Asia-Pacific
Economic Cooperation

Datamineware Technology Inc.
Canada

Human Resource Management in a "Red Queen" Game

Written by

Charles Gastle
Partner
Bennett Gastle Professional Corporation Barristers and Solicitors

Heather Gastle
Law Student
Bond University

> **"***Well, in our country," said Alice, still panting a little, "you'd generally get to somewhere else if you ran for a long time as we've been doing." "A slow sort of country!" said the Queen. "Now, here I see. It takes all the running you can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!"*
>
> Lewis Caroll, *Through the Looking Glass*[1]

In January 2012, John Denning was concerned that DATAMINEWARE[2] was taking too long to release new software and functionality to the marketplace. Its market positioning was dependent on the release of software program updates every three months to outpace its competition. Denning had since changed the process of software development but organizational problems threatening customer service persisted and he faced the imminent loss of one of his key software developers.

Company Background

A "Red Queen" game is one in which a small and medium-sized enterprise (SME) has to exceed the pace of its competitors. DATAMINEWARE Technology Inc. (DATAMINEWARE) was such an SME as it had to stay ahead of a number of large multinational competitors.

DATAMINEWARE was founded by John Denning in Vancouver, Canada in 2003. By 2011, the company had grown to C$35 million in annual net sales and established its predictive software as the standard in its market niche. It competed with large Management Information Systems (MIS) corporations in providing software that analyzed data collected during the clients' "just-in-time" manufacturing process. DATAMINEWARE, with its plug-in features to any existing MIS software, outpaced its competitors by continually providing new and innovative functionality. It also designed custom functionality requested by a particular customer as each had his or her own preferences. This made DATAMINEWARE more sensitive to customer needs than the large MIS companies. As a result, excellent customer service was also important to its position in the marketplace.

Identifying the Problem

Initially, new functionality was released to customers on a timely basis. By 2011, the development cycle had lengthened by 30% to 50%. Even with this extended time period, some projects were incomplete and had to be held over for a second cycle. The developers worked individually and only revealed the lines of computer code that they had written which formed the software program at the end of the development cycle, when it was subjected to testing. Denning realized that the privacy he allowed his developers was counterproductive and one of the reasons why project deadlines were missed.

---

[1] Lewis Carroll, *Through the Looking Glass* (Dover Thrift Editions, 1999).

[2] Datamineware Manufacturing, Inc. was a fictitious company but the issues raised herein were based on a software company located in the United States of America in an industry different than that of data mining.

Denning believed that another reason the development cycles were lengthening was the growing complexity of the software products and functionality that DATAMINEWARE offered. When it was a start-up, writing software code was relatively easy and all the developers were “stars.” Over time, the programs evolved and inevitably, the code for each program had flaws embedded within it that impaired the functionality or caused the system to crash from time to time. The code also had peculiarities as a result of developing custom functionality for particular customers. These legacy issues could trigger problems in the new functionality that was being added in later development cycles.

Another problem was that the software lacked proper documentation embedded in the software code that comprised the computer program. Comments were inserted into the code to explain the logic that had been adopted by the developer. The comments were important for any subsequent developer to understand the manner in which the code was written.

Finally, so much software was written during the longer development cycles that there was not sufficient time to thoroughly test the code. This inability to test was causing customer service problems due to bugs and software instability, once the functionality was released to the customer.

Denning believed that the root of the problem was the way in which the development team was organized and evaluated. Denning had to increase the pace of software development while maintaining productivity and customer service.

DATAMINEWARE’s Development Team

DATAMINEWARE initially had a team of three software developers but by 2011, the team had grown to eight developers, all of whom were from abroad. It was difficult to find MIS developers in Canada and Denning had to hire them from China, India, Israel, Romania and Russia by advertising on the internet and through his development team. Often, they would identify friends who had MIS expertise from where they came from, and due to their unique skill-set DATAMINEWARE was able to arrange their immigration to Canada. Bringing developers from abroad did have disadvantages, though. First, these workers were highly mobile internationally, and many would leave for other local or foreign opportunities. Second, cultural demands placed on developers required them on occasion to return home. To overcome the turnover problem, Denning hired developers even when he did not need to do so because of the difficulty of finding them. As a result, he usually had more developers than he needed at any one time.

