Dual-Track Agile in Early-Stage Startups

Elena Lape
School of Informatics
The University of Edinburgh
Edinburgh, United Kingdom
e-mail: elena@tardis.ed.ac.uk

***Abstract*—Dual-Track Agile is a new type of Agile development where work is broken down into two parallel tracks: generating validated ideas for the development backlog (Discovery track) and turning those ideas into software (Delivery track). There exists a small body of research on Product teams within established companies using Dual-Track Agile. In contrast, this study focuses on how and for what reasons early-stage startups use this model in their product development processes. We interviewed five early-stage technology companies, and found that startups use Dual-Track Agile to ensure that their software meets customers' needs and to build lasting relationships with potential users who can be valuable in later development stages. Most researched startups who identify themselves as Dual-Track Agile adopters do not follow strict methodologies, but instead, their workflows are driven by Product management and software development tools they use. We also found that more defined product development processes are introduced as companies grow.**

***Keywords*—Agile software development; Software engineering; Engineering management; Product development.**

I. Introduction

In an early 2021 publication, Porter and Heppelmann described how an increasingly interconnected world created a competitive landscape in digital product development [1]. Good quality software on its own is no longer enough to please people: consumers are looking for tools that solve their problems better than existing ones. Naturally, companies developing software products need to test, iterate, and experiment with the market constantly to keep up with the competition. Each time a new product enters the market, user expectations also rise — it gives them ideas about what other functionalities there could be.

In order to stand out from the competition and not waste time on features that bring little value to users, technology companies need to find ways to quickly validate product ideas and make sure that only validated ideas are implemented. The Dual-Track Agile, also known as the Continuous Discovery and Delivery model, offers Product development teams a solution to this challenge by combining the Discovery and Delivery activities in parallel. According to this model, teams need to align their software development processes with continuous market research and validation.

Startups' adoption of the Dual-Track Agile model is not widely researched in academia. This study focuses specifically on early-stage company product development practices.

The main goal of this study was to learn how startups adopt Dual-Track Agile in practice. We broke down this goal into three components, and so the following research questions were considered:

- What are startups' motivations for using Continuous Discovery and Delivery, provided they do use it?
- What Continuous Discovery and Delivery practices do startups adopt, and how are they built into their processes?
- How does the scale of the company affect its Continuous Discovery and Delivery practices?

The study was organised in two parts. First, we reviewed the existing work to understand Dual-Track Agile and the challenges that come with it. Then, we interviewed five different startups to learn about how they implement Discovery and Delivery in their teams.

Section 2 presents the related work relevant to this study, Section 3 describes the methodology used to sample and interview startups, Sections 4 and 5 present and analyze interview responses on the two topics of Discovery and Delivery respectively, Section 6 discusses the results of the interviews, Section 7 proposes some limitations to the study, and Section 8 concludes with a summary and future work.

II. Related work

Single- and Dual-Track Agile methodologies share a similar philosophy in that they both recognise the need for flexibility in software and product development.

A. Single-Track Agile

Many companies developing digital applications have adopted an Agile approach to software development. Traditional Agile methodology focuses on software functionality and is flexible to adapt to changing functional requirements [2]. However, there are at least two problems with Agile development, given today's competitive landscape.

First, Agile treats "working software" as the teams' primary measure of progress [3]. This measure emphasizes technical excellence and building out features but does not address the user perspective side of the development. Without a process to closely couple Delivery with market research, the products may end up being high quality

 ISBN: 978-1-61208-894-5

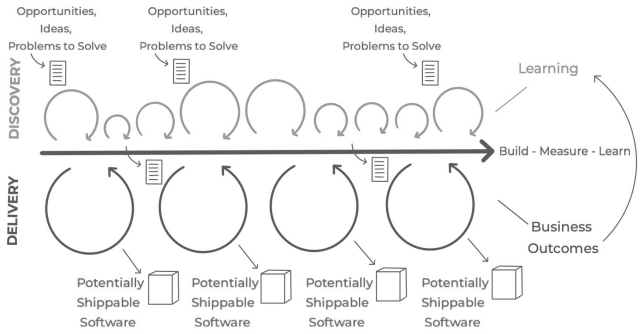

Figure 1: The Discovery-Delivery model (adapted from [9]).

engineering-wise, but something that the customers do not like or use. Without an approach that integrates more User Centered Design elements, Agile becomes not too dissimilar from Waterfall, where "testing and validating" is a discrete phase of software development that is usually final one [4].

Second, Agile assumes the presence of a "client" or a "customer", i.e., a stakeholder who creates and changes requirements and sets priorities for the Development team. Sedano et al. highlight that "Agile methods do not explain how to identify stakeholders, understand them, model them or turn this understanding into stories that developers can readily implement" [5]. Agile does not have inherent support for constantly evolving markets, frequent stakeholders changes, and an ecosystem where user requirements are yet to be uncovered.

Interestingly, the two issues highlighted above have been identified even before the Agile Manifesto was created in 2001. Gilb and Finzi's *Principles of Software Engineering Management* published in 1988, then introduced *Evolutionary Delivery* as a potential solution: delivering a system to users in "miniature incremental versions" to allow for changing markets dictating product pivots, and collecting plenty of feedback [6]. However, their Evolutionary Delivery model did not provide guidance on how exactly teams should prioritise ideas and requirements, or how and at what stage they should involve end-users in the feedback process.

B. Dual-Track Agile

Dual-Track Agile (or Continuous Discovery and Delivery) is a new type of Agile development. It offers to solve traditional Agile's shortcomings by combining Product Discovery (the ability of a company to validate products, services or features before implementation) and Delivery (the technical implementation and deployment of the identified outputs of Product Discovery) activities in parallel, as defined by Trieflinger et al. [7]. According to Cagan, Continuous Discovery and Delivery is a model as opposed to a methodology [8], as is illustrated in Figure 1.

Following the observation that companies see Product Discovery as a necessity but "struggle in finding an approach to conduct Product Discovery and Delivery activities", Trieflinger et al. found five different implementation approaches to Dual-Track Agile in grey literature:

- *Pay to Learn/Pay to Build Cycle.* Product Discovery is carried out in the "pay to learn" cycle, while Product Delivery is carried out in the "pay to build" cycle. In the "pay to learn" cycle, the process begins with the gathering of ideas, such as performing interviews with prospective clients and other related internal stakeholders. To test each idea, one or more hypotheses are established based on these ideas. Experiments are used to test theories, and good ideas may be added to the product backlog. A first iteration of a concept is introduced and shipped to consumers in the pay-to-build cycle, and then feedback is obtained — this serves as input for the next Discovery cycle.
- *Now-next-later Product Roadmap.* Product roadmaps that facilitate the Delivery of goods, features, or services focused on consumer and business value are known as now-next-later roadmaps. Theme-based roadmaps, in contrast to outcome-driven roadmaps, add an additional layer of aggregation (themes) from which the desired outcomes are derived. The three columns 'now', 'next', and 'later' in both roadmap formats indicate the time horizon (in contrast to time-based roadmaps)
- *Product Kata.* Product Kata is a four-step method for assisting Product teams in aligning around a common goal, learning about customer needs, and determining the best solution to build. It is a high-level model with three Discovery-related steps, after which the team must choose the Delivery implementation process ("solution optimization").
- *Lean Sprints.* This type of Dual-Track Agile implementation was introduced by Ash Maurya, author of "Scaling Lean". Sprints are typically two weeks long, and each phase of product development consists of one Discovery sprint and one or multiple Delivery sprints. To generate a variety of ideas and hypotheses, the author suggests forming a cross-functional Product development team that includes designers, software developers, and marketers [10].
- *Dual Track Scrum.* Each track has its own team in the 'Dual-Track Scrum' technique. Lead designers and developers usually make up the Discovery team, while developers and software testers make up the Delivery team. To improve communication and workflow between the two teams, it is recommended that a Product Owner be included on both teams. The Discovery team gathers data from users, creates prototypes, and tests ideas. Ideas are classed as "mature" (ready to be added to the scrum backlog and implemented), and

 ISBN: 978-1-61208-894-5

"immature" (not useful for users or not viable to be implemented by the Development team) [11].

Most Continuous Discovery and Delivery workflows from Trieflinger's literature review assume a cross-functional Product development team, which is largely a characteristic of larger established companies.

Finding suitable workflows is especially tough for smaller teams and startups who may have a limited budget and a small number of employees, but who need to move quickly to meet the market. It remains unclear how early-stage startups adopt this model in practice, how they structure it compared to larger teams, and what value Dual-Track Agile brings to their team or products.

III. METHODOLOGY

We interviewed five founders and early-stage employees at technology startups developing software-based products. The interviews took place between March and May 2021. In this initial study we had limited resources — therefore, we adopted a qualitative approach to interview a small number of startups showing signs of potential to be successful supported by a review of valuable literature. This means that our conclusions are tentative but they are indicative of the issues encountered by startups in developing new products in a Dual-Track Agile way.

A. Selection Criteria

The research target group were C-suite executives, Product Managers, and Engineers who were early employees — one of the first twenty — in companies that are younger than eight years old. The group self-identified as individuals who use Dual-Track Agile or Continuous Discovery and Delivery processes at work.

The target companies were for-profit businesses with a recent valuation (through equity investment received or EBITDA [12]) of at least 1 million US Dollars. Their core business must revolve around a software product(/-s) or a service(/-s) that at the time of the study had at least 50 individual users.

B. Research Participants

1) Participant A/Company A: Company A is a two-year-old SaaS company based in Canada valued at approximately 50 million USD. They develop virtual reality applications for showcasing 3D models of buildings' interiors and exteriors. The company employs 20 people full-time and outsources a remote Software development team overseas consisting of 30 software engineers-consultants. Company A's clients are typically property development firms. The company is a spin-off of another two-year-old company specialising in Virtual Machine (VM) provisioning for consumer applications. Participant A is the co-founder and Chief Technology Officer (CTO) of both companies. He has a professional background in full-stack software engineering and developing VM and Cloud Computing environments.

2) Participant B/Company B: Company B is a two-year-old United States-based software infrastructure cost analytics startup with a valuation of approximately 30 million USD. The company has 12 employees, and their core product is a platform that Software development teams use to plan and predict and optimise their software infrastructure costs. At the time of the study, the product had over 1000 users, including several well-known enterprises. Company B describes their product as open-core: their engine is open-sourced on GitHub [13]. Participant B, the company's co-founder and CTO, has a professional background in site reliability engineering at major technology companies and full-stack engineering at startups. They also have a Bachelor's degree in Computer Science from UC Berkeley.

3) Participant C/Company C: Company C is a two-year-old United Kingdom-based startup developing a digital application that allows users to manage the storage and insurance of their collectable assets. Their target market is primarily luxury car collectors with busy lifestyles who outsource the management of their car portfolio. The product is currently in its beta stage with over 100 user sign-ups, has a valuation of over 1.3 million USD, and five full-time employees. Participant C is a founding engineer of the company. Their professional background is in building full-stack web and mobile applications, and they have a Bachelor's degree in Political Science from The University of Edinburgh.

4) Participant D/Company D: Company D is a United Kingdom-based startup with a recent valuation of around 6 million USD, focusing on providing web scraping solutions for financial institutions and government organisations. Most of the company's work is project-based. Company D has been working on these types of projects since 2016 and currently has 20 full-time employees. Participant D is the Vice President of Technology in the company overseeing the Software development team's work. They have a professional background in low-level software infrastructure engineering and full-stack software development.

5) Participant E/Company E: Company E is a three-year-old company based in Romania, targeting the European higher education market. Their core product allows prospective students and university admissions consultants to manage multiple international university applications in one platform. Company E has recently been valued at 1.4 million Euros and has around 35 employees, half of which are full-time. Participant E is the company's Business Development Manager who joined the company within the first 20 employees. However, they were the first person to be hired to focus specifically on the Discovery Track. Their professional background is in data analytics and entrepreneurship-themed event organising, and they have a Bachelor's degree in IT and Business Management from Manchester University.

