32 Principles and Practices of Highly-Successful Scaled Agile Framework (SAFe) Implementations

Sca·led (*skā'əld*) is the practice of expanding, enlarging, and increasing the size of an enterprise, portfolio, program, or project management framework, systems or software engineering framework, or other operations and maintenance framework. Managers and engineers quickly realized—especially with the advent, emergence, and stabilization of the electronic computer in the 1950s—computer systems were becoming infinitely complex—Beyond the ability of a human being or small group of people to comprehend. Therefore, elaborate management and technical management systems and frameworks were created, expanded, evolved, and refined throughout the 1950s, 1960s, 1970s, and 1980s to manage the complexity of creating and operating computer systems. These included elaborate military waterfall lifecycles from the 1970s and military maturity models from the 1980s. These traditional frameworks consisted of a myriad of program management plans, integrated master schedules, earned value management, lifecycle stages, milestone reviews, enterprise architectures, system architectures, testing regiments, and, more importantly, dozens and sometimes hundreds of lifecycle documents—Thousands of pages each! A single computer system project literally resulted in its own library of documentation at a cost of millions and billions of dollars. The basic theory was that the lifecycle would take 10 to 30 years to complete, tacit human knowledge needed to be codified quickly, and per chance, some engineer or user may need to understand 30 to 50-year-old management and engineering assumptions. Although, these lifecycles were created by mathematicians, scientists, and physicists, the infinitely complex lifecycles undermined system development success rather than increasing it, and 70% to 90% of projects actually failed for applying the cure—Waterfall lifecycles.

Ag·ile • Frame·work (*ăj'əl • frām'wûrk'*) is an approach to computer system development where small, self-organizing cross functional teams collaborate face-to-face with minimal documentation to speak with customers, codify and test their needs, deliver solutions quickly, and rinse and repeat over and over again until customers are utterly delighted. Agile frameworks are based on the fundamental theory that customer needs exist as hidden, inexpressible, tacit knowledge—They ain't gonna know what they want until they see it—And frequent small examples have to be placed directly in their hands until they find a version of the computer system they like—Even if they didn't know it in advance. Agile frameworks enable computer system development teams to release small versions of products and services directly to customers in minutes, hours, and days—And, written documentation may exist in the form of information radiators, whiteboarded requirements and designs, and the code and code comments themselves. Furthermore, complex computer systems could rapidly emerge, evolve, and aggregate as a collection of small frequent software releases or modules over time—And simply refactored (redesigned), discarded, and replaced when they are no longer needed—About every 18 months in accordance with Moore's Law. The basic notion was to talk to customers or end-users directly, document their needs as informal user stories or conversations, quickly codify assumptions, and test these assumptions quickly—Knowing that English is a natural language, ambiguous, and user stories were only incorrect approximations of end-user needs. Thus, agile frameworks enabled enterprises to rapidly adapt to dynamic market cycles very quickly, especially in the globally expanding marketplace with increasing customer bases, diversities of specialized users, and increasing competition.

Sca·led • Ag·ile • Frame·work • SAFe (*skā'əld • ăj'əl • frām'wûrk' • sāf*) is a model for expanding, enlarging, and increasing the application of lean and agile thinking values, principles, practices, tools, and technologies to the enterprise, portfolio, and program levels. Simple agile frameworks such as Extreme Programming (XP), Scrum, Feature Driven Development (FDD), Dynamic Systems Development (DSDM), and Crystal Methods, among a myriad of other competing models, were optimized for small teams to evolve computer systems one module at a time, based on small groups of user stories or backlogs. SAFe enables larger teams called value streams, solution trains, and agile release trains to apply lean and agile thinking to build complex systems composed of larger groups of software modules. SAFe began as an agile program and project management framework, to apply agile values, principles, and practices to multiple simultaneous agile teams, expanded into larger solutions, and groups of solutions called portfolios, and then finally into an enterprise business agility model. SAFe outgrew its basic agile roots and began to adopt lean thinking values, principles, and practices more and more over the years, to the point of becoming a lean framework in of itself—vs. an agile framework. SAFe's basic goal was to provide just-enough roles, ceremonies, and instruments to integrate the work of a myriad of lean and agile teams without becoming a traditional waterfall lifecycle, extending system lifecycles to billions of dollars and decades, or burdening managers and developers with libraries of documentation. SAFe was originally conceived for a team of 125 developers to build a moderately complex computer system in about 18 months, but then expanded into a coordination model for multiple groups of 125 person teams, while recognizing the DevOps ideals of multiple daily releases.

After about 15 years, SAFe capitalized upon the need for agile project management frameworks—To coordinate multiple agile teams—And has become the preferred lean and agile enterprise, portfolio, large solution, and program framework for large commercial firms, public sector agencies—including the military and intelligence communities, and small to medium sized enterprises undertaking complex transformational programs. However, old habits die hard, manufacturing era process and documentation heavy (military) waterfall lifecycles abound—So do traditional portfolio, program, and project management and systems engineering models, and enterprises are tempted to add the new SAFe model to their existing pantheon or temple of traditional models. Basically, why not add a new model to our management and technical pantheon, since our existing pantheon doesn't work very well, and SAFe seems to be popular? We can simply satisfy the old timers and millennials at the same time! More importantly, we can create hybrid traditional-SAFe models, pick and choose which ceremonies to adopt and add them to our dysfunctional pantheon, and allow our army of executives and middle managers to use SAFe to torment a (subcontracted) programmer or two—Sounds like fun, what can go wrong! Well, a lot can go wrong, because SAFe is an all encompassing set of lean and agile values, principles, and practices for enterprises, portfolios, large solutions, and programs, it's meant to streamline the end-to-end value stream, and its applies to everyone—Not just a few poor computer programmers from a developing nation! Continue with debilitating waterfalls, piecemeal SAFe, do SAFe poorly, or have a few bottoms up SAFe teams in a sea of traditional sharks, and there is gonna be hell to pay! Have enterprise executives pull SAFe in, follow the SAFe roadmap, apply lean and agile thinking, continuously retrospect, improve, rinse, and repeat over and over again—And SAFe will function correctly!