Denning found that he could keep developers for no longer than a period of two to three years as they appeared to have a need for novelty and eventually moved on to new projects. As an example, Denning lost a number of developers to social networking start-ups before Facebook emerged as a “killer” application. Some of them would eventually return but there were no policies that Denning could think of to overcome the two to three year shelf-life of a developer:

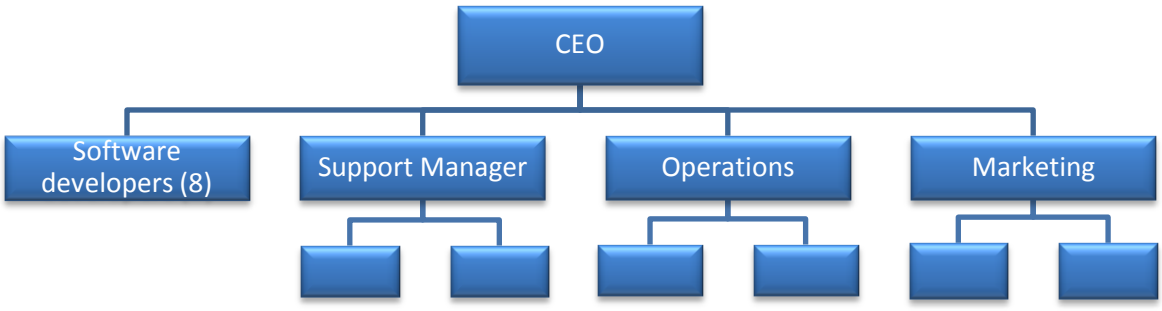
Duration of Employment	Number of developers
1 year	2
2 years	5
3 years	7
4 years	2
5 years	1

The threat of losing a key developer to one of the large MIS companies was a significant threat to DATAMINEWARE as some knew the software company's overall architecture, its functionality and its future development plans. Denning believed that he had done everything he could to keep his developers. He offered a flexible work environment and flexible hours, and paid them a salary that was competitive with what large corporations offered. They also received a discretionary productivity bonus up to 10 percent of their salary, depending on Denning's evaluation of the quantity and quality of the software code they had written. Denning did this subjectively, without any established criteria, and he sometimes gave higher bonuses to those he liked or wanted to keep. He knew that some companies kept their most productive developers by giving them an ownership interest in the company or a percentage of the profits. He considered these alternatives but Denning's corporate counsel informed him that according to Canadian law, minority shareholders had ownership interests sufficient to give them the ability to commence a lawsuit, if they felt that their interests were treated in an oppressive manner by the majority shareholder. He was also told that giving the employees a profit percentage carried with it the requirement to provide financial disclosure. Denning was not prepared to give up an ownership interest in DATAMINEWARE or make financial disclosure to his employees.

Changing the Organizational Structure

Denning had to admit that his own management style contributed to the problems at DATAMINEWARE. He had run the company as if it were a small start-up by controlling every function of the organization. He realized that the company might have outgrown him. From the time of its incorporation, DATAMINEWARE had maintained a flat management structure (Figure 1) in which the developers reported directly to Denning. He acted as his own Chief Technology Officer (CTO).

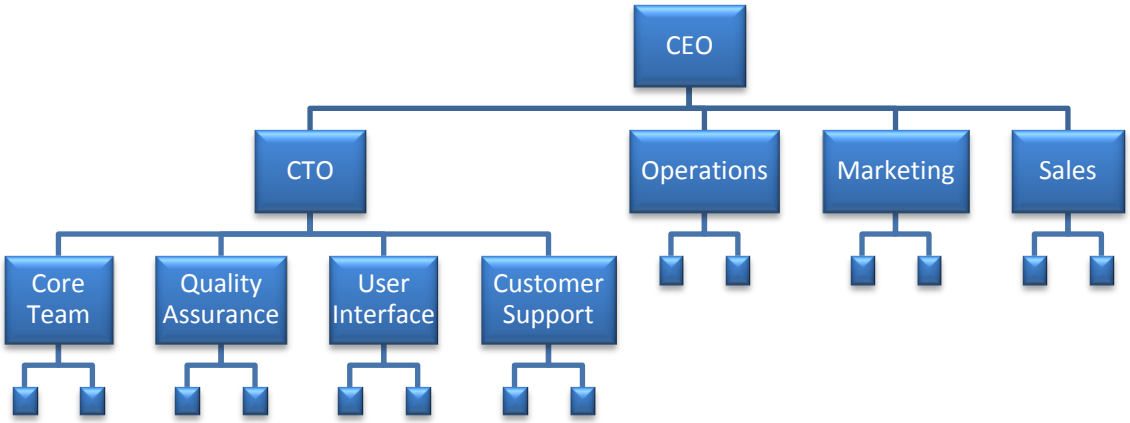
Figure 1: Original Management Structure



Denning decided to hire a CTO and put him in charge of the development team. He advertised for a CTO and hired one with significant experience in developing software through "agile software development" processes. He then redesigned the corporate

structure so that the developers reported to the CTO. The developers were split into different teams relating to core technology, quality assurance, and the graphic user interface. Team leaders were designated for each of these critical functions. He also decided to have customer support report to the CTO due to the fact that customer service issues would give rise to development projects that would have to be integrated into the sprints (Figure 2).

Figure 2: Redesigned Organizational Structure



Changing the Software Development Model

To accelerate the speed of software development, Denning studied agile software development concepts which promoted the rapid development of software in an iterative process. It provided that software development should occur in a series of sprints of no longer than one month in duration.[3] A daily meeting was to be held to synchronize the work of each team member into the overall project.[4] At the end of the sprint, the new features were integrated into the software program and tested. A review meeting was held in which the team demonstrated the new functionality that had been created during the sprint. Once it was tested and passed by quality assurance, it would quickly be made available to customers who would beta test the software and provide their feedback.[5]

Denning decided to implement agile software development and defined the sprint development cycle as a two-week period. One advantage of a short sprint was that the performance of the various developers could be measured on an on-going basis. If a project was not completed within the sprint, it would be held over, with a review undertaken as to why it was not completed on a timely basis. It would also be evident if the developer had poor skills or suffered from a lack of interest. This continual assessment provided a more regular measurement of the developers' work by which decisions could then be made to terminate the developer's employment if warranted.

Denning believed that a two-week sprint would also allow him to improve the testing of the software code and the documentation within it. Each sprint would produce a limited

[3] http://www.mountaingoatsoftware.com/topics/scrum, last visited August 14th, 2012
[4] Ibid.
[5] Ibid.

amount of code that could be quickly reviewed by Quality Assurance and then released to customers for beta testing. The software code could be sent back to the developer if appropriate documentation of the logic used in writing the code had not been included. Also, reviewing the code on an on-going basis would make the development team more familiar with its structure. Only two weeks of new code would be at risk if a developer left without warning. The results of the testing and the inclusion of proper documentation became key criteria in the evaluation of software developers.

Planning the sprints was left to a Product Development Team comprised of the Chief Executive Officer, Chief Technology Officer, the Support Manager and the Product Manager from the marketing department. This Team was responsible for the strategic management of software development and for breaking projects down into parts that could be completed within the two week sprints.

The CTO recommended that the programming language be changed from “C++” in which its software was written to “ASP.NET” language. The CTO explained that C++ was a desktop language whereas ASP.NET was a web language in which software code could be written more quickly. The CTO stated that the move to agile development could not be made successfully unless this change in programming language was made. Denning took time to consider this change. His developers were experienced C++ programmers but some of them had no experience in ASP.NET. Denning agreed to adopt ASP.NET but he was concerned about the impact it would have on the developers.

Communicating Change

Denning had made the commitment to change, but he knew it would be difficult and disruptive to the company’s operations in the short term. The challenge now was to communicate this commitment to the entire organization and ensure that all employees knew that there was no returning to the *ad hoc* culture that had evolved since DATAMINEWARE days as a start-up.

Denning and the CTO held a series of meetings with the developers as well as the customer service and sales teams so that all understood the changes in management and the implementation of agile software development processes. The employees were advised of the critical importance of staying ahead of the competition by being nimble and customer-oriented. They spoke to the developers about retraining in ASP.NET at the company’s expense. The developers were then sent for two-week training sessions on rotation.

Due to the risk that key developers might leave, Denning contacted his corporate counsel to review the existing employment agreement and determine whether DATAMINEWARE might obtain protection from its former developers joining the competition. Pursuant to Canadian law, developers could be prevented from taking a job with a competitor for a period of up to 12 months if the contract contained a non-competition clause. The former employee also had to keep confidential any matter relating to the architecture or other secrets of the code forever. Denning was advised that it was difficult to enforce non-competition and confidentiality clauses in foreign jurisdictions, particularly in an economy not known for a reliable and independent judiciary.

Denning also extended the period of notice a developer had to give to leave the company to take another job. The old employment agreements provided that a departing employee had to give two-weeks' notice. DATAMINEWARE had experienced problems two weeks after a key developer left, when the code the developer had written proved to be inscrutable and had to be abandoned. The notice period was extended in the new employment agreement to four weeks to provide enough time for the departing developer to explain the code written, and the logic on which it was based.

Denning had the development team sign the new employment agreements before they were sent for training on ASP.NET.

Impact of Change on the Development Team

The current state of the software was analyzed by the Product Development Team and projects were identified for the initial two-week sprints, which were quickly introduced. The first sprints were somewhat disorganized, with a number of projects not being completed. It took three months of adjustment before Denning felt that the sprints were working as intended.

Denning found that his developers fell into two groups in their response to the changes that were occurring. The first included those who were willing to embrace agile development process and retrain in ASP.NET. The second group was not ready to adopt agile development, did not want to retrain in ASP.NET or proved incapable of it. These were some of the top developers in the company who did not like to lag behind some of the younger developers writing ASP.NET. As Denning feared, these developers soon left the company causing further disruption by having to replace good talent and losing their knowledge of the code's architecture. Denning had to rely on the non-competition clauses contained in the employment agreements to protect DATAMINEWARE from the risk of having the departing employees join the competition.

Eventually, DATAMINEWARE was left with a team of developers that were ready to work with the new processes and were trained in the ASP.NET programming language, but some issues remained. Some developers were shy and would not speak up in front of others at the sprint meetings that were scheduled every day for the developers to discuss their projects. This made it difficult to monitor and forecast the progress of each task. To remedy this, developers had to be encouraged to speak up at meetings. Denning also held firm that the developers had to forget the old processes and adopt the new ones. During the initial adjustment period, the developers who used to report to Denning came and talked to him, asking to return to the old system and citing the preference for Denning's managerial style over that of the new CTO. Denning made it clear that the CTO had his full support and that this new structure was permanent. Denning found these problems were a type of growing pain, which dwindled over time. After a few sprint cycles had been completed and the employees had time to gain familiarity with the new processes, these problems were eliminated.

Employee Evaluation

Employee evaluation was an important tool in reinforcing the changes made. In the past, employees were given an evaluation once a year on an informal basis. Denning did not really like evaluating employees who had become his friends and he did not want to

confront them with their shortcomings. As long as the company continued to thrive, the employees received favourable evaluations without much criticism. As part of the changes made, Denning, the CTO and the other managers designed a comprehensive evaluation procedure that identified objective metrics. The criteria included, but were not limited to:

- Completing the task assigned within the two-week sprint (10 points);
- Participating in discussions during the daily sprint meetings (up to 5 points);
- Including comments within the software code
  - (1 point for each comment, up to 10 points);
  - (if code is properly described, up to an additional 10 points);
- Testing by quality assurance,
  - (10 points, less 1 point for each bug,);
  - (loss of all 10 points if code was unstable);
- Customers evaluation of code during Beta testing:
  - (rating on a scale of 10 by customer);

The importance of a customer orientation was underscored by including customers in the evaluation of the software code. The new system of evaluation was objective and measurable. Results of the evaluation showed an improvement in performance scores among those developers who accepted the changes and the new programming language. The scores were posted within one month of the end of the sprint so that all could monitor their performance relative to others within the group. This caused some dissatisfaction among the poorer performing developers and this was a problem because some of the top C++ developers were struggling with ASP.NET and it was a further incentive to leave.

An Unexpected Consequence: Customer Service Suffered

The biggest challenge Denning faced in implementing change to DATAMINEWARE was customer service. He commented that “after the first three months, I didn’t have a single happy client and eventually, I had to pay back C$100,000 in rebates to satisfy them.” He knew that due to the disruptions caused by the changes being made, the customer service employees were subject to confrontations by angry customers on a regular basis, which made their working lives very stressful. Understandably, it led them to reject the changes being made. Prior to the change, the customer service agents were able to contact developers directly, requesting them to work on solving a customer complaint. When Denning implemented the change to agile development processes, the customer service employees had to make requests to the CTO to whom they reported. They found that the CTO would make promises about dealing with specific customer complaints but that these would not be addressed. The CTO either failed to assign developers or those who were assigned could not take time away from their projects due to the tightness of a two-week sprint to complete the new code assigned to them. As a result, customer service employees could not react to customer complaints on a timely basis.

The customer service problems had a ripple effect on the sales team of DATAMINEWARE as well. This occurred because each customer had a relationship with their sales person and when customer service did not address the problem immediately, the sales person got the next call. As a result, a number of employees in the company had cause to reject the agile development processes that were being implemented and preferred the old system in which bugs would be fixed quickly and customers appeased.

Complimentary Change: Introducing ITIL

Agile development was not intended for other functions within the company. Denning was aware of processes known as ITIL (Intellectual Technology Infrastructure Library) which was initially developed by the British Government, and which eventually spread around the world to a variety of businesses. ITIL was designed for identifying, planning, delivering and supporting customer service.[6] Just as Denning had worked on implementing agile development processes to solve the problems in software development, he turned to ITIL principles to solve customer service issues.

Implementing ITIL required the introduction of a new customer flag system. The existing system was one that allowed the clients to fill out customer support flags which established a record of a particular problem they were experiencing. Each flag would then be assigned to a developer by the customer service team. A new flag system had to be created to allow ranking them in order of importance. DATAMINEWARE adopted a process indicating that if a flag was rated "high," it would have to be dealt with within two hours. If one was rated "medium," additional time was allowed for the problem to be dealt with. If it was rated "low," it could then wait and be inserted into the next sprint cycle. It again took approximately three months to implement the new customer service processes. Support staff had to be assigned to enter up to 100 flags into the new system to activate it.

But Problems Remained

Even after ITIL processes were introduced and the new flag system created, customer relations issues remained. Customers were still dissatisfied with the time it was taking to have their complaints dealt with. Denning learned that customer service fixes were not being handled within the time period established by the flag system. He found that the CTO was still giving priority to the development projects in situations where he felt that the development of new code was more important. The CTO defended his position indicating that this was necessary so that the new functionality could be developed on time, and that the flags that were delayed were not really that important.

Denning knew that customer service issues had to be given a higher priority in the development process. He knew that until he did so, there would be strong resistance to the changes being made among the customer support and sales teams. He also knew that the ranking of the new software code versus customer service projects had to be made with the input of different departments within the company. He knew that the corporate structure would have to be adjusted and the question was how this should take place and what new processes should be implemented.

Denning was also advised by another developer that Jim You Song was about to leave and join a competitor in Guangdong Province in China. Jim was Denning's good friend and a key developer who had defied the odds and had remained at DATAMINEWARE for the last five years. Jim was unhappy with ASP.NET which he was struggling with and he felt insulted when the evaluation scores were published as he was regularly in the bottom tier of developers. Denning did not want to lose Jim as he had been the

[6] "What is ITIL?" http://www.itil-officialsite.com/AboutITIL/WhatisITIL.aspx.

company's most creative and visionary developer responsible for some of the company's most popular software. Denning feared the damage that could be done if Jim joined a company in a jurisdiction in which non-competition clauses might be meaningless. The question was: What should he do to encourage Jim to stay and continue to ensure that DATAMINEWARE stayed ahead in its Red Queen game with its much larger competitors?