 ISBN: 978-1-61208-894-5

C. Interview Process

The nature of the study is exploratory, therefore the schedule was constructed in a semi-structured manner. To avoid researcher's bias and allow for rich qualitative data to be collected, we used open-ended questions. Interviews with each participant were conducted in a one-on-one over a video communications platform. Each interview took between one and two hours to complete.

At the start of the interview, each participant was asked a number of questions that called for discrete answers, such as company valuation, company size, their technology stack, or concrete product development methodologies. This data allowed us to compare and contrast the startups on the same terms, and draw more objective observations.

Later in the interview, each study participant was asked to first describe their Discovery processes: e.g., what Discovery means to them, who (if anyone) leads their Discovery efforts, what Discovery data (if any) they collect, and what they do with it. Then, they were asked to do the same with Delivery: e.g., how they define Delivery, who leads Delivery at their company, how they organize it and using which tools (if any). Towards the end of the interview, the participants were asked to talk about how they bridge Discovery and Delivery, naming any specific tools, people, or methodologies they rely on.

D. Data Processing

The audio extracted from the videos was processed using Amazon Transcribe to generate time stamped transcripts in JSON format [14]. To parse the JSON data into a human-readable format, we used the tscribe Python library to convert the transcripts into readable document format [15]. This way, we were able to find the participants' responses to specific questions and discussion themes based on the timestamps, summarize their responses, and compare with other startups' responses regarding that topic or theme.

IV. The Discovery Track

This section presents summarized and analyzed responses to Discovery-related interview questions and themes.

A. Interview Responses

1) Participant A/Company A: Company A business as project-based, where the deliverable is a 3D VR property visualisation. Each project starts with the client providing a property plan and is managed using a simple *to-do*, *in-progress*, *done* Trello board [16]. An initial prototype is then created and sent to the client for revisions.

To make revisions, in the earliest stages of the company Participant A would "initially meet the client every single week and walk with the client with a VR headset and listen to them what they wanted to fix in the VR deliverable". However, this proved to be inefficient and "such a hassle".

Participant A "ended up writing a very, very simple application". Clients were given access to a VR application where they could "draw a little red circle and write notes on what they want to change". This helped Company A speed up and scale their Discovery process. The red circle would automatically spawn up a "to-do card on Trello" in the product development Kanban board. This created "a closed feedback loop with the developers".

Automating their Discovery process helped Company A spot patterns across different clients and improve their initial prototypes and shorten the Delivery cycle from months to weeks.

2) Participant B/Company B: Company B operates in the software infrastructure space. In 2019, Participant B co-founded a two-person open source project on GitHub to test their idea. From early stages onward, open sourcing the core of their engine allowed them to verify that there is a market for the tool via growing GitHub popularity and incoming outside contributions. At the same time, they were collecting user feedback and feature requests through GitHub Issues.

As the tool's popularity grew, Company B created a community Slack workspace for the users, as a way to provide customer support and observe the pain points the company could address [17]. Slack also helped them build "close relationships with power-users who were willing to share more detailed feedback".

Feedback and feature requests are translated into "product themes, areas of improvement or new opportunities", which are then "added into quarterly roadmaps". Tasks are prioritized based on the "severity of the issue" (e.g., "a security bug will be very high priority"), implementation scope, and "how much revenue depends on getting this done" — for example, if a paying enterprise client is asking for a particular feature, and Company B identifies that there may be other potential customers facing similar problems, then the feature's priority will be high.

3) Participant C/Company C: Participant C describes the Discovery processes of their collectible asset management app as mostly ad-hoc and heavily based on "building personal relationships with potential customers" that emerged from the founders' personal network.

Participant C believes that that "clearly defined organised Discovery processes found in business manuals distract you from the goals when the company is small", and they remarked that "it would waste everyone's time and it would not fit in correctly; we are working very chaotically together, and that's how we get the best ideas, the best things ever". Participant C stresses that working with "experienced, intelligent teammates" is what allows them to communicate loose research and development plans and feature roadmaps verbally, and "keep them in their head".

 ISBN: 978-1-61208-894-5

In addition, Company C relies on readily available SaaS tools to conduct Discovery. For example, they use FullStory to collect usage statistics and user experience metrics from their private beta [18]. This data is verbally interpreted by the founding team as features and fixes that they then go on to implement.

4) Participant D/Company D: Company D develops web scraping and search products for clients on a project basis. Each project begins with the client providing a list of functional requirements. However, there is a "surprising amount of variance between clients", therefore Company D has to research government regulations and compliance documentation independently.

Given the nature of the contracts being confidential and subject to government regulations, Continuous Discovery happens in-house. Participant D said that after each iteration, they would have "demo days to pitch them to a range of cross-functional teams (sales, engineering) within the organisation". For Company D, the demo days "have become a very core way of being able to prove concepts quickly for things that might not necessarily be immediately visible even to clients". Demo day findings become input for the next Delivery cycle, organised as a 2-week sprint.

5) Participant E/Company E: Company E is developing an all-in-one platform for university application management. At the start of their business, they first quickly built a prototype based on their hypotheses, and then "participated in educational conferences in order to get access to a network of university admission consultant agencies". Then, they "started having phone calls with them presenting what they want to build in the future", showing them the value that the product brings and observing reactions.

Participant E highlights quality over quantity when gathering feedback for their releases. "Every two or three weeks", Participant E shows customers several versions of the product, observes them using it, going step by step, collecting answers to questions such as "What do you need here?", "What do you see there?", "Is this useful to you?", "We're working towards this feature, what do you think about it?".

Company E uses Productboard [19] "to organise answers and ideas into themes", which then go through a prioritisation process with Product Managers based on scope, urgency and "market opportunity" once a month. Once features are determined, they are added to the next Delivery cycle. For "small bug fixes, developers are contacted directly in a dedicated Slack channel".

B. Analysis

Discovery differs significantly in project-based companies versus companies who offer a single platform. Companies A and D typically have an initial list of requirements from clients to build a prototype they can iterate on, whereas Companies B, C and E have to validate the market and then navigate the problem ambiguity themselves.

Companies B, C and E value building personal relationships and directly talking with potential customers to understand their needs over having directed, streamlined Discovery processes. As such, the initial requirements that enter their Delivery Track are somewhat experimental in nature.

It is important to note that in the cases of Companies B, C and E, founders admitted that their personal experience and connections gave them a head start in understanding the market and customer' needs. It is unclear whether this a prerequisite for building a technology product, but it is likely that it helps them save time and resources that would have to be devoted to conduct initial market validation and research otherwise.

Another observation is that founders all emphasize specific tools (Trello, GitHub, Slack, FullStory, Productboard) that each carry a particular workflow. Similarly, automating user feedback collection is a common theme among A (with VR app), B (with GitHub Issues, Slack) and C (with FullStory). We may hypothesise that these tools themselves help startups define Discovery processes and link them with Delivery.

V. The Delivery Track

This section presents summarized and analyzed responses to Delivery-related interview questions and themes.

A. Interview Responses

1) Participant A/Company A: Company A bridges Discovery and Delivery via a Trello Kanban board that is set up for each project. All development is outsourced to a remote Software development team overseas, and the "developers are managed by in-house Project Managers". The Project Managers are responsible for review and verification, as well as task distribution amongst developers.

Each project is set up using a CI/CD (Continuous Integration/Continuous Deployment) framework that the sister company previously built. At the start of each project, the project manager extracts feature requirements from the client's specification and puts them in the *to-do* list. Once a card is moved to *done* by a developer (whether it's for the initial version, or part of a client's revision), this triggers the CI/CD pipeline and generates a build.

Project and revision scope varies from client to client and revision to revision, therefore, time-constrained sprints are not used.

2) Participant B/Company B: Company B's Engineering team works in one-week sprints, which begin with a sprint planning meeting. When Participant B was asked why just one, they said that "the priorities change too much from week to week" at a startup, and contrasted this experience with Google, where sprints were two weeks long.

 ISBN: 978-1-61208-894-5

Company B has a CI/CD pipeline which generates local builds nightly, and a new software version (a production build) is released to customers every two weeks. Participant B notes that "if something is missed in a sprint, it should get picked up and done for the next release". However, security fixes "jump all the possible queues".

3) Participant C/Company C: Company C prioritised building their first version to be "robust, modular and scalable", which took them over a year to implement, and then released a private beta to a select number of customers that they had already built close relationships with for testing and qualitative feedback.

Participant C noted they built the software in a way that would account for pivots and make adding and subtracting new features easy and intuitive. Currently, Company C adds new improvements to their production environment daily.

4) Participant D/Company D: Participant D sees Delivery as "parallel to Discovery, but something that needs to be done in a different part of the brain".

Their engineering division consists of two teams: DevOps engineers, and software developers. For projects that are just starting out, DevOps engineers are responsible for scaffolding out the architecture and implementing a CI pipeline, and the Software team builds out the applications, and then does feature iterations. The engineers work in two week sprints, in line with the internal bi-weekly demos.

However, Participant D notes that this is a new process to them and at the beginning it was not standardised. Company D introduced sprints and teams' division as they scaled.

5) Participant E/Company E: Participant E is not directly involved in the Delivery process, therefore, it is unclear how those are managed in Company E's Engineering team.

B. Analysis

Project-based companies A and D work off an initial set of requirements, therefore, the process is more or less the same for each product they build. They both use CI/CD pipelines to generate incremental and production builds.

Company B delivers their product continuously by definition, starting with incremental commits to their public GitHub repository. Once the paid product was released and the team grew, more streamlined CI/CD processes were implemented to allow for scheduled releases. Bi-weekly releases help them set customer expectations, and help organise Delivery efforts internally for their grown team.

Company C described a different approach to building their software. It is somewhat similar to the Waterfall software development process because only a sophisticated version of the application was released as beta. However, it is different from Waterfall in that Company C conducts research alongside building their product, rather than following an initial set of requirements they set out. Moreover, the software is built in a scalable, modular way to allow for future changes. The reason behind releasing a polished beta and not a makeshift prototype could be the nature of C's target market: their users are busy, high net-worth individuals. These people may have limited time to provide feedback for several prototyped versions, and if the initial version does not appear to bring them value, they may lose interest in the product. In a market that is exclusive and limited in size, this could be detrimental.

All of the research participants but E indicated a strong software engineering background.

VI. RESULTS

To address the three research questions proposed in Section 1, we formulated the results of the study based on summarized and analyzed interview responses.

A. Startups' motivations to use Dual-Track Agile

Provided that the research participants were recruited based on self-identification as "applying Dual-Track Agile (or Continuous Discovery and Delivery) in their product development processes", we can conclude that this Product Management model is indeed used by early-stage companies. In their own words, participants use the model to "verify the market", "verify need", "build incrementally and test user behaviour", "not make too many irreversible assumptions", "test hypotheses".

In addition, the model brings another important benefit for single-product startups: it helps them build relationships with potential customers. Large companies already have existing users, but new entrants face the problem in that they do not have a user base to test with. Therefore, continuously engaging with the market and talking to their users on a one-to-one basis helps create buy-in from customers and use that buy-in to gather feedback and ideas.

B. Continuous Discovery and Delivery practices

Some similarities do exist between startups and literature findings: Companies B, C, E highlighted some form of a roadmap as a way to bridge Discovery and Delivery. However, none of the study participants follow as streamlined of a Dual-Track Agile process as was found by Trieflinger et al., and instead adopt a similar mentality to Evolutionary Delivery [5].

We observed that each company's implementation of the Dual-Track model differed to an extent, which suggests that there is no universal tool or process that startups all adopt. Single product-based startups' (B, C, E) processes are overall quite simple and are largely influenced by specific tools.

Where possible, some of our study participants aim to automate their CI/CD processes and some user feedback collection (e.g. through FullStory for C, GitHub Issues and Slack for B, Trello for A) early on to save time and resources.

 ISBN: 978-1-61208-894-5

The implementation of the model varies also by type of startup's business model. Project-based companies (A and D), or companies that have already grown (B) aim to invest in implementing robust Discovery and Delivery processes, because they are not building for a market whose problems' solutions need to be researched, but instead — for clients who know what they want, or a user base that they already know well.

Early stage single product-based companies Discovery and Delivery processes appear to be characterised by the tools they use (Trello, FullStory, GitHub, etc.). Not over-focusing on the workflows allows them to prove concepts quicker.

Continuous Delivery practices across startups are more uniform and more streamlined than Discovery practices. They all appear to be building software in a modular, scalable way, and most use CI/CD pipelines. Their Discovery practices differ because markets, users and customers they are developing for are not uniform and each require a different approach. For example, a product aimed at asset collectors (C) requires forming personal relationships and trust via meetings and phone calls, whereas a developer tools company (B) can collect feedback via platforms that developers, conveniently, are already using every day, such as GitHub Issues.

Another reason why Continuous Delivery across most startups appears more streamlined than Discovery could be that most participants indicated a background in software engineering. It is likely that they already have experience in developing scalable, modular software for other startups or large modern companies, and so they know how to do it quickly and effectively. In addition, Continuous Delivery as a whole is a well-documented domain and so it is easier to find appropriate practices from the onset.

C. Dual-Track Agile as startups scale

We observed that in small-scale and very early-stage startups, defined Dual-Track Agile methodologies are not used, presumably because the companies are optimising for reducing time spent on tasks that are seen as counterproductive and administrative, and because small team size allows for coordinating efforts ad-hoc.

Companies do introduce some elements of organisation (e.g., sprints, quarterly roadmaps) at early-stage, though, as they scale (A, B, D) and hire more people, and employees take sole ownership of certain parts of the product.

VII. Limitations

First, the sample size of research participants is small as only five startups were interviewed. However, there are thousands of technology product startups incorporated each year. It is unclear whether it is common for early-stage companies in the world to adopt the Dual-Track Agile model in their work. Another important note is that participants were selected from an open call for participation, which means that not all founders and industries are represented. Therefore, the study results may not be generalized to the whole software startup ecosystem.

Second, the interviews are retrospective. It is possible that the participants forgot how their processes evolved over time and did not mention or purposely excluded some details they felt were unimportant, but could have been valuable for the study.

Third, the *Hawthorne Effect* may have played a part — this occurs when participants in a study are aware that they are being observed by scientists and, either consciously or unconsciously, alter the way they act or the answers they give [20]. Even though the participating companies were anonymised in the final study, some participants may have concealed important information about their processes to avoid risking the company's reputation. For example, if a company revealed that their processes change often and are chaotic, and a potential/existing investor recognises the company based on its description, the investor may choose not to invest in the following round.

VIII. Conclusion

In this study, we set out a goal to investigate how startups use Dual-Track Agile in practice, and considered three research questions:

- What are startups' motivations for using Continuous Discovery and Delivery, provided they do use it?
- What Continuous Discovery and Delivery practices do startups adopt, and how are they built into their processes?
- How does the scale of the company affect Continuous Discovery and Delivery practices?

A. Summary

We found that Dual-Track Agile is used by some startup technology companies and not just Product teams within established enterprises. The startups' motivation for using this approach is to not only verify the demand for their products or product features, but also to build close relationships with their users who can be useful in further development stages.

Most interview participants do not use the workflows found in Trieflinger et al.'s review and often rely on a set of particular tools to aid them in their broad Delivery and Discovery efforts. Where possible, early stage companies aim to automate their feedback collection and deployment processes to reduce the length of the feedback loop and speed up development time.

We found that instead of investing in processes, very early-stage products rely on verbal communication and direct prototyping to plan, validate and test their ideas. But as organisations grow in size, more concrete workflows and prioritisation processes, such as quarterly roadmaps or sprints, are introduced for visibility and organisation purposes.

As the research was carried out with five startup case studies, we recognise that our observations and results may

 ISBN: 978-1-61208-894-5

not be representative of the whole early-stage software company ecosystem.

B. Future work

A more systematic study of startups' product development practices should be done. Longitudinal case studies and overt observations of startups would allow us to see how and why exactly their processes evolve over time, as well as validate that the startups' own descriptions of their processes align with their actions.

As this study only involved five early-stage companies, the results may not be representative of startups as a whole. In the future, a larger sample size should be examined to draw more meaningful observations.

The five case studies indicated a relationship between specific SaaS platforms and their product development workflows. However, more research needs to be conducted to assess whether chosen tools are dictating the processes, or processes dictating tools.

This paper does not examine how the early stage companies' chosen practices affect their products' usability and success. A controlled real-time study examining startups who use Continuous Discovery and Delivery versus those who do not, could provide some insight into whether chosen processes contribute to some companies' success and/or failure.

References

1. G. M. Porter and J. Heppelmann, "How smart, connected products are transforming competition," Harvard Business Review, no. 92, pp. 64-88, 2021.
2. D. Salah, R. F. Paige, and P. Cairns, "A systematic literature review for agile development processes and user centred design integration,", 18th International Conference on Evaluation and Assessment in Software Engineering (EASE '14), no 5, pp. 1-10, 2014.
3. O. Hazzan and Y. Dubinsky, "The Agile Manifesto," in Agile Anywhere," Springer International Publishing, pp. 9–14, 2014
4. M. Light, "How the Waterfall Methodology Adapted and Whistled Past the Graveyard," Gartner Research, 2009.
5. T. Sedano, P. Ralph, and C. Peraire, "Dual-Track Development," IEEESoftw., vol. 37, no. 6, pp. 58–64, 2020.
6. T. Gilb and S. Finzi, Principles Of Software Engineering Management.Boston, MA: Addison Wesley, 1988.
7. S. Trieflinger, J. Münch, B. Heisler, and D. Lang, "Essential approaches to dual-track agile: results from a grey literature review" in International Conference on Software Business, pp. 55-69, 2020.
8. M. Cagan, Inspired: How to create tech products customers love, 2nd ed. Nashville, TN: John Wiley & Sons, 2017.
9. "Product Discovery," Aktiasolutions.com, 2020. Available: https://aktiasolutions.com/product-discovery/. [Accessed: 28 June 2021]
10. A. Maurya, "Scaling Lean: Mastering the key metrics for startup growth", Penguin, 2016.
11. J. Ciecholewski, "How to set up Dual-Track Scrum in Jira", Devbridge. Available: https://www.devbridge.com/articles/how-to-set-up-dual-track-scrum-in-jira [Accessed: 1 September 2020]
12. D. T. Emott, "EBITDA Valuation Engine" in Practitioner's Complete Guide to M&As, Hoboken, NJ, USA: John Wiley & Sons, Inc., pp. 264–275, 2015.
13. GitHub. Available: https://github.com [Accessed: 10 September 2021].
14. Amazon Transcribe. Available: https://github.com [Accessed: 10 September 2021].
15. Tscribe. Available: https://aws.amazon.com/transcribe/ [Accessed: 10 September 2021].
16. Trello. Available: https://trello.com [Accessed: 10 September 2021].
17. Slack. Available: https://slack.com/ [Accessed: 10 September 2021].
18. FullStory. Available: https://www.fullstory.com/ [Accessed: 10 September 2021].
19. Productboard. Available: https://www.productboard.com/ [Accessed: 10 September 2021].
20. S. R. G. Jones, "Was there a Hawthorne effect?," American Journal of Sociology, vol.98, no. 3, pp. 451–468, 1992.

 ISBN: 978-1-61208-894-5