SAFe Principles & Practices

1. Es·tab·lish • Top·down • SAFe • Lea·der·ship (*ĭ-stăb'lĭsh • tŏp-doun • sāf • lē'dər-shĭp'*) Chief, authority, program manager, director, executive; To select a powerful leader who will champion SAFe from very top

   ✓ Progressive leader
   ✓ Trusts SAFe outcomes
   ✓ Directs SAFe implementation

   Establishing topdown SAFe leadership single most important critical success factor (CSF) to SAFe adoption at any level, be it at the enterprise, portfolio, large solution, or program-level. This is certainly true of any organizational change initiative, be it traditional, lean, or agile, and has been well-known at least since the 1950s in the Western hemisphere. Whether it is adopting computer systems, automating business processes, or promulgating a new management paradigm, the support of a key executive is absolutely necessary. Even if one is merely adopting SAFe for a single transformational program or initiative, at least the program manager has to be ALL-IN on SAFe adoption, implementation, and use. The quantitative or qualitative strategic benefits or business value of process-oriented paradigms like traditional, lean, or agile program or project management were not very well understood nor accepted for the last 70 or so years. This often led to bottoms up adoption of process paradigms one person at a time until the point when people at the bottom of the hierarchy rose or were elevated to the top of the hierarchy. Well, 70 years later, process believers (asylum inmates) are running today's enterprises, portfolios, and programs, process paradigms have reached the tipping point, and many enterprises are flocking to process paradigms in droves without much business justification at all. The promises and payoffs of lean and agile paradigms are also smoking hot in the public and private sectors and SAFe is seeping into modern enterprises from the top, middle, and bottom. However, organizational psychology and change management theory, laws, and principles haven't changed too much since the stone ages of the 1950s when project management and systems engineering were born. That is, SAFe must be pulled into the enterprise by a powerful, influential, and visible authority such as an executive, director, or program manager and pushed down upon the enterprise, portfolio, or program. This doesn't mean there won't be resistance to change, and the old adage, "the bigger they are, the harder they fall," still applies. Enterprise change management paradigms have suffered 70% failure rates for the last 70 years or so, and SAFe adoptions may also be subject to this historical change management CONSTANT. However, rest assured, the SAFe failure rate will be 100% if driven into a large enterprise, portfolio, or program from the bottoms up—Perhaps a single SAFe champion against an army of hard-headed traditional managers and engineers. Conversely, the SAFe adoption rate is significantly higher, but not guaranteed, if driven from the top down using well-funded change management teams and unrelenting support from the leader who was bold enough to charter a SAFe adoption.

2. Form • SAFe • Lea·der·ship • Team (*fôrm • sāf • lē'dər-shĭp' • tēm*) Planning council, steering committee, governance team, executive group, board of directors; To form a powerful coalition for planning, monitoring, and improving SAFe implementation

   ✓ Form SAFe steering committee
   ✓ Charter SAFe implementation plan
   ✓ Monitor and optimize SAFe adoption

   The next most important critical success factor (CSF) is forming a SAFe leadership team. That is, SAFe transformation, adoption, and use is a team sport, two heads are better than one, and depending upon the size of the SAFe transformation, be it at the enterprise, portfolio, large solution, or program-level, this may affect the size of the leadership team. For the sake of expediency, we're combining at least two major concepts here, one—there has to be a leadership oversight committee to authorize the planning, execution, and measurement of the SAFe adoption, and two—there has to be a SAFe transformation team, largely comprised of SAFe Program Consultants (SPCs) to plan, execute, measure, and improve the SAFe technical ceremonies, practices, and tools. However, in the end, there has to be an independent SAFe leadership team to relieve the program manager of having to make day-to-day SAFe transformation decisions. This isn't to say executive leaders shouldn't participate, be involved, monitor, and be informed about the status of the SAFe transformation, but the executive has other responsibilities, such as managing stakeholders, bounding the context of the SAFe transformation, budgeting and staffing, ensuring the transformation is succeeding, and protecting the transformation from cannibalization by other starving executives who are hungry for the same resources (sharks). A large gathering of well-funded, organized, stable, and talented resources is like a buffet for competing executives. Oftentimes, this steering committee is called a "Lean-Agile Center of Excellence" (LACE) in the SAFe vernacular, a portfolio or program management office (PMO), a Lean or Agile Management Office (AMO), or simply a SAFe governance board or steering committee. Apart from the fact that it is a small team of SAFe trained and certified experts and, more importantly, SAFe champions, preferably SPCs, its goal is to monitor and ensure the successful SAFe adoption without having to burden the primary executive, director, or program manager directly, although they report to the primary leader on a regular basis (i.e., daily, weekly, bi-weekly, monthly, or quarterly). Obviously, the goal of the SAFe adoption is to drive the delivery of high-value adding products and services to market quickly, so the SAFe leadership team must first ensure the SAFe adoption is obtaining quick wins, and then communicating these wins to the sponsoring leader early and often. Depending upon the size of the SAFe adoption, a SAFe implementation team may also be chartered to run the SAFe transformation as an Agile Release Train (ART), which is a critical success factor all by itself. SAFe has evolved into a complex enterprise agility model, necessitating SAFe transformation teams at multiple levels. This is a complex CSF.

3. Ap·ply • Lean • and • Ag·ile • Thin·king (*ə-plī' • lēn • ănd • ăj'əl • thĭng'kĭng*) Doctrine, reasoning, judgement, philosophy, decision-making; To deliver valuable, high-quality products and services with fast lead and cycle times

   ✓ Demand-based, just-in-time
   ✓ Fast, waste-free delivery speed

✓ Rapid value-creating experimentation

An often-forgotten critical success factor (CSF) to SAFe adoption is to apply lean and agile thinking. Although SAFe began as an Agile program management model, SAFe has evolved into a lean thinking model for the entire enterprise. That means, one should understand and master how lean and agile thinking is implemented in SAFe. Lean thinking is a pull or market demand-based end-to-end workflow system to rapidly deliver high-quality, highly-value products and services in a cost-effective, waste-free manner. Although SAFe is an out-of-the-box model to apply lean thinking in an enterprise, portfolio, large solution, or program, it still takes a concerted effort to yield the benefits of lean, agile, and SAFe thinking. At the enterprise-level, SAFe is comprised of seven core competencies to drive lean and agile thinking through the enterprise fabric. Many times, enterprises just want to bootstrap lean, agile, or SAFe onto a dysfunctional enterprise to speed it up, do more with less, or simply modernize its business and technical management practices. It's sort of like an out of shape person jogging uphill for 25 miles with a 75 backpack! Well, it takes 2 to 8 hours for a well-conditioned jogger to run a 25-mile marathon. It will take a million years for an out-of-shape person to attempt a 25-mile marathon, if he or she survives without having a heart attack and dying, even with SAFe strapped to his or her back. Therefore, before one begins a SAFe marathon, its entire team should take stock of lean thinking, what it means, and how to apply it through SAFe. At its most basic level, lean thinking is a paradigm for conducting small, inexpensive business experiments to identify tacit, often-hidden, highly uncertain, and immensely valuable market, customer, or end-user needs, and then targeting products or services at a proven market. It's like firing tracers in the dark until you find a target and then concentrating all of your firepower on that target. Some business analysis is okay to better aim the nighttime tracers, but one doesn't want to start a 25-year billion-dollar waterfall project designing a gun to fire in the wrong direction. Oftentimes, customers and markets don't stay still for very long—Remember "Moore's Law," so large heavyweight multi-decade long business solutions have a short shelf-life, unless you're in the public sector where they last forever! So, if your goal is to successfully field an expensive obsolete legacy system to keep people employed forever, this isn't exactly what we mean by lean thinking. If you're goal is to field a small family of short-lived business experiments to optimize profits and revenues in the near-term while delighting customers, then you're in the right ballpark. None-the-less, learning and mastering lean thinking early and often significantly increases SAFe adoption success.

4. Re·sist • Hybrid • Tra·di·tion·al • Prac·tic·es (*rĭ-zĭst' • hī'brĭd • trə-dĭsh'ə-nəl • prăk'tĭs'əz*) Enterprise architectures, plans, processes, compliance, documents; To resist refining heavy non-value adding processes, documents, and overprocessing

✓ Resist big design up front (BDUF)
✓ Resist integrated master schedules
✓ Resist long heavyweight linear lifecycles

Resisting hybrid traditional practices is one of the most perplexing critical success factors (CSFs) in SAFe. Traditional project management, systems engineering, and other manufacturing process models such as CMMI and ISO 9001 are still pervasive in the early 21st century. Rather than abandoning Frederick Taylor's Scientific Management ideology from 1900, early 21st century firms simply bootstrap SAFe adoptions on top of debilitating multimillion-dollar CMMI and ISO 9001 initiatives sapping the lifeblood out of modern enterprises. During the U.S. Civil War, battlefield surgeons raced to saw off the war-torn limbs from soldiers while they bled to death in field hospitals, before blood types were discovered and blood transfusions were stabilized 25 years later. We live in such an age where organizations are profusely bleeding money with expensive traditional initiatives, while lean and agile surgeons race to save information technology patients with SAFe transformations. Early lean and agile leaders learned to speak the language of traditional executives and middle managers and assured enterprises that SAFe could be easily bootstrapped onto traditional harnesses without upsetting the apple cart. Big Six consultants learned to master this double speak or came to believe that traditional and SAFe paradigms could peacefully co-exist. Unfortunately, SAFe programs were burdened with not only instituting lighter weight lean portfolios, solution trains, and agile release trains, but producing a mountain of paperwork, participating in numerous traditional processes, and passing through a gauntlet of linear waterfall milestone and stage gate reviews too. What this means is that early SAFe programs were burdened with DOUBLE the workload and proponents of traditional enterprise initiatives such as CMMI and ISO 9001 management offices were more than happy to watch fledgling SAFe initiatives die on the vine. SAFe victories were overlooked in silence while 21st century organizations continued to hold expensive parties to celebrate the successes of multi-million dollar CMMI and ISO 9001 initiatives that defied lean and agile physics, blocked organizational queues forever, and undermined the success of both buyers, suppliers, and SAFe implementations. Slowly, SAFe initiatives sought temporary exemptions from CMMI and ISO 9001 compliance and now a few early adopters are finally throwing off these unnecessary training wheels altogether. It is simply not necessary to have a 30-year enterprise architecture, stack of business requirements documents, integrated master schedule, stack of military-era documents from the 1970s, and other widely spaced milestone reviews or stage gates. Even traditional information or cybersecurity controls are being absorbed by modern DevOps pipelines.

5. In·still • SAFe • Thin·king • Ear·ly • and • Of·ten (*ĭn-stĭl' • sāf • thĭng'kĭng • ûr'lē • ănd • ôf'tən*) Imbue, infuse, implant, immerse, saturate; To continuously learn, practice, and master SAFe values, principles, and ideals

✓ Learn basic SAFe values
✓ Internalize SAFe principles
✓ Continuously improve and master

Instilling SAFe thinking is an important critical success factor (CSF). Whereas lean thinking examines SAFe's underlying philosophical and often hidden dark matter or fabric, SAFe thinking focuses on SAFe's visible elements, matter, and practices. Given SAFe's evolution into a comprehensive business agility model, this is a rather tall order. Instilling SAFe thinking simply can't be skipped, assumed, taken for granted, or an ignored imperative—whether it involves learning, understanding, and mastering SAFe's core competencies, rigorously implementing SAFe's many ceremonies, or adopting SAFe's core values,

mindset, principles, or implementation roadmap. It's just hard to succeed with SAFe if one ignores lean portfolio management, large solution management, or essential practices within SAFe. It's often said that Program Increment (PI) planning is the secret sauce or "magic" within SAFe—Well, if that's so, then certainly learn, master, and optimize PI Planning at all levels—conceptual, logical, physical, etc.—And, certainly don't skip or attack SAFe PI Planning. Unfortunately, PI Planning is one of the largest and most visible ceremonies within the SAFe model and is the first to be attacked, skipped, or optimized away! Ironically, SAFe PI Planning is perhaps the single most important element to SAFe success or lean and agile success at scale! So, skipping PI Planning is to remove one's brain before birth to ease a baby's removal! SAFe PI planning is the "brain child" of SAFe, where all stakeholders are periodically brought together, collaborate on a short term plan, make critical course adjustments and adaptations, remove impediments to success, and select a better course of action than the last one. In other words, SAFe PI planning is a fundamental CSF to lean and agile success! However, people, oftentimes untrained traditionalists, attack PI Planning, don't understand its value, refuse to participate, or turn it into a global virtual Skype session in six different time zones with poor telecommunications in order to save money! When SAFe PI planning is held, it is often shortened, abbreviated, poorly executed, facilities are not conducive to collaboration, no collaboration takes place at all, all planning is done before hand, or all critical stakeholder teams are not present nor subject to participation, synchronization, and transparency. It's just hard to be successful with SAFe, if people aren't trained, they're not onboard, they're antagonistic, SAFe is poorly executed, ceremonies are rubberstamped, or they're repurposed for traditional uses—Turning a 15-minute standup meeting into a 6-hour daily brainstorming session for 90 days, or skipping other critical ceremonies. There is simply no substitute for lean thinking, SAFe ceremonies, visualization, collaboration, participation, and other agile values.

6. Fol·low • SAFe • Road·map (*fŏl′ō • sāf • rōd-măp*) Plan, guide, design, scheme, strategy, blueprint; To observe and comply with the 12-stage SAFe implementation roadmap

   ✓ Establish business case
   ✓ Launch agile release trains
   ✓ Improve and expand portfolio

   Following the SAFe Implementation Roadmap to the tee may be the single most important tangible critical success factor (CSF). Unfortunately, much like SAFe PI Planning, the SAFe Implementation Roadmap is often skipped, ignored, suboptimized away, or anemically abbreviated. One of the most common SAFe implementation anti-roadmaps is to hire a single SPC for a large traditional program, have him or her single-handedly stand up an Agile Release Train (ART), perform all of the SAFe ceremonies, tooling, data entry, and metrics reporting alone, etc. And, if you can get the SPC to be the program manager too, for half the price of a traditional program manager, to turn seven jars of outdated technologies into cloud wine, then you're really cooking with gas. There are too many cut throat capitalists looking for low-priced SPCs willing to take on the job of personifying the SAFe Implementation Roadmap to quickly insert a risky outdated technology stack into a 20th century brick-and-mortar organization and plenty of virgin SPCs willing to be sacrificed to traditional volcano gods. That being said, the better high-road, is to rigorously follow the SAFe Implementation Roadmap to the tee—Reaching the Tipping Point, Train Lean-Agile Change Agents, Train Executives, Managers, and Leaders, Create a Lean-Agile Center of Excellence, Identify Value Streams and ARTs, Create the Implementation Plan, Prepare for ART Launch, Train Teams and Launch the ART, Coach ART Execution, Launch More ARTs and Value Streams, Extend to the Portfolio, and Accelerate. What part of the SAFe Implementation Roadmap involves hiring a low cost SPC to do the work of 15 people, work 80 hours a week, bootstrap SAFe onto a hybrid traditional waterfall, continue executing waterfall practices, implement a stack of 2 million 15 year old business requirements, and jam an expensive outdated technology stack of vaporware into the enterprise because some executive was duped by a IT snake oil salesman? This fails to mention that none of this involves any lean nor SAFe thinking at all, except that the SPC is lean and mean! Seriously though, the SAFe Implementation Roadmap demands topdown leadership commitment and resources, a TEAM of SAFe experts, training and certifying EVERYONE, and organizational commitment to TRANSFORMING from outdated traditional practices to lean and agile ceremonies. Did we mention the application of lean thinking, which involves a right-sized bottoms up emergent architecture and design solution vs. heavyweight top down immersion in expensive and unproven vaporware? This is where things go wrong from the start—Choosing a heavyweight solution and hiring low cost IT folks to implement it using hybrid traditional-SAFe practices!

7. Train • and • Cer·ti·fy • Eve·ry·one (*trān • ănd • sûr′tə-fī′ • ĕv′rē-wŭn′*) Teach, instruct, educate, qualify, indoctrinate; To subject stakeholders for formal classroom instruction and evaluation

   ✓ Identify stakeholders
   ✓ Apply classroom instruction
   ✓ Evaluate and qualify stakeholders

   Part and parcel to following the SAFe Implementation Roadmap is the critical success factor (CSF) of training and certifying everyone. Clearly, retaining the services of an external SAFe transformation team to coach executives, steering committees, and programs in how to properly implement SAFe is a tried and proven approach, especially if they're trained and certified. More importantly, the external team should be selected on their experience and track record successfully implementing SAFe rollouts. However, SAFe consultancies should coach and consult, not do the work for the target organization. They can assess the organizational climate for change, determine the best approach and pain points (programs) for targeting SAFe implementation, develop strategic and tactical implementation plans, train and certify key stakeholders, help stand up and execute Agile Release Trains (ARTs), and facilitate retrospectives and process improvement plans. In the end, the target organization should be vested in authorizing the SAFe rollout, providing resources, directing or shepherding self-selected volunteers to participate in SAFe rollouts, and directly staffing the SAFe ARTs themselves. As such, executives, directors, program managers, leadership teams, and personnel fulfilling key SAFe roles should be trained AND certified. It's okay for the

target firm to hire external people with the requisite training, certification, and experience into these roles. However, the organization should strive not only to train and certify strategic and tactical stakeholders, but everyone. Furthermore, training and certification should include all of the SAFe certifications and other continuing education courses in lean thinking, automation, collaboration, teamwork, communication skills, emotional intelligence, servant leadership, etc. SAFe transformation simply doesn't work if you hire an untrained IT worker, assign a SAFe role to that person, and expect that person to understand, support, and be proficient at SAFe. More importantly, without SAFe training and certification, that person may simply not believe in the value, benefits, and power of SAFe and execute the role poorly. Continuous coaching and further training and certification are imperative, because people go native, put their blinders on, jump into a foxhole, and suboptimize their individual or team (silo) performance because it is the easiest to control—Thus not optimizing the whole! Once the battle begins, people can no longer see the forest for the trees, systemic antipatterns and impediments must be identified and mitigated quickly, professional retrospectives externally and independently conducted, and improvements made quickly to re-optimize the whole. Self-selection has a lot to do with SAFe success—That is, allow people to opt-in to SAFe.

8. Form • SAFe • Prod·uct • Man·age·ment • Team (*fôrm • sāf • prŏd'əkt • măn'ĭj-mənt • tēm*) Solution or product research, vision, communication, strategy, roadmap; To charter a team or group for translating epics into selected capabilities and features

   - ✓ Form solution or product management team
   - ✓ Conduct customer research to identify needs
   - ✓ Create solution or product vision and roadmap

   Forming SAFe solution or product management teams is an often-overlooked critical success factor (CSF). Solution and product management teams are responsible for interacting with markets, customers, and end users to understand, measure, and define their needs. Lean portfolio management teams do a lot of the same things at the enterprise or portfolio level when defining strategic themes, portfolios, epics, and epic enablers. However, it is the job of solution and product managers to translate epics into capabilities and features that must be sharply aligned with market, customer, and end-user needs, often times through direct market, customer, and end-user interaction. In the end, capabilities and features are merely hypotheses that must be implemented, fielded, tested, measured, refined, and optimized to satisfy epic and epic enabler measures. For instance, an epic enabler may be that a scalable storage solution is needed for under $2 million. The solution and product management teams interact with market participants that may suggest a virtual commercial storage solution with 2.5 petabytes is sufficient. A solution or system architect may suggest an Amazon Web Services (AWS) storage solution with a particular component design for $200,000 is sufficient to satisfy the capability or feature statement. An Agile Release Train (ART) may quickly implement a market test in 2 to 10 weeks or less, solution or product architects evaluate its performance characteristics, and solution or product management conducts a market, customer, or end-user evaluation to determine its sufficiency. This is not a one-shot deal, and solution or product management may continue refining capability and feature specifications along with solution and system architectures, until an optimal solution is achieved. Product and solution management teams must look at all epics in the portfolio Kanban, select the next one in the pipeline, identify capabilities, and features, conduct tradeoff and market analyses, down select capabilities and features, and prepare them for solution and system architects, who must do likewise. Solution and product management teams must apply lean thinking principles, understand workflow capacity, and stage high-priority capabilities and features for architecture analysis and implementation without exceeding WIP limits or capacity limits. They should collaborate with solution and release train engineers, they should not micromanage solution and release trains, they are not lone wolves, and they should NOT push too many capabilities and features, and strive for full capacity or utilization. Less is more in lean thinking, excess capacity is better than full utilization, and solution and product managers are not traditional project managers who must justify every minute of everyone's day.

9. Form • SAFe • Ar·chi·tec·ture • Team (*fôrm • sāf • är'kĭ-tĕk'chər • tēm*) Solution or system conceptual model for structure, behavior, view; To charter a team or group to define solution or system architectures for selected capabilities and features

   - ✓ Form solution or system architecture team
   - ✓ Identify solution or system design alternatives
   - ✓ Create a solution or system architectural runway

   Forming SAFe solution or system architecture teams is a critical success factor (CSF). They are responsible for interacting with solution and product management teams to evaluate and suggest architectures for capabilities and features. Architecture teams must consider multiple architectural alternatives. One technique or method is using set-based design, where multiple architectural alternatives are considered, the intersection of architectural elements are identified, and a bare minimum architectural alternative is recommended. Much interaction with solution and product managers is suggested, along with interacting with solution and release train engineers, as well as Agile Release Train (ART) teams themselves. Developers, with much hands-on implementation experience and direct market, customer, or end-user knowledge and interaction, may have much to say about solution and system architectural alternatives and element down selection. For instance, a solution or system architecture team may suggest a particular Amazon Web Services (AWS) architectural pattern, replete with individual AWS elements. However, the ART design, development, implementation, or even test team may suggest a different simpler AWS component set, a different pattern altogether, a simpler design, a need for additional components, or even a different solution provider such as an Azure or IBM Cloud with superior features, functions, performance, and price. ART subject matter experts (SMEs) have their pulse on the state of information technology and may even suggest dumping an obsolete technology stack, selecting a superior emerging technology stack, or simply be tired of a clunky old client-server implementation from the 1980s in lieu of a state-of-the-art technology with a vastly superior user experience. This is the art and science of cross functional SAFe Agile teams—That is, leveraging the synergy of multiple disciplines to weigh in on solution and product management, solution and system architecture, design, development, and implementation constraints,

and emerging market trends with superior cost-performance thresholds and end-user experiences (in-real time). This is why solution and product managers are NOT lone wolf traditional project managers, collaboration with solution and release train engineering teams and ART implementation teams is necessary, bottoms up agile decision making is imperative, SAFe roles shouldn't be ignored or combined, SAFe training and certification should NOT be skipped, agile coaching is critical, and emotional intelligence and empathy are priceless. We live in a world of hyper-individualistic narcissists, virtually distributed global teams, and suboptimizing anti-social lone wolves striving to get ahead in spite of the team, the whole, or the market.

10. Keep • It • Sim·ple (*kēp • ˈət • sĭmˈpəl*) Easy, plain, basic, elementary, uncomplicated; To keep the solution or product scale, scope, and risk as simple, narrow, and easy as possible

- ✓ Identify a few high-level epics
- ✓ Select a few value-adding capabilities
- ✓ Build a small set of features and stories

An absolutely essential lean and agile critical success factor (CSF) in SAFe adoption is keeping it utterly simple. But, what are we to keep utterly simple? Everything! That is, every element from the enterprise itself, to the portfolio, value streams, epics, solution trains, contracts, capabilities, solution architectures, agile release trains, features, system architectures, designs, user stories, tests, tooling, metrics, team size, etc. What do you mean? Isn't keeping it utterly simple completely antithetical to the SCALED Agile Framework (SAFe)? Well, yes and no, as Albert Einstein once said, "Everything should be made as simple as possible, but no simpler." Remember, the goal of lean thinking is to build a demand-based value stream to deliver small high-quality value adding just-in-time product and service batches to markets, customers, and end-users. Lean systems enable enterprises to rapidly evaluate epic minimum viable product (MVP) hypothesis statements and rinse-and-repeat many times in rapid succession until business value is peaked, optimized, and achieved. It's very difficult to deliver a Naval battle group to the market in a day, week, iteration, month, SAFe Program Increment (PI), year, etc. Although SAFe was initially conceived as an Agile program management framework of a single large information technology system requiring the work of 125 developers, it was never conceived to manage infinite complexity associated with century long weapon systems. Today, SAFe has evolved into an enterprise-level business agility framework for applying lean thinking principles to conduct many small simultaneous business experiments within tightly managed work in process (WIP) constraints. That is, one cannot or should not conduct simultaneous business experiments that exceed the capacity of the portfolio, value streams, solution trains, or agile release trains. Therefore the number and size of epics, capabilities, features, and user stories should be kept to a very minimum, capacity utilization and WIP limits should be dramatically lowered, business experiment speed increased, market feedback rapidly measured and valued, and optimal solution convergence sought, achieved, and exploited. It doesn't take long to amalgamate a large number of validated business experiments into a complex ecosystem of DevOps enabled microservices based upon principles of lean thinking. In other words, one can still build a battleship, one small business experiment at a time. Consider Tesla, that has amalgamated 200 million lines of code and makes 150 wireless system releases a day to its entire fleet of automobiles in real-time. SAFe evolved from a framework to eat baby elephants in one bite to a lean framework for eating large elephants one bite at a time like lean and agile principles were originally intended.

11. Keep • It • SmalI (*kēp • ˈət • smôl*) Tiny, little, minute, petite, miniscule; To keep the solution or product size, magnitude, and dimensions as small, diminutive, and pintsized as possible

- ✓ Limit size of capabilities
- ✓ Limit size of features
- ✓ Limit size of stories

Another absolutely essential lean and agile critical success factor (CSF) in SAFe adoption is keeping it utterly small. Once again, this applies equally to enterprises, portfolios, value streams, backlogs, epics, solution trains, capabilities, agile release trains, features, agile teams, user stories, and software modules (microservices). In lean and agile thinking frameworks such as SAFe, not only does simple enable complexity, but, likewise, smaller is larger. Infinitely large enterprise architectures, business requirements, system architectures, project plans, Gantt charts, system requirements, and code sizes instantly result in infinitely large budgets, team sizes, schedule lengths, lead times, cycle times, wait times, impediments, dependencies, bottlenecks, and queues. However, keeping portfolio, solution, and agile release train Kanbans and backlogs small, keeping epics, capabilities, features, user stories, and software modules (microservices) small, and keeping portfolio, large solution, agile release trains, and agile teams small, dramatically shortens lead, cycle, and wait times, minimizes impediments, dependencies, bottlenecks, and queues, and enables teams to rapidly field a variety of competing business experiments. In doing so, small releases enable teams to rapidly evaluate epic, capability, feature, and user story hypothesis statements, clear up uncertainty, gather valuable market, customer, and end-user intelligence, re-specify solution and product needs, implement correct value adding solutions, inject these into the market, and immediately garner the associated business value sooner. Bigger is smaller in traditional thinking—the bigger they are the harder they fall, and smaller is bigger in lean and agile thinking—The faster valuable products and services can be fielded the bigger the payoff. It's much easier to optimize user experience with smaller solutions than it is with larger solutions, security is easier, operations and maintenance is easier, and smaller solutions can be amalgamated or aggregated into larger families or ecosystems of cooperating microservices. A valuable microservice or family of highly cohesive, but loosely coupled microservices can be rapidly constructed by small two-pizza teams or even pairs of programmers. It doesn't necessarily require a 125-person agile release train to construct a valuable microservice. SAFe 5.0 enables a smaller enterprise, portfolio, solution train, or agile release train to behave like a much larger traditional organization, rapidly field a large number of small business experiments, achieve revenue targets sooner and capture larger marketshares, and rapidly amalgamate these experiments into larger and more complex solutions. Less is more, smaller is bigger, slower is faster, and SAFe enables small teams to eat larger elephants one bite a time.

12. Keep • It • Vi·su·al (*kēp • 'ət • 'vĭzh'o͞o-ə-l*) Visible, graphical, observable, pictorial, illustrative; To keep the solution or product roadmaps, kanbans, program increment plans, and iterations plans openly visible

- ✓ Use solution or program boards
- ✓ Use visible information radiators
- ✓ Use Obeya-style planning rooms

Keeping it visual is a critical success factor (CSF) associated with SAFe. At its very foundation or DNA, SAFe evolved into a lean thinking framework for business agility. Lean thinking is based upon the Toyota Production System (TPS) from Japan. In Far Eastern cultures, especially Japan's, human communications are based upon intense tacit facial expressions, body language, and visualization, and far less upon explicit verbal or written communication. Visualization is a rich, high-context, multi-channel real-time means of communicating a complete picture from brain to brain without engaging in slower, analytical, left-brained oriented data processing. When one is happy, he or she smiles, conversely, when one is sad or angry, he or she frowns. It's as simple as that! Do you want to implement transparency and communicate rich high context planning information to EVERYONE? Then, construct a Kanban board on the wall for everyone to see, place the unstarted work in the backlog or queue, place the started work in the processing lane, and place the finished validated work in the finished lane? Do you want to hide your project performance from everyone or share it with a small number of high-powered executives while keeping your most productive workers in the cold, so they don't steal your job? Then construct mile-long Gantt chart, place it in a hard to use project management tool with a limited number of licenses, construct a mountain of business requirements and place them in a hard-to-use lifecycle management system, don't put anything on a wall, and hire a small team of database experts to construct complex queries and reports to confuscate project performance. Do you want to succeed with SAFe, form an Obeya (war room), put your portfolio, solution, and agile release train Kanbans on the wall for everyone to see, hold regular live face-to-face Program Increment (PI) Planning sessions with everyone present in front of the boards, use your hands to move work item types around, empower everyone to interact with the boards, and leave them there for everyone to see and update. Do you want to fail with SAFe, buy and mandate the use of a complex agile lifecycle management tool, fat finger all of your work item types into it in advance (individually)—because you hate teamwork, empowerment, and collaboration and you're a selfish individualist, complain about how wasteful two-day PI Planning is (and don't participate), sit in closed rooms reviewing preplanned Kanbans on laptops, and then verbalize your plans on tiny computer monitors that no one can see and without a microphone so they can't hear you either. Place as many obstacles as you can between the people and the small computer monitors as you can so they can't critique your Kanbans, or better yet, don't even show your Kanbans.

13. Min·i·mize • Com·plex·i·ty (*mĭn'ə-mīz' • kəm-plĕk'sĭ-tē*) Not elaborate, detailed, intricate, convoluted, complicated; To keep the solution or product architectures and designs as simple, minimalistic, modularized, and narrowly-scoped as possible

- ✓ Specify a few high-value epics
- ✓ Specify a few capabilities and features
- ✓ Specify a few stories, designs, and modules

Minimizing complexity is also critical success factor (CSF) in SAFe, which is firmly based on lean thinking. As such, vital lean behaviors are to reduce lead, cycle, and wait times, minimize waste, reduce utilization, build in excess capacity, REDUCE queue and batch sizes (complexity), minimize multi-tasking, and other degenerative traditional behaviors. In doing so, lead, cycle, and wait times are decimated, delivery speed is optimized, customers receive needed products and services faster, priceless market feedback and business intelligence can be quickly gathered and analyzed, and the experimental cycle can be cost effectively repeated over and over again. Conversely, increasing complexity slows lead and cycle times, maximizes waste and processing time, maximizes utilization, exceeds capacity limits, INCREASES queue and batch sizes, necessitates degenerative multi-tasking, etc. Complexity increases dependencies, impediments, and frozen queues, stops productivity and throughput, decreases quality, and eliminates the possibility of gathering market feedback to conduct rapid process improvement cycles, and decimates business value and market share as well. Degenerative traditional practices that lead to complexity include large enterprise and system architectures, project plans and schedules, multiple acquisition contracts, linear waterfall lifecycles, stage gates, and milestone reviews, numerous processes and documentation, and manual handoffs between processing stages. The art and science of traditional project management and systems engineering is based upon the fundamental tenet that infinite complexity can be successfully managed. However, lean thinking demonstrates that infinite complexity increases lead, cycle, wait, and feedback times to infinity, while smallness and simplicity dramatically decreases lead, cycle, wait, and feedback times. There are at least two scenarios that exacerbate portfolio, large solution, and program performance based on the effects of complexity on poor performance. In scenario one, a large commercial organization has a small information technology budget but attempts to use SAFe to deliver a complex solution on a shoestring budget, because all of its resources are tied up in capital expenditures (building and manufacturing infrastructure). In other words, they want to use an enterprise-level business agility model to complete two-decades worth of business requirements in 12 months. In this scenario, the commercial entity failed to comprehend lean thinking principles, failed to minimize WIP, and elongated lead and cycle times, rather than decimating them. In a second scenario, a cash rich public sector enterprise attempts a complex solution because it has excess cash reserves, but fails to limit WIP while using SAFe, also elongating lead and cycle times.

14. Vi·cious·ly • Min·i·mize • WIP (*vĭsh'əs'lē • mĭn'ə-mīz' • wĭp*) Excess waste, capacity, overprocessing, delays, defects; To apply principles of lean thinking, just-in-time, demand-based waste-free production, workflow, and minimal lead and cycle times

- ✓ Apply lean thinking principles
- ✓ Pull, demand-based, just-in-time
- ✓ Optimize end-to-end flow and speed

At the heart of lean and agile thinking is the critical success factor (CSF) of viciously minimizing WIP with SAFe. WIP has two closely interrelated meanings—1. Work in progress describes the costs of unfinished goods in manufacturing, and 2—Work in process refers to materials that are turned into goods in a short time period. Both mean the amount, size, and complexity of work—Size of backlog, number of unnecessary processes or documents—Waste, or even overprocessing elements such as excess architectural, functional, or non-functional requirements actually implemented in code, but never used—Another common form of waste. Modern operating systems have 100 to 200 million lines of code, along with common productivity or application lifecycle management tools, 95% of which is unneeded, unnecessary, or never used once at all—Waste. Lean and agile theory suggests that 90% of market, customer, or end-user needs exist as hidden, inexpressible, and tacit knowledge, so fewer requirements (hypotheses) must be formed, coded, and tested before moving onto the next subset of requirements, with the goal of keeping the total set as small as possible—Often in the form of microservices. In traditional paradigms, armies of analysts and architects are hired to estimate, guess, or predict a large number of requirements for the next 30 years at a cost of millions of dollars and years of analysis paralysis—Building uncertainty or risk into traditional solutions. Programmers spent another few decades coding these mountains of requirements, and systems were fielded for markets, customers, and end users that didn't exist, while newer, better, and faster technologies emerged in 18-month cycles—Moore's Law. These hundred million lines of code applications were simply tossed in the trash at a cost of billions of dollars, when the projects succeeded at all. Public sector agencies simply added survivors to their portfolios of legacy systems. 90% of information technology budgets are tied up in operations and maintenance of unneeded legacy systems. It's unnecessary to plan a 30-year project for a 100-year system and construct a multi hundred million line of code application at a cost of billions of dollars using process and documentation heavy lifecycles from the 1970s. It's better to specify a handful of requirements, code and test them in hours and days, and ask market, customer, and end-user representatives whether they like or need the code. If the representatives don't like it or ask for changes, delivery organizations rinse and repeat many times until an optimal solution is converged upon. Internet firms in the 1990s released new code in 24-hour cycles, automobile companies make 150 software releases per day, and dot coms make 10,000 releases a day. Compare that to a 30-year lifecycle from the 1970s.

15. Min·i·mize • Ar·chi·tec·tur·al • Com·plex·i·ty (*mĭn'ə-mīz' • är'kĭ-tĕk'chər-əl • kəm-plĕk'sĭ-tē*) All-encompassing enterprise, solution, or system structural conceptual models; To resist creating big up front architectures and designs with long lead times

- ✓ Minimize enterprise architectures
- ✓ Minimize system architectures
- ✓ Minimize big design up front

An important and often-forgotten critical success factor (CSF) in SAFe is to minimize architectural complexity in SAFe. Enterprises want to bite off more than they can chew and attempt bold new transformations to stave off decades of entropy, decay, technical debt, and rot. Information technology morphs every 18 months or less, so enterprises easily fall way behind the eight ball after missing a few short cycles. Simply buy a new state-of-the-art television, cell phone, laptop computer, or other mobile device, and your favorite manufacturer releases a better, faster, and cheaper version the very next day. Imagine large enterprises which have dozens of 10 to 30-year-old legacy systems. Who would still be using a black and white cathode ray television with an antenna—Well, no one, except for large enterprises? Enterprises seem to hold their breath patching, operating, and maintaining decade old legacy systems that don't serve their needs, until the business case for chartering bold new enterprise transformations seems intuitive to even the most conservative executives. Then, the dam breaks, tens of millions and billions of dollars magically appear, stacks of unimplemented business requirements, enterprise architectures, and process and document-heavy waterfall lifecycles from the 1970s, 1980s, and 1990s suddenly appear, and everyone wants a brand new, infinitely complex system architecture to cure all known diseases. Unfortunately, this approach defies the very laws of lean and agile physics—Enterprises simply can't have everything all at one time—Even with SAFe! Truth be told, enterprises do NOT need everything all at one time for the next 50 years—That's simply nonsense! Even if enterprises could have a magical multibillion-dollar big bang architectural solution for the next 50 years today, it would be obsolete next week, just like their mobile phone! Apart from rapid technological obsolescence of big bang enterprise and system architectures, engaging in black magic is simply not the answer—Hiring an army of $500 an hour mathematicians to predict business and system requirements, building them over decades, and then surprisingly discovering the multi-million dollar requirements were never needed! So, what's the solution? Do some actual market research with ordinary, small, and cost effective cross-functional product management teams, gather a few (alleged) market, customer, and end-user requirements, rapidly build and test a small minimum viable product (MVP), and rinse-and-repeat in multiple short cycles over hours, days, and weeks, until a valuable feature is refined and converged upon. