Utilizing Agile Methodologies to Transfer Knowledge and Ownership for thetrainline.com

Vinod Sankaranarayanan

Project Manager ThoughtWorks

Pravin Thakur

Offshore Development Head thetrainline.com

Abstract

While its natural of people to roll-off and roll-on to projects, there will also come times when an entire engagement is transferred over from one group to another. This could involve transfer from one department to another within the same organization, or transfer from one vendor to the customer, or from one vendor to another vendor. At its most fundamental level, such a transfer involves change. Elements of change management play crucial roles in these initiatives. The other important aspect on such programs is the true scope of a transfer. In many ways, the industry has done a disservice to this activity by terming it as ‘Knowledge Transfer’. It really is ‘Ownership Transfer’ and it involves much deeper dimensions than just knowledge. Agile methodologies and the principles of Agile delivery aid and align well for such onerous tasks.

In this brief report, we share our experience of one such program embarked upon and completed with great success. The objective was to transfer ownership of thetrainline.com from ThoughtWorks back the Trainline team. The intention was to bring the ownership over to London without disruption of release activities and retaining if not increasing efficiencies in project delivery. Of course, the unstated need was to ensure the production systems do not go down.

Background

ThoughtWorks had been working with the Trainline.com for over five years. Over this time, the team had built and enhanced the thetrainline.com system. The platform transacted more than a billion pounds of ticket sales annually and is the bedrock of the organization. Given certain strategic directions taken by the trainline board, the IT team was asked to transfer ownership of the Trainline.com system over to the London teams.

This was challenging because,

- The ThoughtWorks team comprised of about 100 members in Bangalore. The Trainline team comprised of about 60 members in London.
- Along with the transfer, a parallel line of ramp-down had to occur in Bangalore, thus reducing the number of people to support the new team in London.
- Since we had feature based teams in Bangalore, we had one large monolith on which any team could work on. London teams were structured as product based teams. Meaning, each team worked on a specific

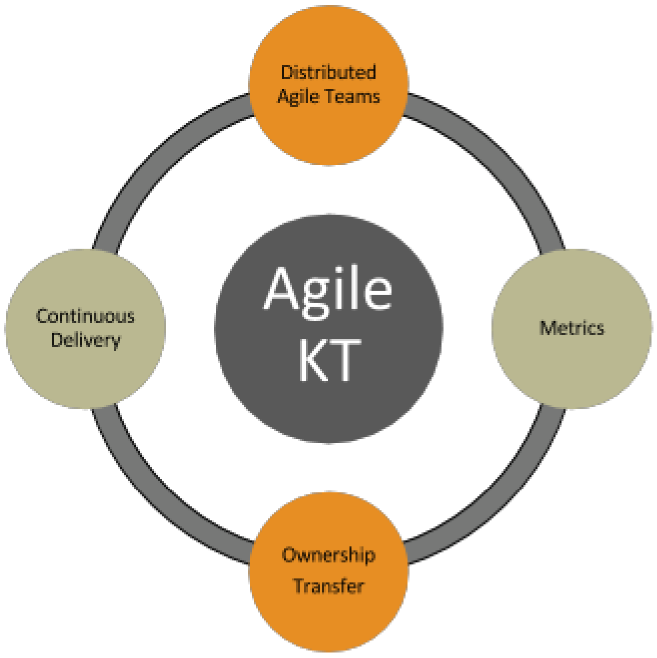
area of the application. Hence the large platform had to be broken down into meaningful products to be transferred over to the London Teams.

Project Demographics

In all,150 members from two organizations were involved in this exercise. The profile of the members spanned across, developers, quality analysts, business analysts, PMs and a few leadership roles. The members had on average about 6 years of IT experience.

Solution Strategy and Approach

Since the teams had always been Agile, the solution was to use Agile methodologies to effect the transfer.



The picture above is a broad stoke of the various agile themes we utilized.

Distributed Agile Teams

- Remote Pairing and distributed development
    - It was not feasible to bring all of the team members to a single location owing to costs as well as visa related matters. Our only option was to create a distributed team. We made sure to purchase good circum-aural headsets for the pairing. We undertook exercises to identify the best tools that will aid in remote pairing. Purchase costs of these tools were not deeply debated as the larger purpose was efficient transfer. The teams also got an excellent indication of the commitment and support from the leadership on making this a success. We had one Development Manager for the distributed team and one BA providing requirements for the entire distributed set-up.
    - We effectively used tools and technologies for remote stand-ups, iteration planning meetings as well as remote retrospectives.
- Using the concept of stories for the transfer

- Every iteration had a story that catered to the effort spent for transfer as well as identified discrete elements that were to be transferred at the end of the iteration. However, these stories were not created from a functional perspective or had description of the code to be transferred. Expectation around being able to support a sub-functionality in production was the main criteria for accepting the story to be complete.

Metrics

- Creating Knowledge Transfer Units
    - We logically broke the monolithic platform and identified discrete modules to be transferred over. There were called Knowledge Transfer units. These could have been defined across various dimensions viz. functional streams, technology streams, or architecture layers. This was quite important as it sets the tone and direction of the knowledge transfer. Working code was unequivocally identified as the foundation to be transferred over. Hence we chose the logical architecture with code as the first means to transfer over. This also allowed the transformation from feature-based teams to product teams since the transfer was made to teams who would eventually take over that unit as a 'product'.
    - Prioritizing to identify the top 70% that are 'must haves'
- Utilizing continuous integration as a way to keep the teams honest on the knowledge garnered
- We toyed with the idea of using number of regression defects before and after the transfer for specific units as a way to measure the effectiveness if the transfer. But that became infeasible given the team structures and ownerships were also changing along the way.
- Adhering to a Fixed Budget and Fixed Schedule
    - The overall program was identified to be x million pounds within a timeline. Since scope was the most flexible constraint, we moved forward with the understanding that we will cease the program the moment the Budget runs out.

Continuous Delivery

Part of the challenge was to ensure operations continued unabated. In a sense this was similar to changing pilots on a cruising aircraft.

- We continued to deliver commercially important projects to ensure the organization sustained its growth during this period
- Identifying risks and continuing to evolve them as we moved along. Some of the risks identified are given below
    - Difference of opinion amongst teams
    - Attrition on Bangalore and London side
    - Not being able to solve production issues
    - Measuring the success of the program. We don't know what good looks like
- Part of continuous delivery was also to identify projects that can be utilized for the transfer itself. Since we did not have the right set of commercial projects to aid in the transfer, we used the exercise of breaking the monolithic platform an aid for the transfer
- Focusing on service tests and other automation test elements as a way to transfer functional and domain knowledge
- Modifying processes along the way to suit the London team's takeover, ensure there were no snags

between or subsequent to the handover

Ownership Transfer

A major component in the whole exercise involved transferring ownership over to the London teams. A lot of subtle aspects surround this. The final outcome obviously is for the new team to be able to effect any change and provide any support independent of the Bangalore teams. A lot of activities were focused on ensuring this ownership transfer occurred smoothly.

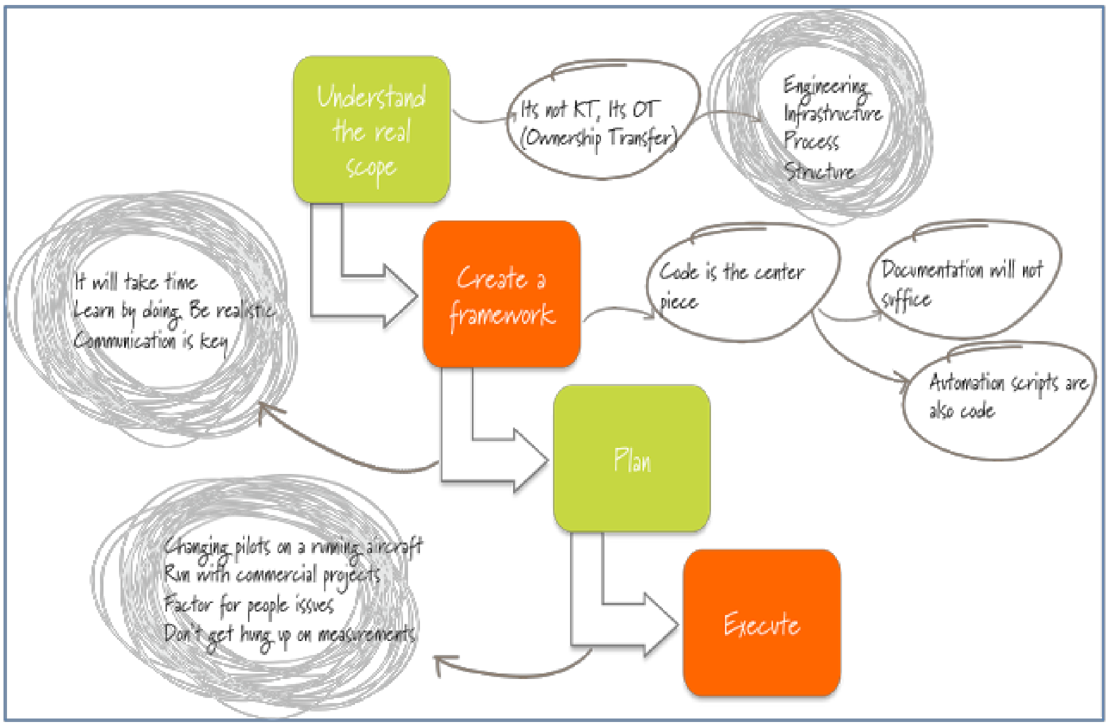
- Redefining production support.
    - Since an issue can impact multiple areas of the application, redefining how teams can come together and quickly fix production issues was a major element to this activity. A lot of this had to do around setting expectations at individual levels.
- Resetting expectations with other stakeholders (business groups)
    - A broader group of stakeholders get impacted with change in ownership. Business teams that were used to reaching out to Bangalore had to re-directed to reach out to teams in London. Also with the new model of product ownership, one project now involved more than one team to execute the project. This required discussions around process changes as well.
- Redefining individual expectations
    - This was a crucial aspect especially in London since the team's responsibilities and area of ownership had increased.
- Infrastructure
    - Ownership transfer also involved moving out hardware from Bangalore and augmenting hardware in London to support the additional load. This mini-project was also used as a way to transfer knowledge
- Changing some team structures.
    - With a smaller team, which will veritably provide a smaller throughput, some of the erstwhile roles, had to be either absorbed within functional steams or made redundant. The end-to-end testing teams and automation teams were absorbed into functional and regression teams.
- Having a steering group that met every week to take a health-check and remove impediments for the program.

Outcome

The program got over in ten months. We had expected to ramp down the Bangalore team to ten members. But with the evolving situation, we ended with a final team size of 20 members. The purpose of the transfer was more than served with Trainline posting record profits, part of it accrued from the cost savings through the program.
As can be expected in Agile projects, there was no big-bang release to signal the end of the program. A series of smaller bangs occurred over time, with different units getting transferred over. The end of the program was more of an anti-climax. People had to be told to realise that it is over.
Till date we have delivered six projects through the transition and as part of the new set-up.

Summary

Agile and lean principles can be applied over a diverse spectrum of activities apart from just project delivery. This knowledge transfer exercise is one such example. In a typical build operate transfer model, an agile flavor will add significant value to the execution. Since transfer programs are quite critical, an agile approach allows one to work on it and evolve over time, while continuing to deliver commercially important projects.
Also, since the dynamics involved in a program such as this is sensitive, agile methodologies allow us to get early feedback and make course corrections quickly.


Understand the real scope
Its not KT, Its OT (Ownership Transfer)
Engineering Infrastructure Process Structure
Create a framework
Code is the center piece
Documentation will not suffice
Automation scripts are also code
It will take time Learn by doing. Be realistic Communication is key
Plan
Changing pilots on a running aircraft Run with commercial projects Factor for people issues Don't get hung up on measurements
Execute

The diagram above provides a quick glance of the framework to be applied to charter an Ownership Transfer program.

Understanding the Real Scope
Many of us commit the mistake of looking at this as a pure knowledge transfer exercise. But a true ownership transfer exercise involves four elements. These are,

- Engineering changes
- Process changes
- Infrastructure
- Team Structure

Any new team will change the way things are done to suit their comfort. This can involve how engineering level aspects, such as continuous integration, test driven development and so forth. It will also involve changes to processes such as inceptions, showcases and so forth. Similarly infrastructure may need to be handed over or transferred over. Scale may change. A team size difference will bring about changes to structure as well.

Create a Framework

A transfer program is not a regular project. Many things are different from a regular project. The end goal often is uninterrupted production runs and smooth as ever project deliveries. Hence it is important to put a framework in place that can be understood and internalized by all constituents. As always working code needs to be at the center of this framework. All activities need to be designed keeping this key tenet in mind. Documentation only aids as a tool to share information. People seldom go back and refer to documents a second time. And it is practically impossible to keep documents current.

Plan

Planning involves identifying realistic costs and timelines. Ownership cannot be transferred in two or three months. It will take a much longer period of time and a few releases. Communication and setting the right expectations with the project sponsor and other stakeholders is key. Also, when it comes to a program of this nature previous project data will not help. Team dynamics, the background behind the transfer and prevailing atmosphere becomes vital in determining the success of the program. It is important to focus on pairing and working on the code as a means to learn. Code walk-throughs and documentation will not be adequate to ensure a successful transfer.

Execute

As in any project, a number of issues are bound to come up while execution. People related issues would be very high on these types of programs. This can mess up carefully laid plans. Similarly, measurements and metrics are quite tenuous to track for such programs. The real proof would be a seamless production system and smooth releases. If planned and executed over time, this can be achieved.