensive expertise in debugging adCenter's weblogs at a deep level since we'd spent so much time building systems looking for anomalies on them. In addition we had also deployed multiple systems in Delivery Engine and were responsible for a lot of the detailed traffic information that were being dropped into the weblogs. Therefore we were the ideal people to understand, analyze and fix issues. Everything ultimately was debugged through our team.

This ownership allowed us to build a well architected Minerva offline traffic quality module, with vastly greater computing resources than ever before – a several thousand machine distributed COSMOS grid developed by Microsoft with similar functionality to Hadoop.

adCenter's BI system transitioned to the new architecture in February 2008. The first version was a direct re-platform, and no new features, so that we could measure any data discrepancies. After re-platforming, we iterated on the Minerva system every release. In 2009 we were able to release a plethora of features including Minerva module with multi-model support and rapid deployment capabilities. We finally had our deep computational infrastructure for weblog processing.

X. SMARTPRICING DEPLOYS

Remember our great idea of the Conversion Marketplace? In August 2008 adCenter was ready to venture into self-service publisher networks.

There was only one thing stopping adCenter from incorporating self-service inventory: Smartpricing.

In Smartpricing an ad server automatically provides a price discount based on probability of conversion of the traffic [15]. This enables the ad network to monetize lots of lower quality inventory without the advertiser incurring any loss of performance due to the lower quality nature of the third party sites. Smartpricing had been deployed by other ad networks such as Google and Yahoo and was considered minimum requirements for entry into self-service.

Guess what could system used machine learning methods to predict conversion rate of every impression in adCenter and alter bid prices based on the traffic quality? ....and what system was already fully tested and running in Delivery Engine? What was Smartpricing anyway, but the Conversion Marketplace, but where the advertisers were having their bid adjustments set on their behalf?

The changes were small. Quality bands were literally the leaves on a decision tree that was predicting traffic quality. Rather than output traffic quality bands (eg. 1, 2, 3 etc) and reading advertiser bids, we switched them to output an exact price discount percentage as the leaves of the decision tree.

The Business partners who lined up against Conversion Marketplace now lined up in vigorous favor of Smartpricing. This was a great lesson in why maintaining personal and professional integrity was important.

Price adjustments were switched on. We could now trade across self-service networks.

XI. TRAFFIC QUALITY SYSTEMS AS FAR AS THE EYE CAN SEE

By the end of 2009 we had an end-to-end system with (a) ARTEMIS architecture responsible for real-time quality scoring (b) Smartpricing executing real-time price discounts (c) MINERVA system determining final billability of all traffic (d) Conversion tracking system, (e) Fraud workbench to help identify account fraud, (f) Blacklist capture system, (g) Crawlers, (h) Packet sniffers, and many other systems, working end-to-end to detect, discount and nullify bad traffic [4], [25], [26].

We had our own product-copy automated model testing infrastructure, endorsed by our Testing team, that ensured that new models could be deployed out of release cycles. This meant the fraud model updates could be deployed whenever needed, and yet with complete safety; critical for a 24/7 uptime system. We created weekly meetings to review model releases with the Business, and were able to be responsive to requests from the Business to address new issues.

Meanwhile, the secret talks that we began in 2006 eventually paid dividends. We jointly issued the Click Measurement Guidelines in 2009 and are now audited by accounting firms on our click/impression billing processes [18], [19], [20]. This also happens to be the same system and certification system that is used for television ratings. We announced that our systems were MRC certified in 2009, which was the earliest agreed date for public release of the standards and results.

XII. WHAT WORKED AND WHAT DIDN'T?

Large system development is challenging not only for technical reasons, but also because large numbers of different stakeholders and groups need to be engaged, and need to work in concert with each other. At Microsoft groups include (a) Product Team, (b) Multiple Engineering Teams (Delivery Engine, GUI, BI), (c) Business teams, (d) Program Management, (f) Support teams (three levels: 1, 2, and 3), (g) Help and Documentation, (h) Release Management, (i) Security, (j) Privacy, (k) Marketing Communications, (l) External groups including LCA's Crimes Unit, Malware and viruses group, Bing Search proper, and so on.

All of these teams are under their own organizations. Having a large team means that great things can be accomplished. But it also means that any individual anywhere in one of these parallel teams can stop progress. A release needs all groups to agree to work and complete their work. Work can be stopped at any point.

At various times we worked on skunk works projects. For example, ARTEMIS was moved to deployment readiness by a group of developers working without a mandate to deploy, and that cut deployment time and increased our quality, safety and performance as we had more time with the system. However the greatest successes that we had was when we were able to find alignment in incentives between my objectives and those of my partners. When this occurred, we were able to find dozens of developers and managers all queued and eager to deploy a system.

Below are lessons that I believe would be useful for practitioners to consider:

1. **Make it easy for other teams to work with you:** For example, a well-defined APIs was the key to engaging multiple teams in Smartmatch. We set up weblog access systems so that our ClickFraud investigation team could work on cases, and then asked the human investigators to catalog and tag bot cases. Our ClickFraud Bot signatures were simply a copy of the weblogs – making it easy for them to save them and for us to use them.
2. **Expose the data:** Expose data to partners and other teams. Quality Systems in adCenter made available our output to vast numbers of partners to use. This brought to bear large numbers of users who could report anomalies using this data.
3. **Do What's Right even if it takes you far afield:** Smartmatch wasn't what I was hired to do, but had enormous benefits to adCenter. It was a huge win. It also enabled me to deploy several other significant Delivery Engine systems.

4. **Partial systems are better than none at all:** Getting an organization to make a big investment and big commitment is very hard and often is too much. However, if the pieces are architected well, you should be able to build small pieces which are themselves useful. At various times I had offline systems without online, and then vica versa. By keeping the items modular, small, and independently useful, I was able to keep them above the prioritization cut-line, and then fold them into a a much bigger and more capable system. Development Managers often joked that I was building huge stealth projects in adCenter, one small, innocuous, bolt at a time.

5. **"The System" Includes Humans!** Always keep in mind that "the system" that you're deploying includes humans, processes, and methodologies. In order to be very accurate, we had to staff and train a Clickfraud Investigation Team on how to read and understand weblogs, and recognize and tag bot cases so that we could train against them. In order to deploy models to production, we needed to gain agreement with multiple teams including Security, Program Management, Release Management on a rapid release methodology that ensured automated testing of our releases so did not put the platform or advertisers at risk but yet allowed us to respond quickly to new attacks.

6. **Understand other peoples' incentives:** Understanding other peoples' incentives will avoid you wasting time on proposals which have no chance of deployment.

7. **Solve your partner's problems:** When you're in an organization you work with a lot of different people. Don't just view those people as "obviously having to work on your project", even if their mandate appears to be the case. For each of those people, try to understand what problem they need solved, and see if you can help them to solve it. This same advice goes for your employees, external teams, bosses, and so on – ask yourself "what problem am I solving for these people"?

8. **Never let a crisis go to waste:** Crises are bad for everyone – but if you find yourself in the middle of one, it is also the best time to ensure that the resources are put in place so that it never happens again.

9. **Look for non-obvious solutions:** Look for ways around roadblocks. Deploying to Delivery Engine wasn't the obvious path, but resulted in a better system.

10. **Communicate:** Telling people about the systems that you're building can open doors. In the case of Smartmatch, our work captured the imagination of our General Manager as well as research teams, and we found ourselves scores of people working on our problem. Yet at the same time, communication will only be successful if you've also done the hard work of figuring out how to align incentives.

11. **Maintain Professionalism and Personal Integrity:** Always maintain integrity with different stakeholders. Integrity means that you do as you say. Professionalism means maintaining the composure and disposition to always solve problems and focus on what can be achieved. There were several occasions in which blocking stakeholders in one project, assisted us months or years later (eg. Quality Bands was blocked, but the same stakeholders were strong partners who advocated for Smartpricing).

12. **Don't give up:** Big systems mean they may take longer to reach production.

Large-scale systems development is not something that can easily be taught from a textbook, as nothing other than the real-world will have a mixture of entities and interests. However it is possible to provide heuristics like these to help conduct the project in a way that gives it a higher chance of success. Data Mining Practitioners need to engage in large developments with a mixture of integrity, positivity, persistence and ingenuity, and doing so can move mountains.

ACKNOWLEDGMENTS

It took a large number of talented people to build adCenter's Fraud detection, CFR Data Processing, ClickFraud, Smartpricing, Conversion tracking systems. Going beyond the call of duty was the norm at adCenter, and because of the efforts of the people below, we were able to create what may be the largest and most sophisticated system of its type in the world. At a personal level, my fondest memories are of the people that I worked with, and I have the greatest respect for all of them. After five years at Microsoft I wish to express my sincere thanks and gratitude to Jing Ying Zhang, Gang Wu, Wesley Brandi, Julien Beasley, Kieran Morrill, John Ettedgui, Sid Siddhartha, Hong Yuan, Feng Gao, Peter Azo, Raj Mahato, Albert Roux, Ron Mills, Brandon Sobottka, Matthew Rice, Sasha Berger, Jigar Mody, Dennis Minium, Kamran Kanany, Tudor Trufinescu, Dinesh Chahlia, Ken Pierce, Hank Hoek, Tao Ma, Karl Reese, Narayanan Madhu, Dimitry Berger, Rageesh Maniyembath, Meena, Joseph Morrison, Kiran Vemulapalli, Anthony Crispo, Matthew Bisson, Igor Chepil, Matthew Ford, Sachin Ghani, Amjad Hussain, Steve Marlar, Bill Morency, Gerry Moses, Steve Sullivan, Brian Burdick, Ian Ferreira and many others. None of this would have been possible without you, and the work that you did has made a major difference to online ecommerce.

REFERENCES

[1] Jody D. Biggs, Brett D. Brewer, Brian Burdick, David Max Chickering, Christopher Andrew Daniels, Ewa Dominowska, Gary W. Flake, David Jakubowski, Brendan Kitts, Christopher Meek, Yusuf I. Mehdi, Tarek Najm, Richard Liam Pelly, Yosha R. Ulrich-Stumat, Lightweight and heavyweight interfaces to federated advertising marketplace, US Patent, 2007.

[2] Buehrer, G., Stokes, J. and Chellapilla, K. (2008a) A Large-Scale Study of Automated Web Search Traffic, Proceedings of the 4th international workshop on Adversarial information retrieval on the web (AIRWEB) 2008, April 22, 2008. http://research.microsoft.com/apps/pubs/?id=69505 (2008a)

[3] Buehrer, G., Stokes, J. Chellapilla, K. and Platt, J. (2008b) Classification of Automated Web Traffic, in Weaving Services and People on the World Wide Web, Springer Berlin Heidelberg

[4] Brandi., W., Kitts, B., Mahato, R., Zhang, J. (2013), Armored Ads, 2013 IEEE International Conference on Intelligence and Security Informatics (ISI), IEEE Press. Seattle, WA.

[5] S. Clifford, Microsoft Sues Three in Click-Fraud Scheme, NYTimes.com, June 15, 2009, http://www.nytimes.com/2009/06/16/business/media/16adco.html, 2009.

[6] Checkmate Strategic Group vs Yahoo! Inc., Case No. 2:05-cv-04588-CAS-FMO, Amended Class Action Complaint, http://www.dmnews.com/yahoo-settles-with-checkmate-strategic-group-lawsuit/article/91798/

[7] Crafts by Veronic vs. Yahoo! Inc. Class Action Complaint, Case Number 2:2006cv01985,

[8] Cranton, T. (2009), Using Enforcement to Crack Down on 'Click Fraud', Microsoft on the issues website, http://microsoftontheissues.com/cs/blogs/mscorp/archive/2009/06/15/using-enforcement-to-crack-down-on-click-fraud.aspx, 2009.

[9] Crawford, K. (2004), Google CFO: Fraud a big threat, CNN Money, December 2, 2004, http://money.cnn.com/2004/12/02/technology/google_fraud/

[10] Daswani, N. and Stoppelman, M. (2007) The Anatomy of ClickBot A, Usenix, HotBots 2007, http://static.usenix.org/event/hotbots07/tech/full_papers/daswani/daswani.pdf

[11] Daswani, N., Mysen, C., Rao, V., Weis, S., Gharachorloo, K. and Ghosemajumder, S. (2008) Online Advertising Fraud, in Jakobsson, M. and Ramzan, Z. (eds), Crimeware, Symantec Press http://shumans.com/onlineadvertisingfraud.pdf

[12] Delaney, K. (2005), In 'Click Fraud,' Web Outfits have a costly problem, The Wall Street Journal, April 6, 2005, http://online.wsj.com/article/0,,SB111275037030799121,00.html

[13] Edelman, B. (2006) The Spyware - Click-Fraud Connection -- and Yahoo's Role Revisited, April 4, 2006, http://www.benedelman.org/news/040406-1.html#e1

[14] M. Fichman and J. Cummings, Multiple Imputation for Missing Data, Tepper School of Business. Paper 113, 2003.

[15] Google AdSense, About Smartpricing, http://support.google.com/adsense/bin/answer.py?hl=en&answer=190436, 2013.

[16] Google vs Lanes Gifts, http://adwords.blogspot.com/2006/05/lanes-gifts-v-google-settlement.html

[17] B. Jansen and T. Mullen, Sponsored search, International Journal of Electronic Business, Vol. 6, No. 2, 2008.

[18] IAB/ABCe International Spiders & Bots List, Internet Advertising Bureau, http://www.iab.net/iab_products_and_industry_services/1418/spiders

[19] IAB (2005), IAB Impression Measurement Guidelines, http://www.iab.net/media/file/US_meas_guidelines.pdf

[20] IAB (2009), IAB Click Measurement Guidelines, http://www.iab.net/media/file/click-measurement-guidelines2009.pdf

[21] Kitts, B., (1997), Regression Trees, unpublished. Available from

[22] Kitts, B. and LeBlanc, B. (2004), Optimal Bidding on Keyword Auctions, Electronic Markets, Vol. 14, No. 3, pp. 186-201.

[23] Kitts, B., LeBlanc, B., Meech, R., Laxminarayan, P. (2006), Click Fraud, Bulletin of the Association for Information Science and Technology, Vol. 32, Iss. 2., pp. 23-24. January 2006, American Society for Information Science and Technology

[24] Kitts, B. et. al. (2006), Click Fraud Protector, Application number: 11/559,291, Publication number: US 2008/0114624 A1, Filing date: Nov 13, 2006 http://www.google.com/patents?id=KpaqAAAAEBAJ&zoom=4&dq=click%20fraud%20protector&pg=PA1#v=onepage&q&f=false

[25] Kitts, B., Najm, T., Burdick, B. (2007), Identifying Automated Click Fraud Programs, US Patent Publication Number US 2008/0281606 A1

[26] Kitts, B., Zhang, J., Wu, G., Brandi, W., Beasley, J., Morrill, K., Ettedgui1, J., Siddhartha, S., Yuan, H., Gao, F., Azo, P., Mahato, R. (2014), Click Fraud Detection: Adversarial Pattern Recognition over 5 Years at Microsoft, in press.

[27] Kitts, B., Zhang, J., Wu, G., Mahato, R. (2013), Click Fraud Botnet Detection by Calculating Mix Adjusted Traffic Value: A Method for De-Cloaking Click Fraud Attacks that is Resistant to Spoofing, 2013 IEEE International Conference on Intelligence and Security Informatics (ISI), IEEE Press. Seattle, WA.

[28] Kitts, B., Zhang, J., Roux, A., Mills, R. (2013), Click Fraud Detection with Bot Signatures, 2013 IEEE International Conference on Intelligence and Security Informatics (ISI), IEEE Press. Seattle, WA.

[29] Kohavi, R., Mason, L., Parekh, R., Zheng, Z. (2004), Lessons and Challenges from Mining Retail E-Commerce Data, Machine Learning Journal, Special Issue on Data Mining Lessons Learned.

[30] Samuel Lassoff vs Google Inc. Class Action for Residents of NY and NJ, August 18, 2006

[31] Marson, I. (2005), Search Engines accused over click fraud, ZDNet, April 6, 2005, http://www.zdnet.com/search-engines-accused-over-click-fraud-3039194000/

[32] Donald Metzler, Susan Dumais, Chris Meek, (2006), Similarity Measures for Short Segments of Text, http://team/sites/Broadmatch/Shared%20Documents/MetzlerDumais MeekECIR07-Final.docTuzhilin, A. *The Lanes Gifts vrs Google Report*, 2006, http://googleblog.blogspot.com/pdf/Tuzhilin_Report.pdf

[33] Microsoft, Microsoft Corporation vs Eric Lam et. al., Civil Case No. C09-0815, Complaint for Injunctive Relief and Damages, United States District Court Western Ditrict of Washington at Seattle, June 2009.

[34] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, Numerical Recipes: The Art of Scientific Computing, NY: Cambridge University Press, 2007.

[35] Olsen, S. and Kane, M. (2003), Stefanie Olsen and Margaret Kane, Yahoo to buy Overture for $1.63 billion, CNet News, July 14, 2003, http://news.cnet.com/2100-1030_3-1025394.html

[36] Olsen, S. (2003), Yahoo-Overture: Wake-up call for Microsoft?, ZDNet News, July 15, 2003, http://www.zdnet.com/news/yahoo-overture-wake-up-call-for-microsoft/130401

[37] Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

[38] Stitelman, O., Dalessandro, B., Hook, R., Reader, T., Provost, F. (2013), Using co-visitation networks for detecting large scale online display advertising exchange fraud, Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1240-1248

[39] Tuzhilin, A. The Lanes Gifts vrs Google Report, 2006, http://googleblog.blogspot.com/pdf/Tuzhilin_Report.pdf

[40] J. Wang, H. Zeng, Z. Chen, H. Lu, L. Tao, W. Ma, "ReCoM: Reinforcement Clustering of Multi-Type Interrelated Data Objects", In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'03), pp. 274-281, Toronto, Canada, July 2003. http://team/sites/Broadmatch/Shared%20Documents/p16477-wang.pdf

[41] L. Whitney "Bing's U.S. search market share continues to climb", CNet News, June 12, 2013. http://news.cnet.com/8301-10805_3-57588873-75/bings-u.s-search-market-share-continues-to-climb/ 2013

[42] G. Wu and B. Kitts, "Experimental comparison of scalable online ad serving", Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2008), pp. 1008-1015. 2008.