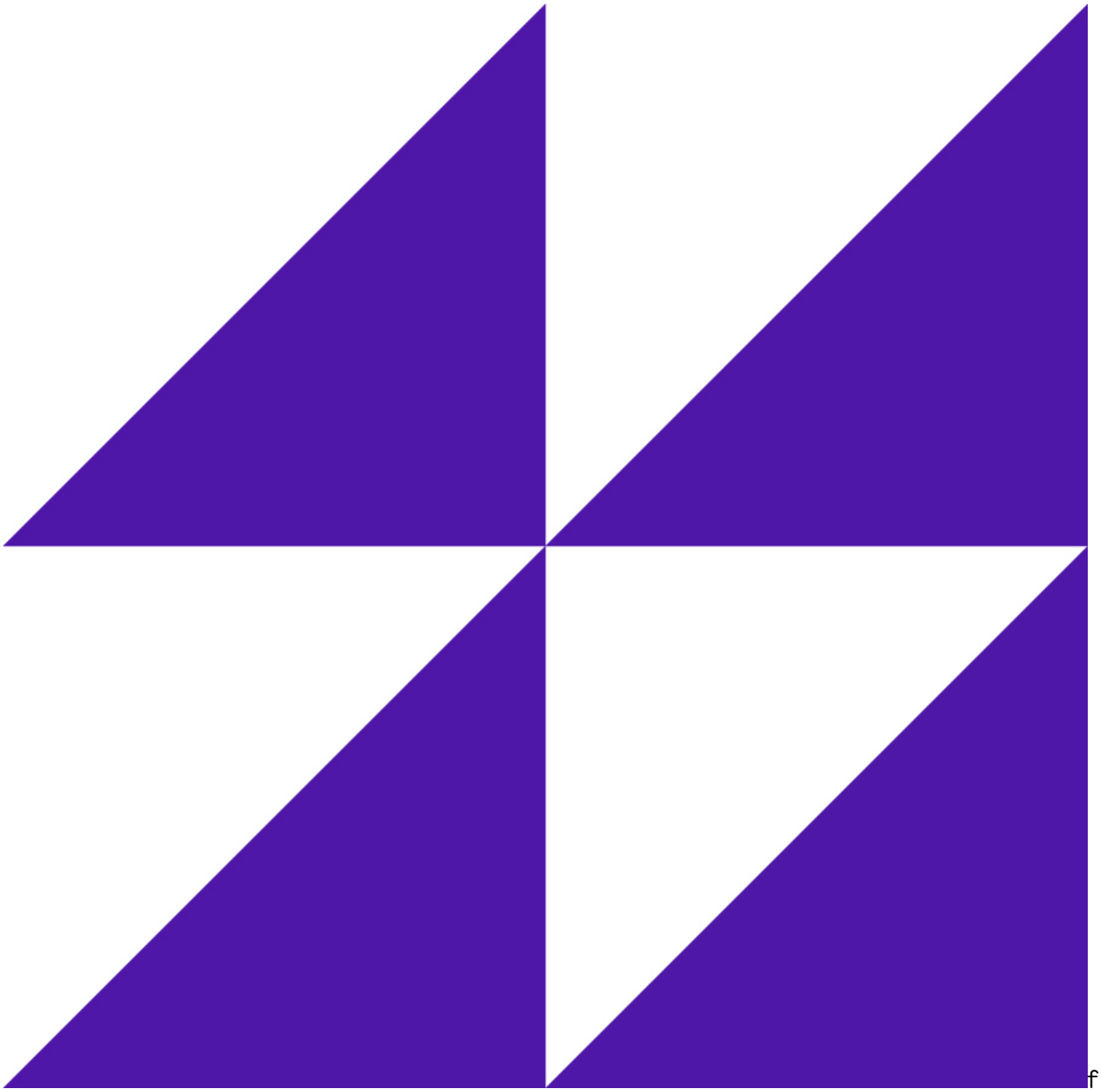
Disciplined Agile

Bibliography, References, & Glossary

This is for your own exploration and not required for the exam.



f

INNOVATION

New ideas. New products. New methods. Constant transformation is what helps us tackle new problems and find fresh solutions. We know that the world isn't one-size-fits-all. We bring new takes on everyday things to constantly move forward and make life better across the world

Bibliography

Ackoff, R. (2010, October 23). *What if Russ Ackoff gave a TED Talk?* Retrieved from You Tube: http://www.youtube.com/watch?feature=player_detailpage&v=OqEeIG8aPPk

Adkins, L. (2010). *Coaching Agile Teams: A Companion for ScrumMasters, Agile Coaches, and Project Managers in Transition.* Addison-Wesley Professional.

Ambler, S., & Lines, M. (2017). *An Executive's Guide to Disciplined Agile: Winning the Race to Business Agility.* CreateSpace Independent Publishing Platform.

Ambler, S., & Lines, M. (2019). *Choose Your WOW! A Disciplined Agile Delivery Handbook for optimizing your Way of Working.* Independently published.

Anderson, D. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business.* Blue Hole Press.

Anderson, J. (2013). *The Lean Change Method: Managing Agile Organizational Transformation Using Kanban, Kotter, and Lean Startup Thinking.* CreateSpace Independent Publishing Platform.

Argyris, C. (2015). *Chris Argyris, Bibliography of Works*. Retrieved from Action Science: Organizational Development for the 21st Century Manager: http://www.actionscience.com/argbib.htm

Beck, K. (2001). *Agile Manifesto*. Retrieved from Agile Manifesto: www.agilemanifesto.org

Bens, I. M., & MacCausland, J. (2012). *Facilitation at a Glance! A Pocket Guide of Tools and Techniques for Effective Meeting Facilitation.* Boston, MA: Goalqpc.

Bockman, S. (2008). *Team Estimation*. Retrieved from Agileworks: http://agileworks.blogspot.com/2008/01/team-estimation-game-by-steve-bockman.html

Bockman, S. (2015). *Practical Estimation: A Pocket Guide to Making Dependable Project Schedules.* Amazon Digital Services LLC .

Bridges, W. (2009). *Managing Transitions: Making the Most of Change.* De Capo Lifelong Books.

Brightline Project Management Institute. (2019). *Brightline Transformation Compass*. Retrieved from Brightline: www.brightline.org

Bungay, S. (2010). *The Art of Action.* Boston, MA: Nicholas Brealey Publishing.

Cameron, E. (2001). *Facilitation Made Easy: Practical Tips to Improve Meetings & Workshops* (2nd ed.). London, UK: Kogan Page.

Caruso, J. (2004). *The Power of Losing Control.*

Cockburn, A. (2000). *Writing Effective Use Cases.* Addison-Wesley Professional.

Cockburn, A. (2012, May 01). *Steps to an Agile Frame of Mind*. Retrieved June 01, 2014, from Alistair Cockburn: http://alistair.cockburn.us/Steps+to+an+Agile+Frame+of+Mind+course/v/slim

CollabNet VersionOne. (2019, May 7). *CollabNet VersionOne Releases 13th Annual State of Agile Report*. Retrieved from CollabNet: https://www.collab.net/news/press/collabnet-versionone-releases-13th-annual-state-agile-report

Collison, C., & Parcell, G. (2004). *Learning to Fly: Practical knowledge management from some of the world's leading learning organizations.* Chichester, West Sussex: Capstone.

