



Agile Software Development

Lecturer: **Raman Ramsin**

Lecture 3

DAD: Process



DAD Lifecycles: Agile (1)

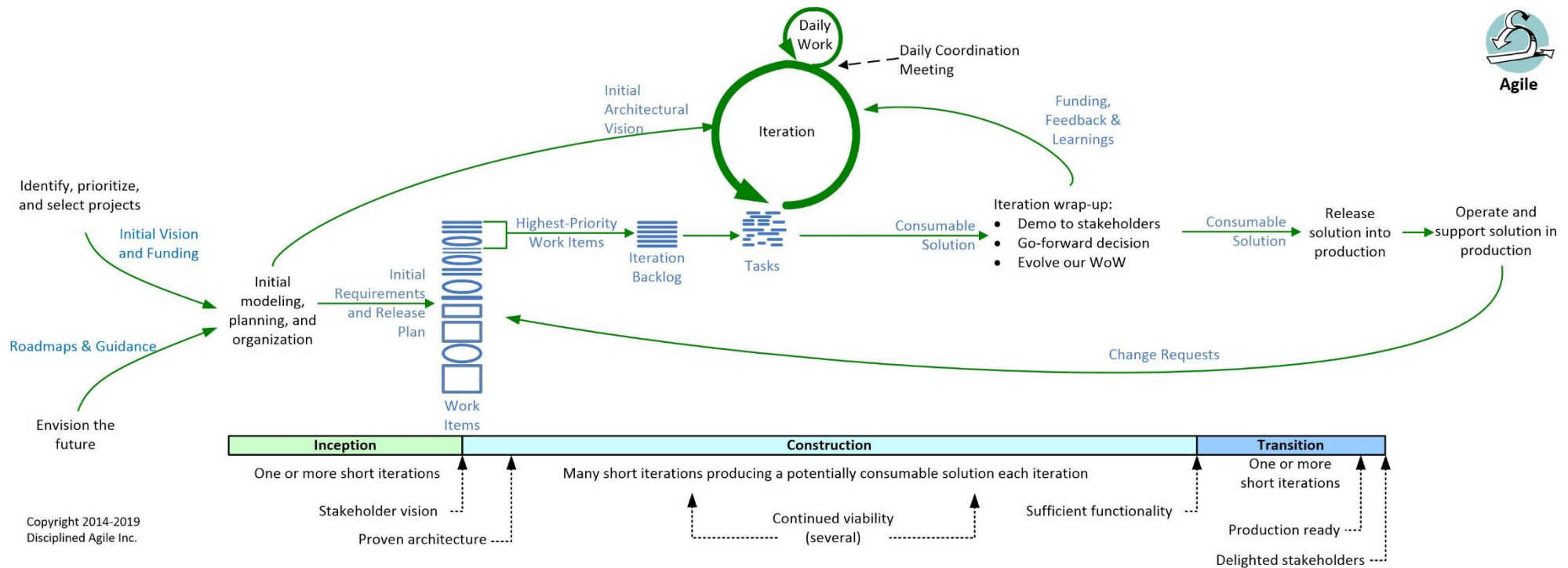
- Based upon the Scrum lifecycle with proven governance concepts adopted from the Unified Process (UP) to make it enterprise-ready.
- Inception: Produces an agreed-to vision regarding the expected outcomes for the team and how we're going to achieve them.
- Construction: Organized into short iterations of typically two weeks or less, in which the delivery team produces a potentially consumable solution (MMR).
 - Work items are prioritized by the Product Owner, primarily by business value.
 - In each iteration, the team pulls a small batch of work off of the work item list.
 - Periodical ceremonies: Detailed planning at the beginning of each iteration; demo at the end, and retrospective to evolve our WoW; daily coordination meeting.
- Transition: Ensures that the solution is ready to be deployed and if so then deploys it.



DAD Lifecycles: Agile (2)

- Explicit milestones: This lifecycle supports a full range of straightforward, risk-based milestones:
 - Stakeholder Vision.
 - Proven Architecture.
 - Continued Viability.
 - Sufficient Functionality.
 - Production Ready.
 - Delighted Stakeholders.
- Enterprise guidance and roadmaps: Important flows come from outside of the delivery lifecycle (from the overall DevOps strategy).
- Operations and support: When working on the new release of an existing solution, change requests are received from existing users.

DAD Lifecycles: Agile



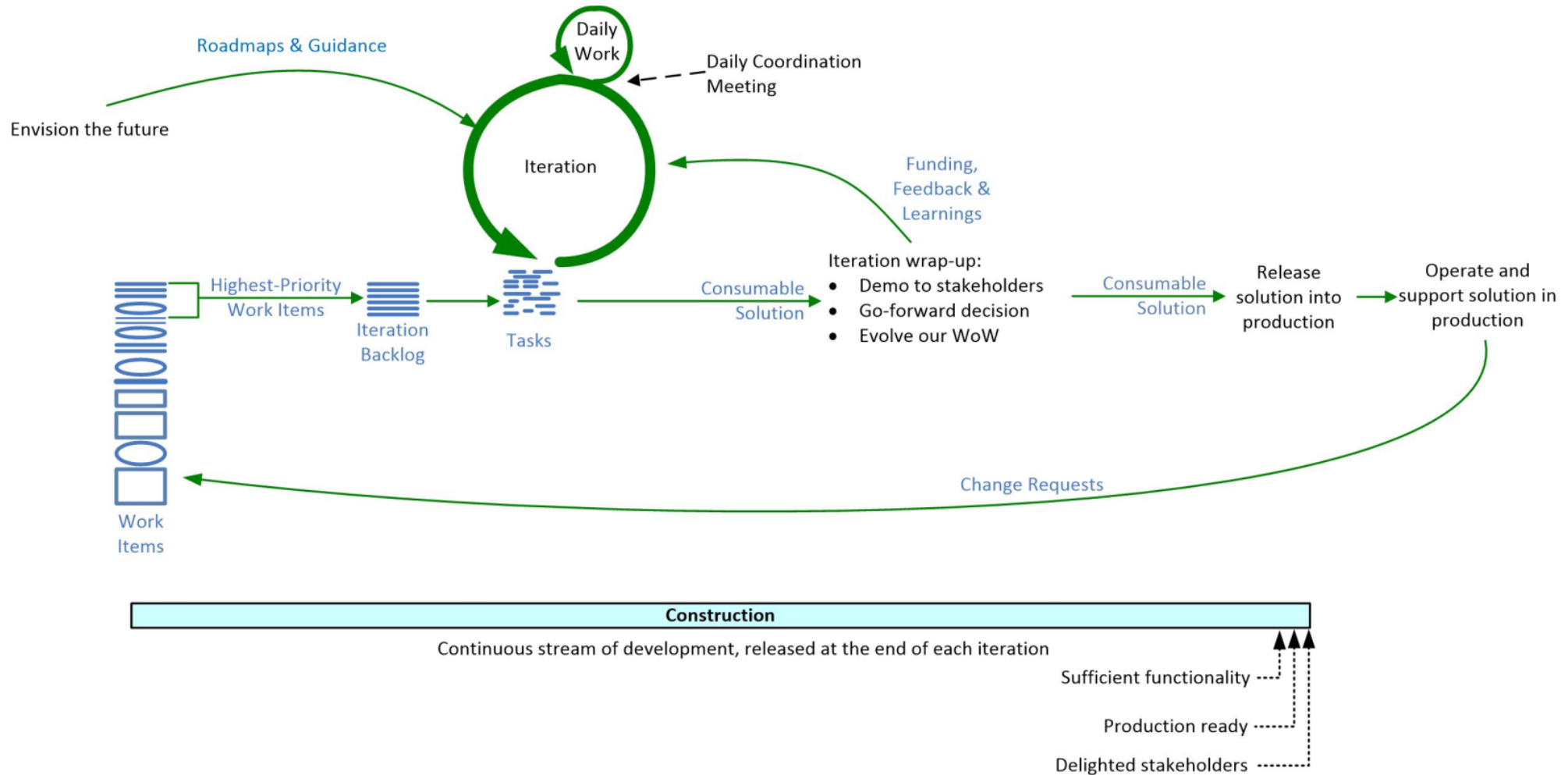
[Ambler & Lines 2020]



DAD Lifecycles: Continuous Delivery – Agile

- Teams typically evolve to this lifecycle from the Agile lifecycle, often adopting iteration lengths of one week or less.
- Key difference with Agile lifecycle: New functionality is released at the end of each iteration rather than after several iterations.
- Automation and technical practices are key.
 - Teams require a mature set of technical practices around automated regression testing, continuous integration (CI), and continuous deployment (CD).
- Inception has already occurred.
 - If significant change occurs, such as a major shift in business/technical direction, Inception will be repeated to reorient the team; so this is an activity, not a phase.
- Transition has become an activity.
 - Through automation of testing and deployment, it has evolved from a multi-day or multi-week effort to a fully automated activity that takes minutes or hours.

DAD Lifecycles: Continuous Delivery – Agile



[Ambler & Lines 2020]



DAD Lifecycles: Lean (1)

- Promotes lean principles such as: minimizing work in progress, maximizing flow, a continuous stream of work instead of iterations, reducing bottlenecks.
- Often adopted by teams who are new to agile/lean who face rapidly changing Stakeholder needs, especially by
 - teams evolving (sustaining) an existing legacy solution;
 - traditional teams that don't want to take on the risk of the cultural and process disruption usually caused by agile adoption (at least not right away).
- Teams address work items one at a time.
 - A major difference with the Agile lifecycle is the lack of iterations. New work is pulled from the work item pool one at a time (not in small batches).



DAD Lifecycles: Lean (2)

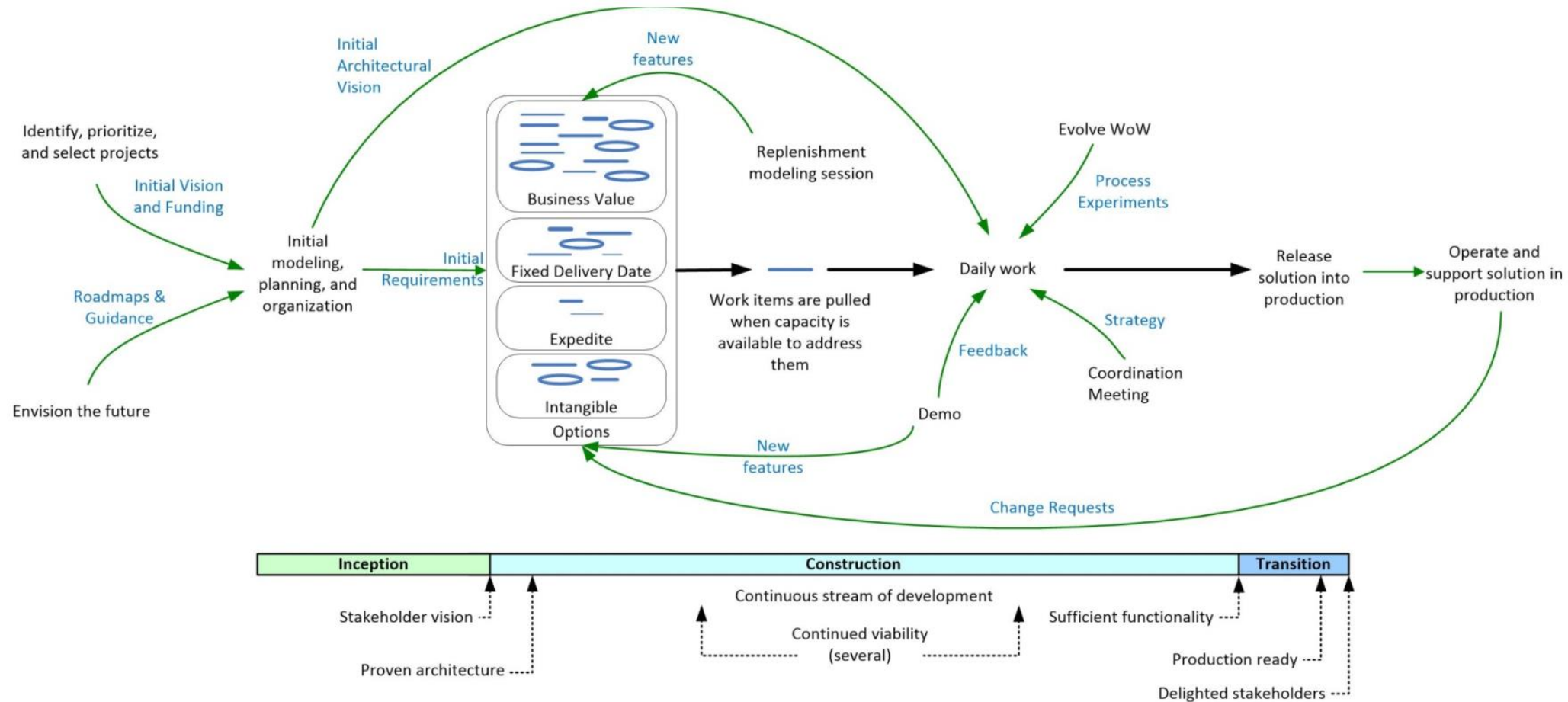
- Work items are maintained as a small options pool.
- Prioritization is performed on a JIT basis, with the team choosing the most important work item at the time when they start working on it.
- Work items are prioritized and organized into categories:
 - prioritized by value (and hopefully risk);
 - fixed delivery date;
 - to be expedited (often a severity 1 production problem or request from an important Stakeholder);
 - intangible (such as refactoring or training).



DAD Lifecycles: Lean (3)

- Practices are performed when needed, as needed.
 - Planning, holding demos, replenishing the work item pool, holding coordination meetings, making go-forward decisions, and others are performed on a JIT basis.
- Teams actively manage their workflow (in the pipeline).
 - Lean teams use a Kanban board for this; each column represents a state such as To Do, Being Explored, Being Built, Being Tested, and Done.
 - Work is depicted in the form of tickets (stickies on the whiteboard), with a ticket being a work item from the options pool or a task of a work item.
 - Each column has a work in progress (WIP) limit that puts an upper limit on the number of tickets that may be in that state.
 - As the team performs their work they pull the corresponding tickets through the process on their Kanban board so as to coordinate their work.
 - The team strives to minimize Lead Time (average time from start to finish).
- There is still an Inception phase and a Transition phase as well as risk-based milestones to support consistent governance.

DAD Lifecycles: Lean



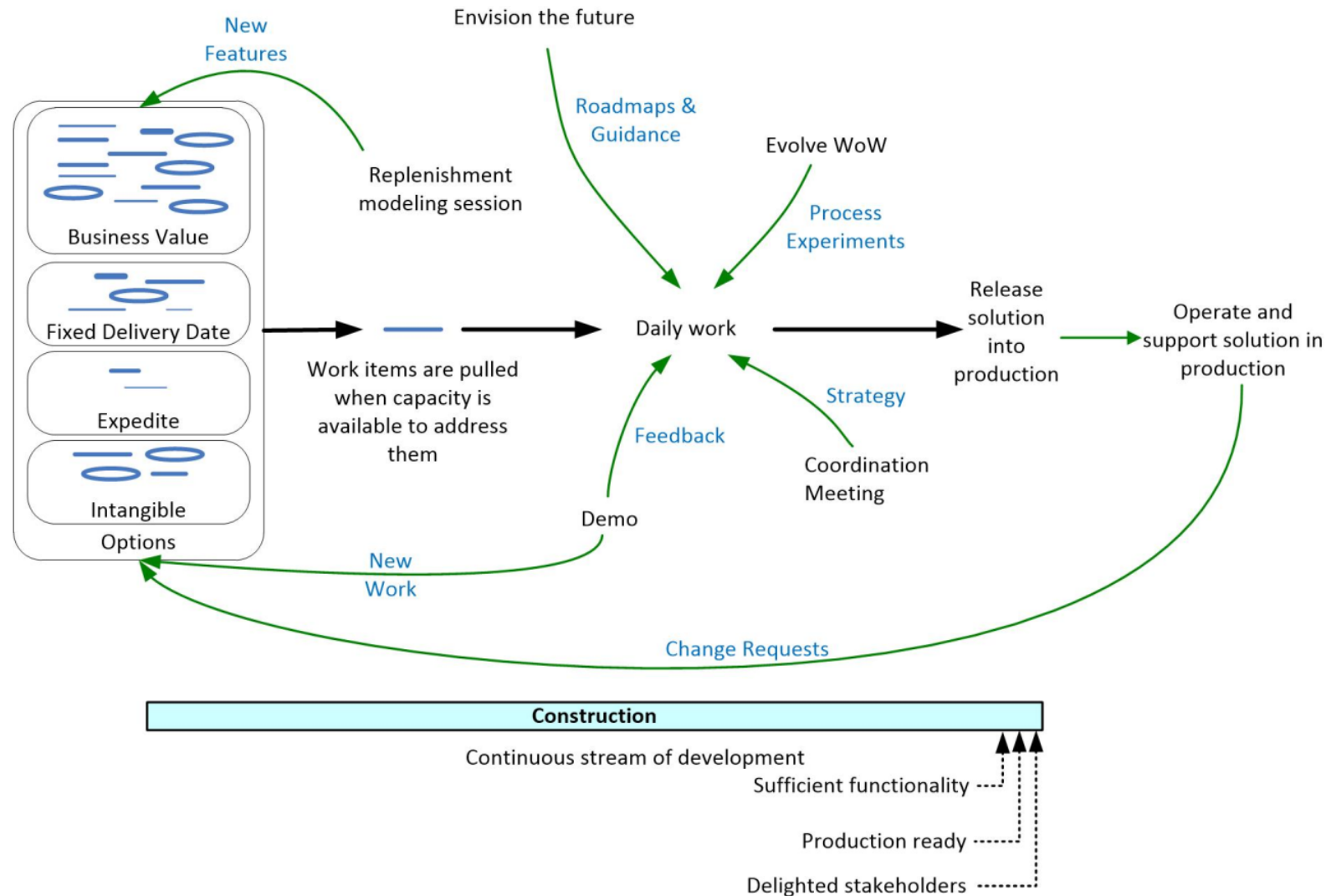
[Ambler & Lines 2020]



DAD Lifecycles: Continuous Delivery – Lean

- Teams typically evolve into this lifecycle from either the Lean lifecycle or the Continuous Delivery: Agile lifecycle.
- Delivery of new functionality is truly continuous.
 - Changes to production are delivered several times a day, but the functionality may not be turned on until needed (a DevOps strategy called Feature Toggles).
- Automation and technical practices are key. This is similar to the Continuous Delivery: Agile lifecycle.
- Inception and Transition have disappeared from the diagram, for the same reasons they disappeared for Continuous Delivery: Agile.
- Explicit milestones and incoming workflows similar to the Continuous Delivery: Agile lifecycle.

DAD Lifecycles: Continuous Delivery – Lean



[Ambler & Lines 2020]



DAD Lifecycles: Exploratory (Lean Startup) (1)

- Based on the Lean Startup principles advocated by Eric Ries.
- The philosophy of Lean Startup is to minimize up-front investments in developing new offerings in the marketplace in favor of small experiments.
 - The idea is to run some experiments with potential customers to identify what they want based on actual usage.
- This is a simplified scientific method:
 - we come up with a hypothesis of what our customers want;
 - we develop one or more Minimal Viable Products (MVPs) which are deployed to a subset of potential customers;
 - we observe and measure how the customers work with the MVP(s);
 - based on the data we collect, we decide how we will go forward:
 - Pivot and rethink our hypothesis;
 - Rework MVPs to run new experiments based on our understanding of customer needs;
 - Discard one or more ideas;
 - Move forward with one or more ideas and “productize” them.

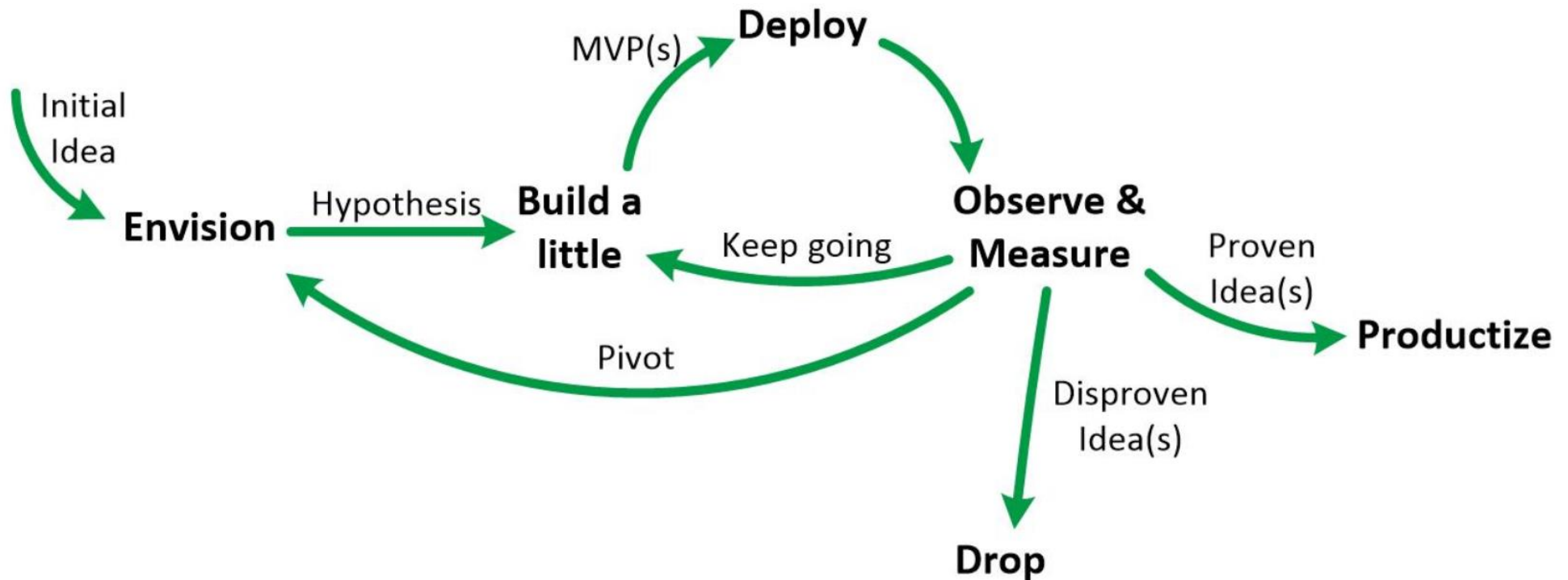


DAD Lifecycles: Exploratory (Lean Startup) (2)

- MVPs are prototypes (at best).
 - The MVPs we create are built hastily, often “smoke and mirrors” or prototype-quality code, whose sole purpose is to test out a hypothesis.
 - Delivering them as products is not a wise choice.
- Run several experiments in parallel.
 - Ideally this lifecycle entails running several experiments in parallel to explore our hypothesis.
 - This is an improvement over Lean Startup which focuses on a single experiment at a time:
 - it is easier to run a single experiment at a time, but it takes longer to get to a good idea and runs the risk of identifying a strategy before other options have been considered.
- Failed experiments are still successes as they actually reduce your risk of product failure.
- Follow another lifecycle to build the real product.



DAD Lifecycles: Exploratory (Lean Startup)



[Ambler & Lines 2020]



DAD Lifecycles: Program (1)

- Describes how to organize a team of teams.
 - Large agile teams are rare in practice, but they do happen. This is exactly the situation that scaling frameworks such as SAFe, LeSS, and Nexus address.
- Includes an explicit Inception phase for the large team to get organized.
- Subteams/squads choose and then evolve their WoW.
 - Subteams, sometimes referred to as *squads*, should be allowed to choose their own lifecycles as well as their own practices.
 - We may choose to impose some constraints on the teams, such as following common guidance and common strategies.
 - We will need to come to an agreement around how we'll proceed with cross-team system integration and cross-team testing (if needed).
- Subteams can be feature teams or component teams.
 - A feature team works vertical slices of functionality, implementing a story or addressing a change request from the UI all the way through to the database.
 - A component team works on a specific aspect of a system, such as security functionality, transaction processing, or logging.



DAD Lifecycles: Program (2)

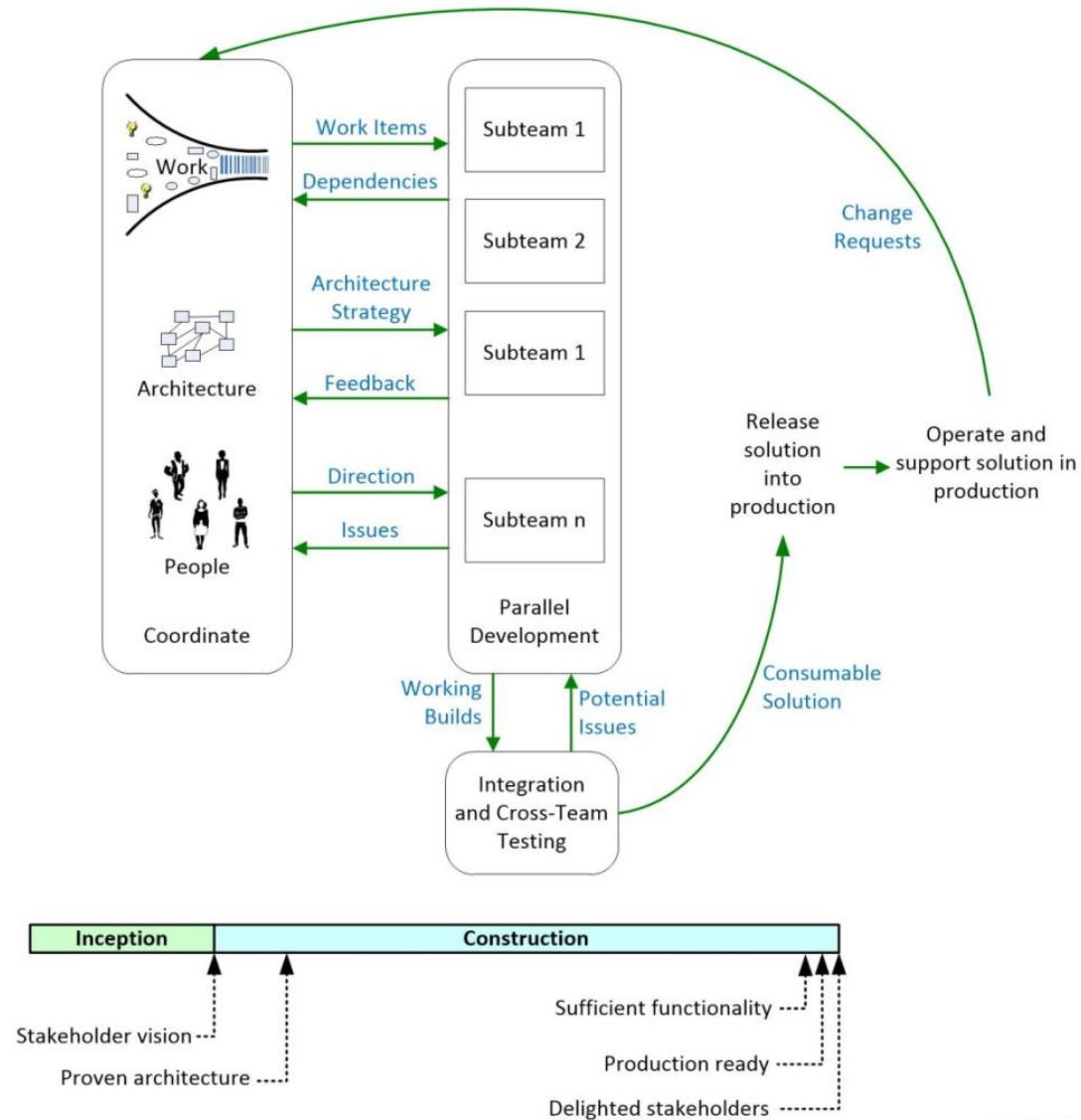
- Coordination occurs at three levels: Coordinating the work to be done, coordinating technical/architectural issues, and coordinating people issues.
 - This coordination is respectively performed by three leadership subteams, comprising the Product Owners, the Architecture Owners, and the Team Leads.
 - The team may need to get together occasionally to plan out the next block of work – this is done when and if it makes sense.
- System integration and testing occurs in parallel.
 - A separate team performs overall system integration and cross-team testing.
 - Ideally this work should be minimal and ideally entirely automated in time.
 - We often need a separate team at first, often due to lack of automation, but we should automate as much of the work as possible and push the rest into the subteams.
- Subteams are as whole as they can be.
 - The majority of the testing effort should occur within the subteams along with continuous integration (CI) and continuous deployment (CD).



DAD Lifecycles: Program (3)

- We can deploy any time we want.
 - We prefer a CD approach, but teams new to agile programs may start by releasing quarterly and then improve the release cadence over time.
 - Teams who are new to this will likely need a Transition phase, some people call these “hardening sprints” or “deployment sprints” the first few times.
- Scaling is hard.
 - Some problems require a large team, but to succeed you need to know what you’re doing: if you’re struggling with small-team agile then you’re not ready for large-team agile.
 - Team size is only one of seven scaling factors that we may need to contend with, the others being geographic distribution, domain complexity, technical complexity, organizational distribution, skill availability, and regulatory compliance.

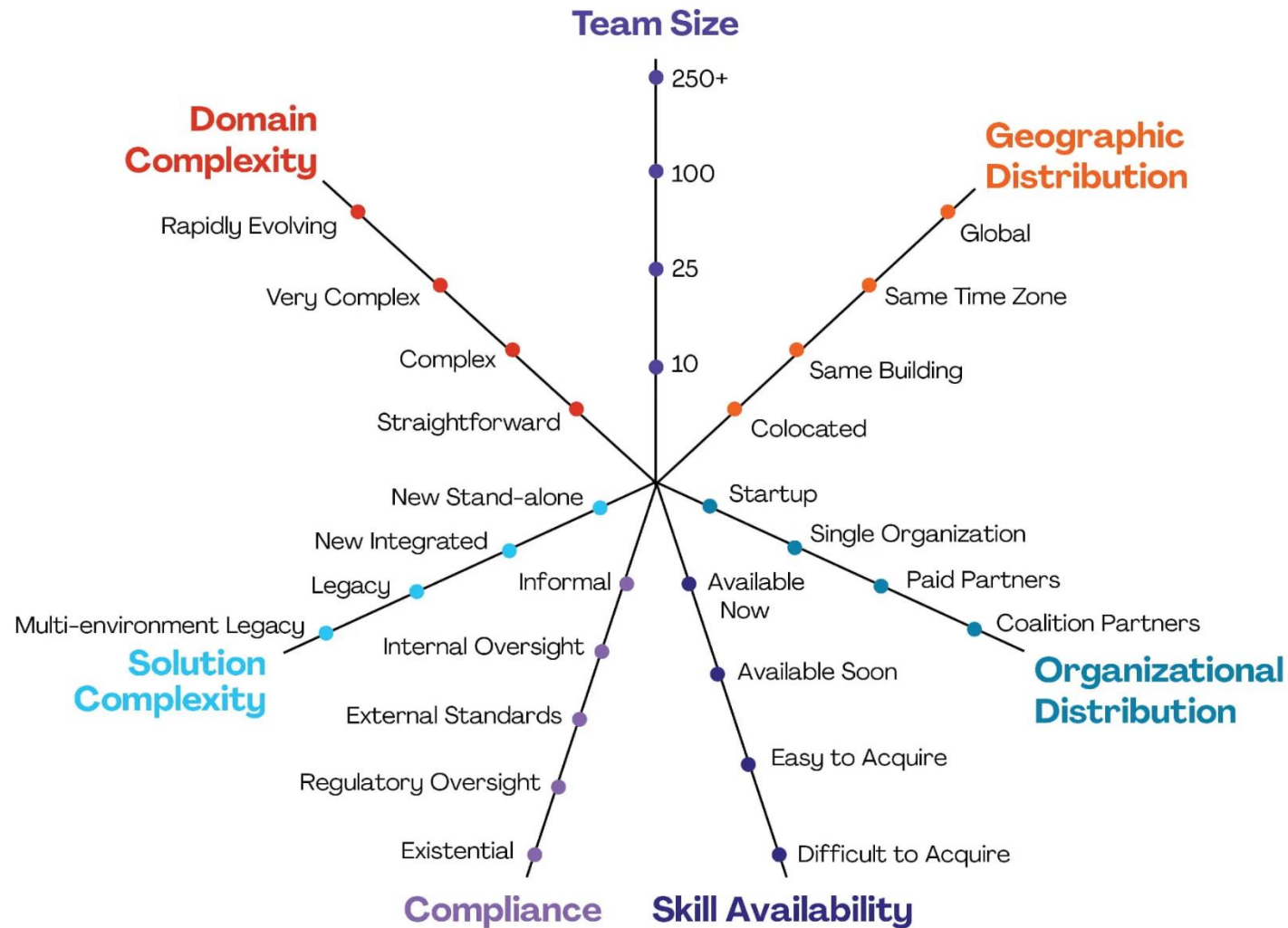
DAD Lifecycles: Program



[Ambler & Lines 2020]

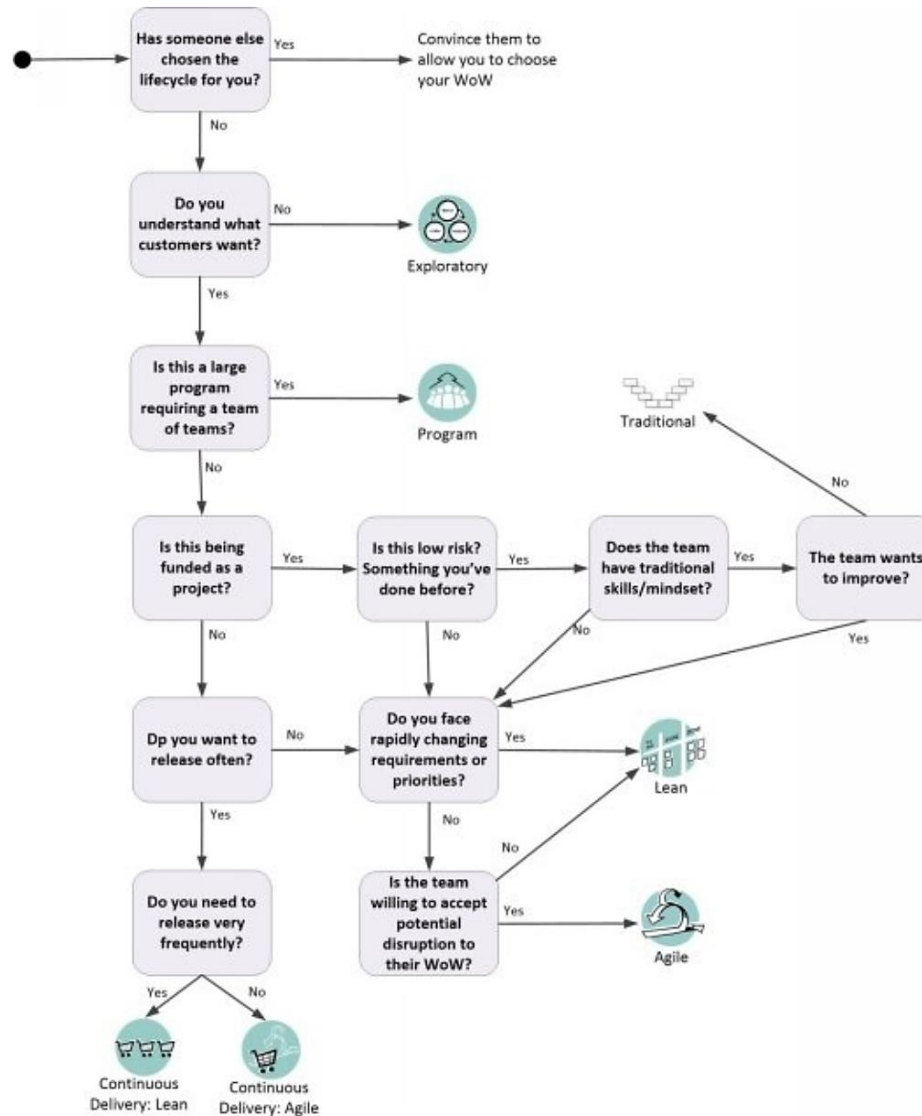


DAD Lifecycles: Program – Scaling Factors



©Project Management Institute. All rights reserved.

DAD: Choosing Your WoW (Way of Working)



[Ambler & Lines 2020]

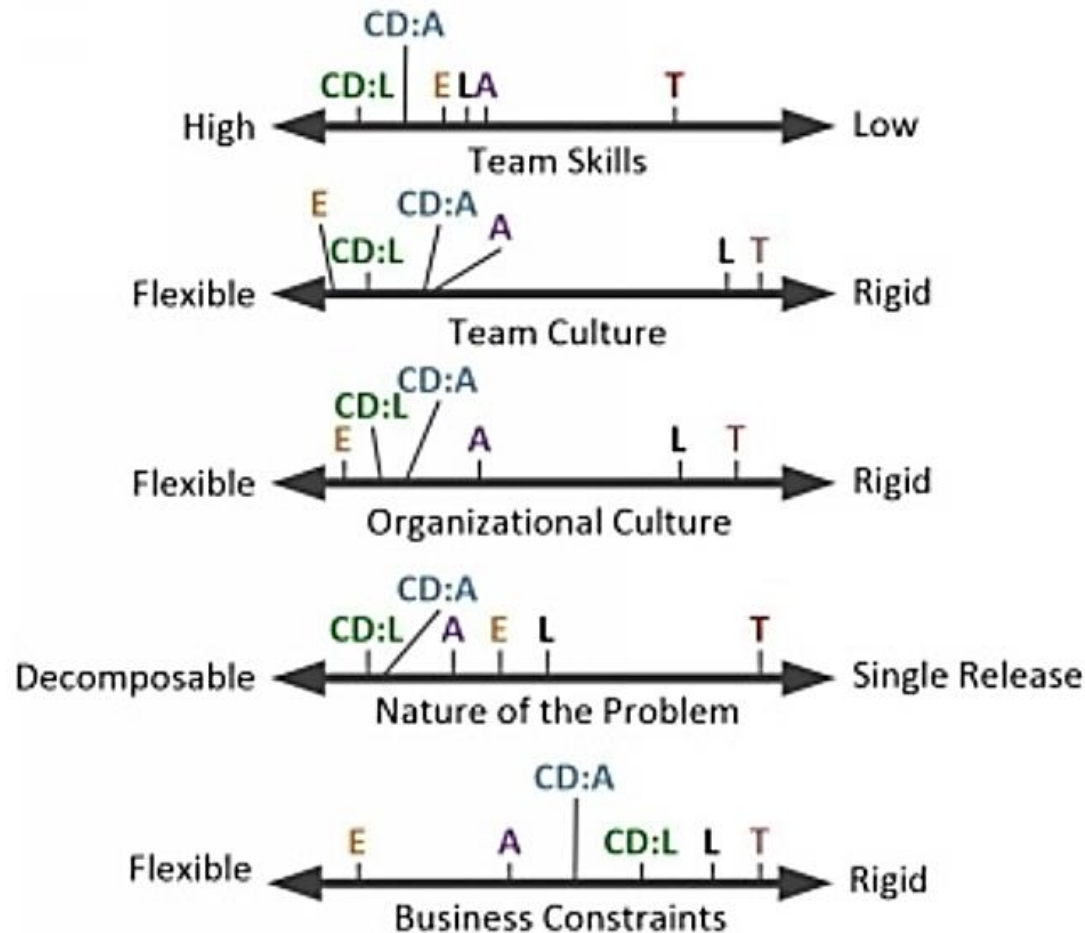


WoW: Selection Factors for Choosing a Lifecycle

- Team skills: Continuous Delivery (CD) lifecycles require skill and discipline.
- Team and organization culture: CD and Agile require flexibility within the team and within the organization; Lean can deal with rigid organizations.
- The nature of the problem: CD lifecycles work well for building and releasing in very small increments; other DAD lifecycles work well in small increments.
- Business constraints: The key issue here is Stakeholder availability and willingness, although financial/funding flexibility is also critical.
 - The Exploratory lifecycle requires a flexible, customer-oriented and experimental mindset on the part of Stakeholders.
 - Agile, because it tends to release functionality in terms of complete features, also requires flexibility in the way that we interact with Stakeholders.
 - CD lifecycles require less Stakeholder flexibility due to being able to release functionality that is turned off, thereby providing greater control over when something is released (by simply toggling it on).



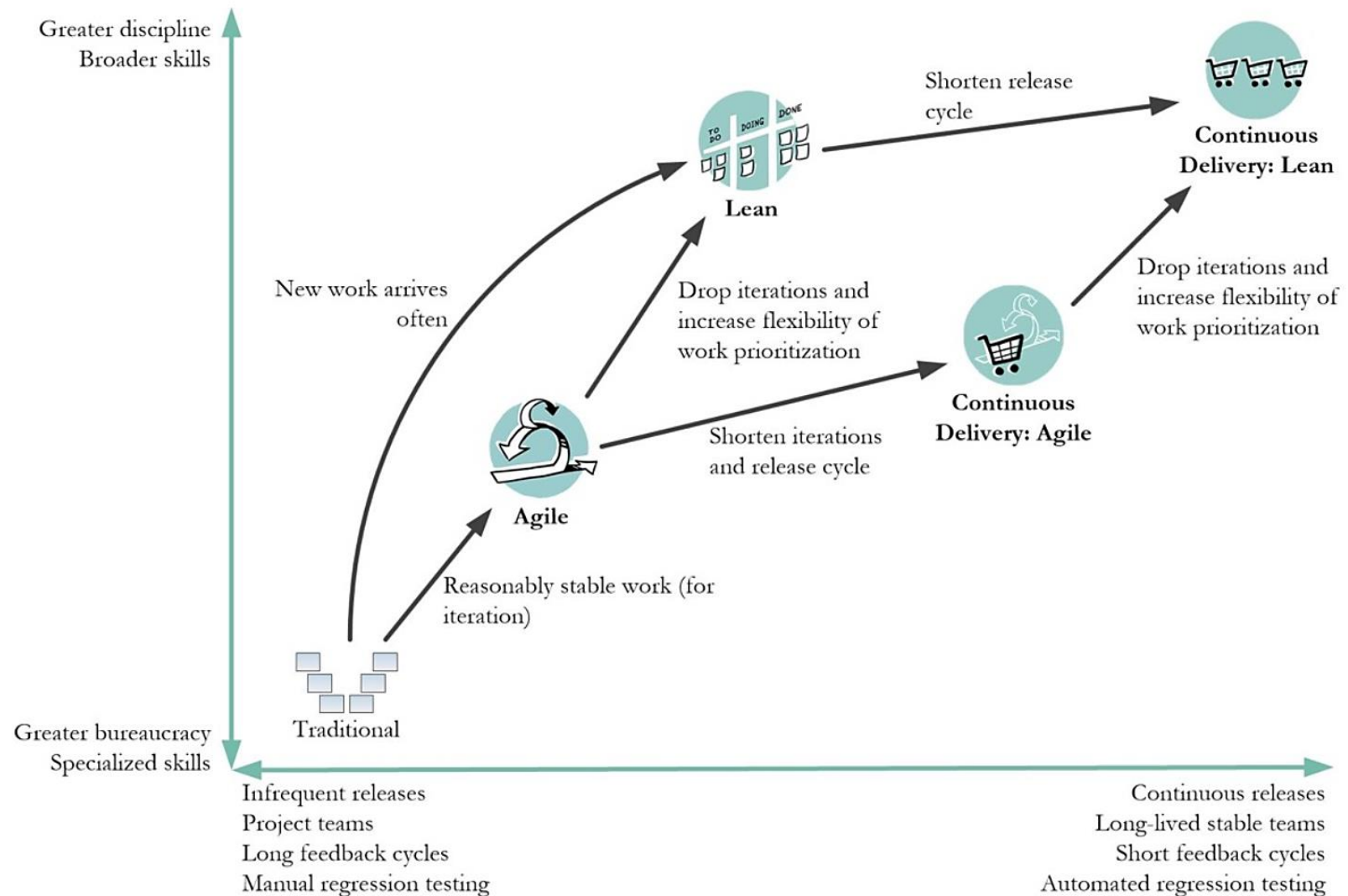
WoW: Selection Factors for Choosing a Lifecycle



A – Agile
 CD:A – Continuous Delivery: Agile
 CD:L – Continuous Delivery: Lean
 E – Exploratory
 L – Lean
 T – Traditional

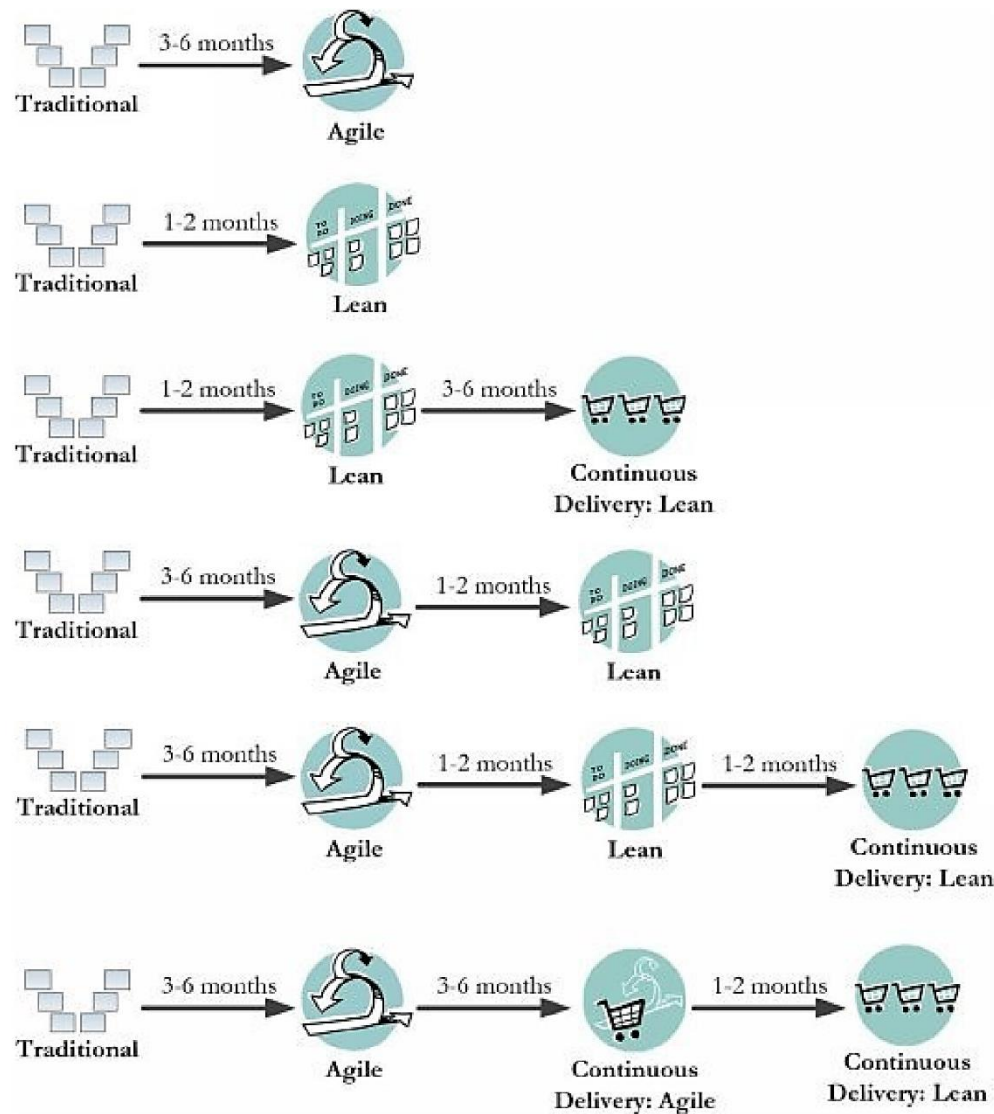
[Ambler & Lines 2020]

DAD: Evolving Lifecycles (1)



[Ambler & Lines 2020]

DAD: Evolving Lifecycles (2)



[Ambler & Lines 2020]



References

- Ambler, S.W., Lines, M., *Disciplined Agile Delivery*, IBM Press, 2012.
- Ambler, S.W., Lines, M., *Choose your WoW: A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working*, Project Management Institute, 2020.