Deming, W. E. (1993). *The New Economics for Industry, Government, Education.* The MIT Press.

Deming, W. E. (2013). *Quotes by Deming*. Retrieved from Deming blog: https://blog.deming.org/w-edwards-deming-quotes/large-list-of-quotes-by-w-edwards-deming/

Denne, M., & Cleland-Huang, J. (2003). *Software by Numbers: Low-Risk, High-Return Development.* Upper Saddle River, NJ: Prentice Hall PTR.

Derby, E. (2006). *Agile Retrospectives: Making Good Teams Great.* Pragmatic Bookshelf.

Farrell, J. D., & Weaver, R. G. (2000). *Practical Guide to Facilitation: A Self-Study Resource.* Amherst, MA: HRD Press.

Garvin, D. A. (July-August 1993). Building a Learning Organization. *Harvard Business Review*.

Goldratt, E. (2018, July 30). *Goldratt Books*. Retrieved from Twitter: https://twitter.com/goldrattbooks/status/1023916431036108800

Gottlieb, M. R. (2003). *Managing Group Process.* Santa Barbara, CA: Praeger Publishers.

Graesser, A. C., Fiore, S. M., & Greiff, S. (2018). *Sage Journals*. Retrieved from Advancing the Science of Collaborative Problem Solving: https://journals.sagepub.com/stoken/default+domain/10.1177%2F1529100618808244-free/full

Guernsey III, M. (2013). *Test-Driven Database Development: Unlocking Agility.* Addison-Wesley Professional.

Heath, C., & Heath, D. (2010). *Switch: How to Change Things When Change Is Hard.* Crown Business.

Herzberg, F., Mausner, B., & Snyderman, B. B. (1959). *The Motivation to Work.* John Wlley.

Highsmith, J. (2002). *Agile Software Development Ecosystems.* Addison-Wesley Professional.

Hofer-Alfeis, D., Klementz, G., & Krause, H. (2001). Corporate Knowledge Management: The Organization for Knowledge Management and Business Transformation in a Large Enterprise. *APQC Fourth Knowledge Management Conference* (pp. 89-94). Houston, TX: American Productivity and Quality Center.

Hogan, C. (2002). *Understanding Facilitation: Theory & Principles.* London, UK: Kogan Page.

Hohmann, L. (2006). *Innovation Games: Creating Breakthrough Products Through Collaborative Play.* Boston, MA: Addison-Wesley Professional.

Hope, J., & Fraser, R. (2003). *Beyond Budgeting: How Managers Can Break Free from the Annual Performance Trap.* Harvard Business Review Press.

Karl Rahner. (2015, October 20). Retrieved October 20, 2015, from https://en.wikipedia.org/wiki/Karl_Rahner

Keith, N. L. (2001). *Project Retrospectives: A Handbook for Team Reviews.* Dorset House.

Kirkpatrick, D. L. (1985). *Kirkpatrick, D. How to Manage Change Effectively: Approaches, Methods, and Case Examples* . Jossey-Bass.

Kotter, J. P. (2012). *The Heart of Change: Real-Life Stories of How People Change Their Organizations.* Harvard Business Review Press.

Ladas, C. (2011). *Scrumban: Essays on Kanban Systems for Lean Software Development.* Modus Cooperandi Press.

Larman, C. (2003). *Agile and Iterative Development: A Manager's Guide.* Addison-Wesley Professional.

Leas, S. B. (1998). *Discover Your Conflict Management Style.* Rowman & Littlefield Publishers.

Lencioni, P. (2011). *The Five Dysfunctions of a Team: A Leadership Fable.* Jossey-Bass.

Lines, M., & Ambler, S. (2018). *Introduction to Disciplined Agile Delivery: A small Agile Team's Journey from Scrum to Disciplined DevOps. , 2nd Edition.* CreateSpace Independent Publishing Platform.

Little, J. (2014). *Lean Change Managment: Innovative Practices For Managing Organizational Change.* Happy Melly Express.

Lucas, R. W. (2003). *The Creative Training Idea Book: Inspired Tips and Techniques for Engaging and Effective Learning.* New York, NY: AMACOM.

Mann, D. (2010). *Creating a Lean Culture: Tools to Sustain Lean Conversions, 2nd Edition.* New York, NY: Productivity Press.

Marquardt, M. J., & Davies, L. H. (2012). *Global Teams: How Top Multinationals Span Boundaries and Cultures with High-Speed Teamwork.* Boston, MA: Nicholas Brealey America.

Mascitelli, R. (2002). *Building a Project-Driven ENterprise: How to slash waste and boost profits through lean project management.* Northridge, CA: Technology Perspectives.

Maurer, R. (2004). *One Small Step Can Change Your Life: The Kaizen Way.* Workman Publishing Company.

Maurya, A. (2012). *Running Lean: Iterate from Plan A to a Plan That Works.* O'Reilly Media, Inc.

McKergow, M., & Bailey, H. (2014). *Host: Six new roles of engagement for teams, organizations, communities, movements.* London: Solutions Books.

Nonaka, I. (1988). Toward Middle-Up-Down Management: Accelerating Information Creation. *Sloan Review*, https://sloanreview.mit.edu/article/toward-middleupdown-management-accelerating-information-creation/.

Ohno, T. (1988). *Toyota Production System: Beyond Large-Scale Production.* Productivity Press.

Paulenn, D. (2003). *Virtual Teams: Projects, Protocols, and Processes.* Hershey, PA: Idea Group Publishing.

Pink, D. H. (2011). *Drive: The Surprising Truth About What Motivates Us.* Riverhead Books.

Pugh, K. (2005). *Prefactoring.* O'Reilly Media.

Rackham, N. (1988). *SPIN Selling.* McGraw-Hill Education.

Reinertsen, D. (2009). *The Principles of Product Development Flow: Second Generation Lean Product Development.* Redondo Beach, CA: Celeritas Publishing.

Rostron, S. S. (2002). *Accelerating Performance: Powerful Net Techniques to Develop People.* London, UK: Kogan Page.

Rother, M. (2009). *Toyota Kata: Managing People for Improvement, Adaptiveness and Superior Results.* McGraw-Hill Education.

Scholtes, P. R. (1998). *The Leader's Handbook: Making Things Happen, Getting Things Done.* New York: McGraw-Hill.

Scott Ambler and Associates. (2013, July). *2013 Agile Project Initiation Survey Results*. Retrieved from Ambysoft: http://www.ambysoft.com/surveys/projectInitiation2013.html

Shalloway, A. (2008). *Lean Anti-Patterns and What to Do About Them.* Retrieved from Agile Journal: www.agilejournal.com/content/view/553/39

Shalloway, A., & Trott, J. R. (2004). *Design Patterns Explained* (Second Edition ed.). Addison-Wesley Professional.

Shalloway, A., & Trott, J. R. (2014). *Lean-Agile Pocket Guide for Scrum Teams.* Net Objectives Press.

Shalloway, A., & Trott, J. R. (2016). *Leanban Primer: Lean Software Development at the Team Level.* Seattle, WA: Net Objectives Press.

Shalloway, A., Bain, S., Pugh, K., & Kolsky, A. (2011). *Essential Skills for the Agile Developer.* Addison-Wesley Professional.

Shalloway, A., Beaver, G., & Trott, J. R. (2009). *Lean-Agile Software Development: Achieving Enterprise Agility.* Addison-Wesley Professional.

Shore, J., & Warden, S. (2008). *The Art of Agile Development.* Sebastopol, CA: O'Reilly Media, Inc.

Sinek, S. (2011). *Start with Why: How Great Leaders Inspire Everyone to Take Action.* Portfolio.

Snowden, D. (2014, December 26). *The Strathern variation.* Retrieved from Cognitive Edge: https://cognitive-edge.com/blog/the-strathern-variation/

Straus, D. (2002). *How to Make Collaboration Work: Powerful Ways to Build Consensus, Solve Problems, and Make Decisions.* San Francisco, CO: Berrett-Koehler Publishers.

Takeuchi, H., & Nonaka, I. (1986). *The New New Product Development Game.* Retrieved from Lean Enterprise Institute: http://www.iei.liu.se/fek/svp/723g18/articles_and_papers/1.107457/TakeuchiNonaka1986HBR.pdf

Teasley, S., Covi, L., Krishnan, M., & Olson, J. S. (2000). How Does Radical Collocation Help a Team Succeed? *Communications of the ACM*, 339-346. Retrieved from http://possibility.com/Misc/p339-teasley.pdf

Thomas, K. W., & Killmann, R. H. (1974). *The Thomas-Kilmann Conflict Mode Instrument.* Mountain View, CA: CPP, Inc.

Townsend, P. L., & Geghardt, J. E. (2007). *How Organizations Learn: Investigate, Identify, Institutionalize.* Milwaukee, WI: ASQ Quality Press.

Trott, J. (1999). *The Search Behavior of Engineers.* Seattle: Unpublished.

Tumonis, W. (2014, November 21). *How Social Loafing Hurts Group Creativity and How to Reduce It.* Retrieved from Ideation Wiz: https://www.ideationwiz.com/social-loafing-and-group-creativity/

Weinberg, G. M. (1986). *The Secrets of Consulting: A Guide to Giving and Getting Advice Successfully.* Dorset House Publishing.

Weinberg, G. M. (1991). *Quality Software Management: Systems Thinking.* Dorset House.

Wheatley, M., & Kellner-Rogers, M. (1998). *A Simpler Way.* Berrett-Koehler Publishers.

Womack, J. P., Jones, D. T., & Roos, D. (2007). *The Machine That Changed the World: The Story of Lean Production-- Toyota's Secret Weapon in the Global Car Wars That Is Now Revolutionizing World Industry.* Free Press.

Resources by Category

Related to an agile way of working

Scrum

- https://hbr.org/1986/01/the-new-new-product-development-game
- https://scrumguides.org/scrum-guide.html

DA FLEX

- https://portal.netobjectives.com/pages/flex/

Disciplined Agile roles

- https://www.pmi.org/disciplined-agile/people/roles-on-dad-teams

Visualization

- https://visualizationexamples.com/

Metrics

- https://www.projectmanagement.com/blog-post/61916/Agile-Metrics--Questions-and-Answers
- https://www.projectmanagement.com/blog-post/61874/Goal-Question-Metric--GQM--and-Agile

Kaizen and Plan-Do-Study-Act Cycle

- https://en.wikipedia.org/wiki/Kaizen
- https://en.wikipedia.org/wiki/PDCA

Agile Retrospectives

- https://pragprog.com/titles/dlret/agile-retrospectives
- https://www.youtube.com/watch?v=w8w-dFrmovQ
- https://www.tastycupcakes.org/
- https://www.funretrospectives.com/

User Stories

- https://www.mountaingoatsoftware.com/books/user-stories-applied

Related to coaching

- *Coaching Agile Teams: A Companion for ScrumMasters, Agile Coaches, and Project Managers in Transition*, Lyssa Adkins
- *Agile Coaching*, Rachel Davies, Liz Sedley

Related to emotional intelligence

- *Emotional Intelligence: Why It Can Matter More Than IQ*, Daniel Goleman
- *Emotional Capitalists: The New Leaders*, Martyn Newman
- *The EQ Edge: Emotional Intelligence and Your Success*, Steven Stein, Howard Book
- *Primal Leadership: Unleashing the Power of Emotional Intelligence*, Daniel Goleman, Richard Boyatzis, Annie McKee
- *Hardwiring Happiness: The New Brain Science of Contentment, Calm, and Confidence*, Rick Hanson
- *One Second Ahead: Enhance Your Performance at Work with Mindfulness*, Rasmus Hougaard, Jacqueline Carter, Gillian Coutts
- *Flow: The Psychology of Optimal Experience*, Mihaly Csikszentmihalyi
- *Thinking, Fast and Slow Paperback*, Daniel Kahneman

Related to overcoming imposter syndrome

- Sakulku, Jaruwan (2011). "The Impostor Phenomenon." *International Journal of Behavioral Science*. 6 (1): 73–92
- Clark, M.; Vardeman, K.; Barba, S. (2014). "Perceived inadequacy: A study of the impostor phenomenon among college and research librarians." *College & Research Libraries*. 75 (3): 255–271.

Disciplined Agile Glossary

Term	Description
Acceptance Test-Driven Development	Acceptance Test-Driven Development (ATDD) employs a test-first mindset where Business, developers, and testers work closely together to write acceptance tests before implementation begins. ATDD is very similar to Behavior Driven Development and are discussed together.
Acceptance Testing	Formal testing conducted to determine whether a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system.
Agile	Agile software development is a conceptual framework for undertaking software engineering projects that embraces and promotes evolutionary change throughout the entire life cycle of the project. Scrum and XP are two software development methods based on Agile tenets. See also: Scrum, XP.
Agile Coach	A coach is a specialist role where the focus is on guiding people through their learning journey. Types of agile coach include: • Team coach • Specialized Coach (e.g. Agile DW/BI Coach, Agile Marketing Coach) • Enterprise Coach
Agility at scale	The Disciplined Agile tool kit supports both types of agility at scale: • **Tactical agility at scale**. This is scaling agile at the team level. The aim is to apply agile deeply to address all the complexities/scaling factors (team size, geographic distribution, organizational distribution, domain complexity, technical complexity, and regulatory compliance) appropriately. • **Strategic agility at scale.** This is scaling agile at the organizational level. This is the application of agile and lean strategies broadly across your entire organization. This includes all divisions and teams within your organization, not just your IT department.
Architecture Owner	The architecture owner is the person who owns the architecture decisions for the team and who facilitates the creation and evolution of the overall solution design. It may not be necessary to formally designate a team member as an architecture owner on small teams. The person in the role of team lead will often also be in the role of architecture owner. This isn't always the case, particularly at scale, but it is very common for smaller agile teams.
Backlog	A list of work to be completed that has various degrees of commitment associated with it depending upon where it is being used.

BDD	Behavior Driven Development is a method of defining the behavior required prior to doing design or implementation. It is very similar to Acceptances Test-Driven Development except that it typically is done in the Given <an event> When <a situation> Then <the desired behavior> (GWT). We discuss both together.
Build Verification Test	A group of tests to determine the health of a build at a high level. Typically, these tests exercise the core functionality of code and help determine whether further testing is warranted. These are also known as "smoke tests."
Burn-down Chart	A chart showing the rate (story points/day) at which stories or tasks are being completed.
Burn-up Chart	A chart showing the rate (story points/day) at which business value is growing.
Business	The "Business" is a shorthand for someone from another part of the organization who is not part of the technical development team but has expertise and/or responsibility for profitably addressing customers and markets.
Business Agility	Business agility is the quick realization of business value predictably, sustainably and with high quality. It is the ability of an organization to rapidly adapt to market and environmental changes in productive and cost-effective ways. Business agility focuses on value realized by having stakeholders identify, prioritize and sequence the work to be done and allocate it appropriately to the product/service teams. Sometimes referred to as enterprise agility.
Business Value	Business value is what drives product development. It is the measurable or intangible return the Business anticipates it can realize from the product. The Business is responsible for identifying Business value. Business value is what the Business is willing to pay for. Development work should always be traceable to Business value. Here are types of business value. • Value delivered to the customer • Discovering what is of value • Discovering how to build it • Building it • Preparing for consumption • Improving our own internal methods • Mitigating risk
Cadence	The flow or rhythm of events, especially the pattern in which something is experienced. In Lean-Agile, cadence is the rhythm of when backlogs are refreshed, work is readied for synchronization, retrospections are done, demos are presented, and teams reflect on their work.

Capability	The combination of people and people skills, processes, and technologies required to equip the Business / customer to use a product. All three are required since until the Business can begin to use a product, the product merely represents potential.
Center of Excellence	Typically, a group of experts whose job is to educate, coach, and mentor people. Sometimes called a Community of Experts.
Code Analysis	The process of checking that code conforms to design guidelines, looking for common coding and design errors per coding standards. The team makes an agreement to conform to these coding standards; thus, code analysis serves an integrity check with how well the team is working according to their standards.
Code Coverage	A metric used to describe the degree to which source code has been tested. Code coverage is expressed as a percentage of lines of code tested over the total lines of code.
Cognitive Bias	Cognitive bias occurs when an individual sees things from his own subjective view instead of seeing as it really is. Humans automatically see the world through their own filter. When this happens, risks are often not noticed nor attended to.
Colocated Team	A team in which everyone on the team sits in the same room. With "pure" Co-location: • The team has their own, dedicated room. • Key stakeholders are also in the room, at least part time every day. Team members often have access to "quiet rooms" where they can occasionally have private conversations. See also *Distributed Sub-team*, *Fully Dispersed Team*, *Near-located Team*, *Partially Dispersed Team*, *Sub-team*.
Collaboration	Collaboration benefits organizations, teams, and individuals by enabling colleagues to work together to create new ideas from a starting point that is: uncertain, poorly understood, or innovative; adaptive or intangible; or requires change to accomplish. Collaboration assumes that no one has the best, correct, or full answer and that the solution can only be found by exploring and synthesizing diverse points of view and using thought experiments and what-if scenarios. Collaboration is creative and the work of humans not machines.
Commonality / Variability Analysis (CVA)	A technique that enables developers to write code that can be easily modified later. It complements iterative development. It is described in Shalloway and Trott's Design Patterns Explained: A New Perspective of Object-Oriented Design.
Community of Practice	A group that focuses on sharing and enhancing skills within a group of like-minded people organized on a **volunteer basis.** CoPs may adopt common

	strategies include internal discussion forums, training sessions, "brownbag lunches", and even be involved in certification of practitioners.
Constraint	Anything that limits a system from achieving higher performance or throughput; also, the bottleneck that most severely limits the organization's ability to achieve higher performance relative to its purpose or goal.
Complex Adaptive System	A complex adaptive system is a system in which a perfect understanding of the individual parts does not automatically convey a perfect understanding of the whole system's behavior.
Continuous Process Improvement	A philosophy and attitude for analyzing capabilities and processes and improving them repeatedly to achieve customer satisfaction. Also known as Continuous Quality Improvement.
Cooperation	Cooperation benefits organizations, teams, and individuals by spreading the work across numerous functions, abilities, and methods (e.g., human and machine can cooperate to produce work). Cooperation assumes that the work to be done is well understood, generally predictable, and has an accepted action plan. It is about helping to execute an already determined plan. Cooperation improves efficiency.
Cost-Benefit Analysis	A method of project evaluation that compares the potential benefits with the anticipated costs.
Cost of Delay	The cost, usually in terms of lost revenue or opportunity, caused by the delay between conceiving the idea and having customers realize value from it. This can be expressed both as a total anticipated amount at the start or an ongoing basis as the project goes forward. For example, if a product will achieve a revenue of $100,000 a month, then taking two months to get it available costs $200,000. (Reintertsen 2009) says, "If you only quantify one thing, quantify the Cost of Delay."
Coupling and cohesion	**Coupling** refers "to the strength of a connection between two routines or classes. Cohesion describes how strongly the internal contents of a routine or class are related to each other. **Cohesion** refers to how "closely the internal elements or operations in a routine or class are related." Some people refer to cohesion as "clarity" because the more operations are related in a routine or class the easier it is to understand the code and what it's intended to do. Coupling and cohesion are complements of each other. The goal is to create architectures and routines and classes with internal integrity (strong cohesion) and small, direct, visible, and flexible relations to other routines and classes (loose coupling)."

Cross-Functional Team	Teams that have all the capabilities to deliver the work they've been assigned. Team members can specialize in certain skills but, the team is capable of delivering what they've been called on to build.
CRUFT	Test case documentation suffers from the CRUFT challenges associated with all forms of documentation. CRUFT is a mnemonic to express the effectiveness of an artefact. Each letter stands for a variable of effectiveness. Multiply the variables together to get the score. **C** The percentage of the artifact that is currently "**correct**". **R** The chance that the artifact will be **read** by the intended audience. **U** The percentage of the artifact that is **understood** by the intended audience. **F** The chance that the material contained in artifact will be **followed**. **T** The chance that the document will be **trusted**. Example, if each of these were 0.95 (very high), the score would be 0.77! The only way we can write effective documentation is if we know what stakeholders actually need and how they will work with the deliverable documentation that we produce.
Customer	The recipient of the output (product, service, information) of a process. Customers may be internal or external to the organization. The customer may be one person, a department, or a large group. Internal customers (outside of IT) are sometimes called the "Business."
Customer Satisfaction	The result of delivering a product, service, or information that meets customer requirements.
Cycle Time	The total elapsed time to move a unit of work from the beginning to the end of a process. Cycle time includes process time, during which a unit is acted upon to bring it closer to an output, and delay time, during which a unit of work is spent waiting to take the next action.
Daily Coordination Meeting	A regular, short meeting of the team where status is exchanged, progress is observed, and impediments are noted and removed. There are many approaches to do this. Also known as the Daily Standup.
Definition of Done	The criteria for accepting an MBI, feature, or story as finished. Specifying these criteria is the responsibility of the entire team, including the business.
Demonstrable Product	A version of a product that can be demonstrated to customers. It may not be quite ready for release or delivery, since that usually requires additional work (such as artwork and production plans). It is the primary deliverable of an iteration.
Design Pattern	A collection of best practices for solving problems in a recurring context. The more general term is "pattern" because patterns are involved with analysis, design, implementation, and testing.

Disciplined Agile	Disciplined Agile (DA) is a process-decision tool kit that provides straightforward guidance to help people, teams, and organizations to streamline their processes in a context-sensitive manner.
Disciplined Agile Enterprise (DAE)	A Disciplined Agile Enterprise (DAE) is able to sense and respond swiftly to changes in the marketplace. It does this through an organizational culture and structure that facilitates change within the context of the situation that it faces. Such organizations require a learning mindset in the mainstream business and underlying lean and agile processes to drive innovation.
Disciplined Agile Information Technology (DAIT)	The Information Technology (IT) process blade encapsulates the activities required to provide IT capabilities to the rest of the organization. It is part of the Disciplined Agile Enterprise (DAE) layer of the Disciplined Agile (DA) tool kit.
Disciplined Agile FLEX	Disciplined Agile Flow for Enterprise Transformation (DA FLEX) is an approach based on Lean-Thinking and process patterns that improves an organization's ability to achieve business agility – the quick realization of value reliably, sustainably and with high quality.
Disciplined DevOps	Disciplined DevOps is the streamlining of IT solution development and IT operations activities, along with supporting enterprise-IT activities such as Security and Data Management, to provide more effective outcomes to an organization.
Distributed Sub-team	A sub-team in which collaboration occurs as a cross-functional, whole team. Some coordination will be required between sub-teams. See also *Colocated Team, Fully Dispersed Team, Near-located Team, Partially Dispersed Team, Sub-team.*
Emergent Design	Allowing a design to emerge over time, as part of the natural evolution of a system. Requires good practices and testing to ensure that the system is not inadvertently allowed to decay or become overly complex over time. See Emergent Design: The Evolutionary Nature of Professional Software Development by Scott L. Bain.
Error Proofing	The ability to catch errors immediately, and to take corrective action to prevent problems down the line. Four strategies for error proofing are: 1) eliminate possibility of error, 2) reduce occurrence if elimination is not possible, 3) reduce consequences of errors, 4) capture and address error early if they cannot be eliminated.
Extreme Programming	See: XP.
Facilitator	An individual specifically trained to enable groups to work more effectively, collaborate, and achieve synergy. The facilitator is a 'content neutral' party; does not take sides or express or advocate a point of view during the meeting, advocates for fair, open, and inclusive procedures to accomplish the group's work.

Feature	A feature is a business function that the product carries out. Features are large and chunky and could be implemented by using many stories. Features may be functional or non-functional. Features form a basis for organizing stories.
FIT	Framework for Integrated Test
Flow	Flow is the ability to produce value continuously. A flow-based process delivers information on a regular cadence in small batches. In fact, cadence helps lower transaction costs and makes small batches more economically feasible. Flow focuses on reducing cost of delay by realizing the most important business value quickly. This also reduces risk.
Fully Dispersed Team	A team in which everyone works from different locations. This requires significant coordination between people when the work requires collaboration. It works well when the work is piecemeal with little dependencies between tasks. It enables a work from anywhere, anytime approach. It often requires people to gather at critical times to coordinate and plan the next tranche of work. See also *Colocated Team*, *Distributed Sub-team*, *Near-located Team*, *Partially Dispersed Team*, *Sub-team*.
Functional Test	The activity that validates a feature against customer requirements. Functional tests are usually done by a tester as part of the customer team.
Gemba	The "Gemba" is the place where value-added work is being done: a work cell, the developer team room, the help desk, the customer's office. Management must go to these locations to observe, evaluate, coach, and engage with the team. This contrasts with management practices that rely on management hierarchies, team leads or formal status meetings in conference rooms or management offices.
Generalizing specialist	A generalizing specialist is someone who: has one or more specialties; has at least a general knowledge of the overall process; has at least a general knowledge of the business domain in which they work; actively seeks to gain new skills in both their existing specialties as well as in other areas; and shares their skills and knowledge with others.
Good Practice	A superior method or innovative practice that contributes to the improved performance of an organization, usually recognized as "best" by other peers for given contexts. It is often worthwhile to focus on "good" practices rather than striving for "best" practices.
GQM	Goal Question Metric. Is an approach to measuring progress on software. It defines three levels: The goal (conceptual level), the question (at the operational level), and a set of quantitative metrics that help assess progress in a measurable way. See also *KPI* and *OKR*.
Harness Tests	A group of test scenarios with expected outcomes related to a specific system module to confirm that defects have not been introduced because of

	current iteration programming activity. For example, a pricing test harness would confirm no defects from current pricing module programming.
Impediment	A technical, personal, or organizational issue that is preventing or delay in progress on delivering product.
Information Radiator	A type of visual control that displays information in a place where passersby can see it and get information about the project without having to ask questions. To be effective, the information must be current and must be easy to see and understand, with sufficient detail to explain status.
Intangible	Intangible metrics and deliverables are indirect drivers of change. For example, culture, leadership, behaviors, mental models, assumptions, teaming dynamics, social physics, and the ability to work on the organization. Intangible drivers of change are used to positively impact technical delivery and, when ignored, often have a negative impact on technical delivery and results.
Integration System Test	The activity that verifies that software code does not harm other parts of the software product. Preferably, Integration system testing is done by a tester with automated tools.
Inventory	Work items that have been started but not completed.
Iteration	A time-boxed period during which the team is focused on producing a demonstrable product, some amount of functionality that is ready to be shown to the customer and potentially ready to be delivered. Usually, iterations are 2-4 weeks long. In Scrum, an iteration is called a "sprint."
Iteration Plan	The list of user stories for the upcoming iteration.
Iteration Planning	The activity to prioritize and identify the stories and concrete tasks for the next iteration. Also known as "loading the front burner."
Just-in-Time Design	The process of waiting until you know what the design needs to be and then refactoring code to meet these new needs before adding the functionality that is forcing the design change.
Kaizen	A Japanese term that means gradual unending improvement by doing little things better and setting and achieving increasingly higher standards. Masaaki Imai made the term famous in his book, Kaizen: The Key to Japan's Competitive Success.
Kanban	Kanban is an approach to managing work by means of pull. There are various approaches. They all create visibility and use signaling when to start work. A kanban board is used to show Work-in-Process (WIP).
KPI	Key Performance Indicator is an approach for measuring performance. Organizations use KPIs to evaluate progress and success in particular activities such as projects, programs, initiatives. See also *GQM* and *OKR*.

Lean	The approach that produces value for customers quickly through a focus on reducing delays and eliminating waste which results in increased quality and lower cost.
Lean Agile	An approach to software development that incorporates principles, practices, and methods from lean product development, Agile software development, design patterns, test-driven development, and Agile analysis.
Lean-Agile Team (Team)	The whole cross-functional group comprised of persons with skills who can perform three roles – Business analysts and SMEs (requirements and validation), developers, and test. The team selects the iteration goal and specifies work results, has the right to do everything within the boundaries of the project guidelines to reach the iteration goal, and organizes itself and its work.
Lean Governance	Lean governance is the leadership, organizational structures, and streamlined processes to enable teams within an organization to work as partners in sustaining and extending the organization's ability to produce meaningful value for its customers. Here are principles of lean governance. • Collaboration over conformance • Enablement over inspection • Continuous monitoring over quality gates • Transparency over management reporting Lean governance ensures that people and teams: • Are empowered to carry out their work. • Regularly and consistently create real business value. • Provide appropriate return on investment (ROI). • Deliver consumable solutions in a timely and relevant manner. • Work effectively with their stakeholders. • Work effectively with their colleagues. • Adopt appropriate processes and organizational structures. • Present accurate and timely information to stakeholders. • Mitigate risks. • Sustain and extend your strategies and objectives.
Manual Test	A document that lists the steps that a person follows to complete a test pass. Not automated.

Minimum Business Increment (MBI)	The minimum amount of business value that can be built, deployed and consumed that makes sense from a business perspective. It contains all the pieces required for realization of value. Among other things, this means the MBI: • Adds value for the Business and its customers • Provides valuable feedback that the right functionality is being built • Provides valuable feedback that the functionality is being built the right way • Provides functionality that can be verified as an increment that can be delivered • Enhances the ability of the organization to deliver value in the future
Minimum Viable Product (MVP)	The smallest piece of work to be used to validate a hypothesis about a potential product. The MVP is geared towards startups, toward the first time a product/service is released. It is usually built by a small team that can pivot.
Near-located Team	A team in which team members sit near one another, typically in the same area of a single floor, but do not have their own dedicated work room. See also *Colocated Team*, *Distributed Sub-team*, *Fully Dispersed Team*, *Partially Dispersed Team*, *Sub-team*.
OKR	Objectives and Key Results (OKR) is a framework for defining and tracking objectives (clearly defined goals) and their outcomes (specific measures to track that goal). Measures should be concrete, measurable, and specific. See also *GQM* and *KPI*.
Open-Closed Principle	The Open-Closed Principle suggests that it is better to create designs that can accommodate change by adding to them, rather than by changing them. "Open-Closed" means we are "open to extension but closed to modification." This is a principle, and it is impossible to follow it literally at all times, but it can guide us in refactoring as well.
Operating Model	An organization is a complex system for delivering value. An operating model breaks this system into components, showing how it works. It can help different participants understand the whole. It can help those making changes check that they have thought through all elements and that the whole will still work. It can help those transforming an operation coordinate all the different changes that need to happen.
Operational Metrics	Operational metrics are direct drivers of change and governance. Operational metrics show status across the entire program or project. There are two kinds of operational metrics: • **Program- and project-level operational metrics.** These are owned by the Product Owner and are calculated weekly. Examples include Release Burn-up, Total Top-Line, satisfaction, and process improvement.

	• **Daily operational metrics.** These are owned by the scrum master and are calculated daily. They are based on velocity. Examples include unestimated stories, open stories, open defects, tests passed, open issues and impediments.
Partially Dispersed Team	A team in which Some people work together at a single location, whereas others work at different locations. Often used to support a "work from home" strategy. See also *Colocated Team*, *Distributed Sub-team*, *Fully Dispersed Team*, *Near-located Team*, *Sub-team*.
Pattern	A collection of best practices for solving problems in a recurring context, represented as collections of forces and provide a professional language for high-fidelity communication among developers. The subject of many books including Design Patterns Explained (Shalloway and Trott 2004).
PDSA	Plan-Do-Study-Act (PDSA) cycle is an iterative four-step problem-solving process for quality improvement. Also known as the Deming Cycle, Shewhart Cycle and Deming Wheel. • Plan: Development of plan to effect improvement. • Do: Plan is carried out. • Study: Effects of plan are observed. • Act (or Adjust): Results studied, and learning identified for future usage.
Performance Test	A test that ensures the required level of performance of a product is met. This test checks both that the functionality works and that the time required to do the work is acceptable.
Persona	A description of the typical skills, abilities, needs, tasks, behaviors, and backgrounds of a particular set of users. As an aggregation, the persona is a fiction but is used to ensure groups of users are accounted for.
Phase	In Disciplined Agile, phases provide focus and reduce risk. Phases enable teams to focus on critical themes and senior management to govern consistently. There are three phases common across the project life cycles: • **Inception.** Initiate the endeavor in a streamlined manner, including initial planning and modeling, and obtain agreement regarding the vision with the primary stakeholders. • **Construction.** Develop a consumable solution in a collaborative and incremental manner. The team requests feedback on a regular manner which they then act on accordingly. • **Transition.** Release/deploy the solution to stakeholders, ideally at the point where a new minimum business increment (MBI) has been developed.
Planning	The activity that seeks to prioritize and define the stories and tasks for the next iteration. Also known as "loading the product backlog."

Portfolio Level	The Portfolio Level includes a diverse group of stakeholders working across organizational boundaries. The Portfolio Level is responsible for identifying, priority, sizing, and sequencing work based on Business value. Roles involved in the portfolio include the Value Stream Owners, the Business and Technology Sponsors, and stakeholders. Business strategy flows into the technology organization at the Portfolio Level.
Practice	A practice is something we, as a community, have discovered is simply always a good thing to do. For something to be promoted as a practice, it must be truly universal (you can always do it, and everyone should do it).
Process	A series of actions, changes, or functions performed in the making or treatment of a product.
Process Blade	A process blade addresses a specific organizational capability, such as finance, people management, data management, agile solution delivery, procurement, and more. A process blade encompasses a cohesive collection of process options – including practices, strategies, and workflows – that should be chosen and then applied in a context sensitive manner.
Process Goals	A process goal captures the detailed, process-related decisions and options for a cohesive subset of a team's way of working (WoW). Process goals provide guidance so that a team can tailor and scale agile strategies given the context of the situation that they face. Sometimes called a process capability.
Product	A collection of tangible and intangible features that are integrated and packaged into releases that offer value to a customer or to a market.
Product Owner	The product owner is the one individual on the team who speaks as the "one voice of the customer." He or she represents the needs and desires of the stakeholder community to the agile delivery team. As such, he or she clarifies any details regarding the solution and is also responsible for maintaining a prioritized list of work items that the team will implement to deliver the solution. While the product owner may not be able to answer all questions, it is their responsibility to track down the answer in a timely manner so that the team can stay focused on their tasks.
Product Vision	A short statement of the vision that is driving the project, expressed in business and customer terms while taking technological enablers into consideration: Who it is for, the opportunity, its name, the key benefit and differentiators. Typically, the Product Manager provides the product vision. Sometimes called the "Project Charter."
Program Increment	A larger development time-box that uses cadence and synchronization to facilitate planning, coordinate higher-level retrospectives, and aggregate work for feedback. The Program Increment is usually composed of multiple team iterations, often 4-6 team iterations.

Program Level	The Program Level is the center of responsibility that is focused on an application area. Work is comprised of releases, enhancements, production support, and maintenance requests. The Program Level includes cross-functional representatives including the Product Manager, Business PM, and Technology Delivery Manager who prioritize work across the value stream. This level also is responsible for program ecosystem and operational metrics.
Project	A collection of releases, iterations, team members, and stories that creates a product. May have defined end dates or be on-going. These may be: • All of the software a team or related teams are working on • A specific product or product group • A version of a company's product suite • A specific client implementation
Quality Assurance	A role and activity that ensures integrity of the code to be released. The job of QA is to prevent defects from happening in the first place… it is not 'merely' to find bugs.
Refactor	The disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior. Refactoring encourages and supports incremental design and implementation, the conversion of proofs of concept and early code into more suitable, stable code Two types of refactoring are fixing bad code and injecting better design into good code as requirements change.
Refactoring to the Open-Closed	The same tools and techniques that are used to clean up poor design and code (aka Legacy Refactoring) can be used to change a design just enough to allow for a clean addition or change to it. We take a design that was adequate initially, but which cannot now be changed cleanly to accommodate a new or changed requirement. Refactor until the code follows the open-closed principle, and then we integrate the new code.
Release	A version of a product that is released externally to customers. Releases represent the rhythm of the business and should align with defined business cycles. A release contains a combination of Minimum Business Increment that form a releasable product. A release may be internal and may be used for further testing.
Release Testing	The activity that validates that the product is good enough to release to customers. Release testing is typically performed by a tester and is sometimes called "Quality Assurance" (QA). Release Testing explores the statement, "We have built something that users can use." Often, release testing exposes requirements that fail to satisfy actual users' needs. See also: *Quality Assurance*, *Test*.
Retrospection	Retrospection is the structured reflective practice to learn and improve based on what has already been done. The purpose of retrospection is to

	build team commitment and to transfer knowledge to the next iteration and to other teams.
Role	The role assumed by an individual on a given project. People may serve different roles on different projects or even different iterations. See specific role definitions.
Root Cause	A factor that caused nonconformance to plan and should be permanently eliminated through process improvement. QA helps lead Root Cause Analysis.
Scrum	An iterative and incremental Agile software development methodology or framework for managing product development.
scrum master	The Scrum Master champions the needs of the team to the organization and for championing the needs of the organization to the team. The Scrum Master is the team's facilitator, shepherding the team as a servant-leader by creating a positive and safe environment, improving team cohesion, being creative and focused, asking difficult questions about process aimed at challenging the team to improve, bringing issues and impediments to the team, Product Owner, or management, and helping to develop the product backlog. This goes beyond the common understanding in Scrum. For this reason, Net Objectives refers to this role as a "Team Agility Coach."
Self-organizing team	In a self-organizing team, the people who do the work are the ones who plan and estimate the work. Team members volunteer to do certain work, collaboratively planning together, rather being assigned the work by a manager.
Standard Work	Standard work is the work process agreed to by the team doing it. By 'standard' we do not mean imposed, nor fixed. But rather the best way to do something. Standard work acts as a backdrop for doing our job and immediately seeing how well we are doing. Its intention is to create tension between what we're doing and what we're supposed to be doing. This promotes learning.
Stakeholder	A stakeholder is someone who is materially impacted by the outcome of the solution. In this regard, the stakeholder is clearly more than an end-user: A stakeholder could be a direct user, indirect user, manager of users, senior manager, operations staff member, the "gold owner" who funds the team, support (help desk) staff member, auditors, your program/portfolio manager, developers working on other solutions that integrate or interact with the one under development, and maintenance professionals potentially affected by the development and/or deployment of a software-bases solution. There are four stakeholder categories: • **End users.** These are the people who will use your solution, often to fulfill the goals of your principals. They typically want solutions that are usable and enable them to do their jobs more effectively.

	• **Principals.** These are the decision makers who ultimately pay for and then put your solution to use. This includes gold owners, senior business management, and purchasers of the commercial solutions. • **Partners.** These people make the solution work in production. This includes operations staff, support staff, trainers, legal experts, installers, application hosting companies, and application developers on external solutions which integrate with yours. • **Insiders.** These are members of the development team and people who provide technical and business services to your team. This includes enterprise architects, database administrators, security experts, network experts, tool smiths, marketing experts, and sales staff.
Story	An invitation for a conversation about requirements, features, and/or units of business value that can be estimated and tested. Stories describe work that must be done to create and deliver a feature for a product. Stories are the basic unit of communication, planning, and negotiation between the Development Team, Business Owners, and the Product Manager.
Story Point	The sizing metric used in the Team Estimation game. They express relative complexity for purposes of estimating how much to place into an iteration. A common approach is to use Fibonacci numbers (1,2,3,5,8, …) where 1 is low complexity, 8 is more complex) and then use Team Estimation or Planning Poker to get the team to agree on estimates.
Story Point Burn-up Chart	A chart showing the rate of progress the team is achieving in completing the project. It is measured as the number of story points completed per iteration and tracked toward the full project scope. Along with velocity, it visibly shows the duration and length of the current project.
Sub-team	Sub-teams are organized around job functions, such as testing or programming. Significant collaboration and communication required between sub-teams.
Subject Matter Expert (SME)	A person who can speak with authority on an aspect of a project or who knows to whom to talk in order to get answers. Subject Matter Experts may represent a technology, the business, the customer, process, or any other topic of importance to the project.
Swarm	The activity of a teamlet that forms to complete a work item. The rules for swarming are: • Focus on one story at a time. Within an iteration, teams should have only a few stories open at a time because the focus is on burning down stories. Do not dissipate energy by focusing on too much at once. • The swarm is the priority. While individuals may work on other tasks, their priority should be the swarmed story. • Swarms are skill-based. If you have the skills to contribute to burning down a story, and you have capacity, you are expected to join in the teamlet, even if it is not exactly in your job description to do so.

System Evolution	System evolution is the process of building functionality from the system perspective. That is, building one layer (e.g., database) and then another on top of it (e.g., business layer), etc.
Systems Thinking	Systems thinking is considering a system to be an interrelated and interdependent set of parts which is defined by its boundaries and is more than the sum of its parts (subsystems). Changing one part of the system affects other parts and the whole system, with predictable patterns of behavior. Positive growth and adaptation of a system depend upon how well the system is adjusted with its environment, and systems often exist to accomplish a common purpose (a work function) that also aids in the maintenance of the system or the operations may result in system failure. The goal of systems thinking is systematically discovering a system's dynamics, constraints, conditions and elucidating principles (purpose, measure, methods, tools, etc.) that can be discerned and applied to systems at every level of nesting, and in every field for achieving an optimized end state through various means.
Tactical scaling factors	DA identifies six factors involved in choosing and tailoring your people, process, and tooling as you move to organizational agility. • Team size • Geographic distribution • Organizational distribution • Compliance • Technical complexity • Domain complexity Understanding the context that teams and organizations face involves identifying where they are for each of the factors, which can range from relatively simple to relatively complex. It helps to plot these tactical scaling factors on a six-axis spider chart, each axis moving from simple to complex. Five process goals (Explore Scope, Identify Architecture Strategy, Develop Test Strategy, Accelerate Value Delivery, and Coordinate Activities) require about 80% of the tailoring effort involved in addressing the tactical scaling factors.
Task	Tasks are descriptions of the actual work that an individual (or sub-team) does in order to complete a story. Typically, there are several tasks per story. Tasks have the following attributes: • A description of the work to be performed, in either technical or business terms • An estimate of how much time the work will take (hours, days) • Exit criteria and verification method (test or inspection), and an indication of who will be responsible for the verification. All tasks must be verified, not just "done"

Team Lead	An important aspect of self-organizing teams is that the team lead facilitates or guides the team in performing technical management activities instead of taking on these responsibilities him or herself. The team lead is a servant-leader to the team, creating and maintaining the conditions that allow the team to be successful.
Team Level	The Team Level consists of individual teams responsible for producing and implementing a Business value increment, for quality assurance, and for continuous incremental improvement. Work is composed of the stories and tasks for a specific release, enhancements, production support, and maintenance requests. The Team Level also includes shared services.
Team Member	The role of team member focuses on producing the actual solution for stakeholders. Team members will perform testing, analysis, architecture, design, programming, planning, estimation, and many more activities as appropriate. Note that not every team member will have every single one of these skills, at least not yet, but they will have a subset of them, and they will strive to gain more skills over time. Team members will identify tasks, estimate tasks, "sign-up" for tasks, perform the tasks, and track their status towards completion. In addition to the general responsibilities described earlier, team members have several additional responsibilities.
Team Working Agreement	A team working agreement defines their internal Way of Working (WoW) and how they are willing to interact with other teams. External working agreements are sometimes defined in terms of Service Level Agreements (SLAs).
Test	Automatic and manual inspections of code and process to ensure correctness and completeness. Types of tests include Unit, Integration, System, Regression, Performance, and User Acceptance (Customer Acceptance)
Test-Driven Development	An evolutionary approach to development. In TDD, each test is written before the functional code that makes the test pass. The goal of TDD is specification and not validation, to think through a design before code is written, to create clean code that always works.
Time-box	An allocated period of time allowed for getting work done. This means that we commit to completing at a certain time, typically with a certain amount of effort (people and resources). We commit to getting as much work done as possible within this time frame.
Unit Test	The activity that verifies that software code matches the design specifications. Typically, unit testing is performed by a developer – by the code creator or by a "buddy" – however, testers often advise developers in testing approaches. Each unit test confirms that the code accurately reflects one intention of the system.

Use Case	In Lean-Agile, use cases express the details for a requirement story. If the use case becomes so large that it cannot be implemented in a single iteration, then the requirement story associated with the use case can be broken down into iteration requirement stories first. Alternatively, if a use case is more abstract, it can represent several requirement stories. Use cases should be expressed in the style of Cockburn detailed use cases. They may be white box or black box and include a main scenario, exceptions, and alternatives. Use cases can refer to business rules and data definitions.
User Acceptance Test (UAT)	The activity that verifies that software code matches the business intent. UAT belongs to the customer. They decide what constitutes an acceptable product. Unit tests help ensure that the acceptance tests are not about functional failures, but about the actual acceptability of the approach. Thus, UATs should not result in "it crashed" but «I would like the menus to be more descriptive."
Validation	The process of determining whether the work meets the needs of stakeholders.
Value-Added	A term used to describe activities that transform input into a customer (internal or external) usable output. An activity on a project is value-added if it transforms the deliverables of the project in such a way that the customer recognizes the transformation and is willing to pay for it.
Value Realization	Value is realized when customers begin to put to use what has been developed and get benefit from using it. This requires ops, marketing, support, and training.
Value Stream	A value stream is the set of actions that take place to add value to a customer from the initial request through realization of value by the customer. The value stream begins with the initial concept, moves through various stages for one or more development teams (where Agile methods begin), and on through final delivery and support. A value stream **begins** and **ends** with a **customer.**
Value Stream Owner	A person responsible for systematic management approach with immediate impact on the critical elements of a company's value streams. Makes change happen across departmental and functional boundaries.
Velocity	Velocity measures how many points the team spends during an iteration. The following velocities are used in the planning game to determine how many stories the customer should give to the team to estimate for the iteration: • **Story velocity.** How many story points the team completes in an iteration. • **Task velocity.** How many "engineering hours" the team actually can perform in an iteration.

Verification	The process of determining whether the work meets the specification for it.
Visual Control	A visual control is a lean tool that three primary purposes: • It shows when there is some abnormality in the process including blockages, people being overcapacity, dates at risk, and more • It provides an explicit workflow that the team is using • It provides management a view as to how work is flowing • A visual control can take many forms. Simple Kanban boards and complex product management systems are examples of visual controls.
Voice of Customer	The "voice of the customer" is the term used to describe the stated and unstated needs or requirements of the customer. The voice of the customer can be captured in a variety of ways: Direct discussion or interviews, surveys, focus groups, customer specifications, observation, warranty data, field reports, complaint logs, etc.
Waste	Any activity that consumes resources, produces no added value to the product or service a customer receives, or delays development of value. Types of waste include 1) anything that does not add customer value, 2) anything that has been started but is not in production, 3) anything that delays development, 4) extra features, 5) making the wrong thing.
XP (Extreme Programming)	A software development methodology adhering to a very iterative and incremental approach. XP consists of a number of integrated practices for developers and management; the original twelve practices of XP include Small Releases, Acceptance Tests, On-site Customer, Sustainable Pace, Simple Design, Continuous Integration, Unit Tests, Coding Conventions, Refactoring Mercilessly, Test-Driven Development, System Metaphor, Collective Code Ownership, and Pair Programming.
Yesterday's Weather	The expectation that a team will complete as many story points worth of work in the next iteration as they did in the last. Of course, this will only be true after they have done a few iterations and have hit a somewhat steady level. Typically, this is after 3-4 iterations. The term comes from the fact that before weather satellites, the most accurate way to predict weather was to say it would be the same as the day before. Hence, "yesterday's weather" means we expect what happened before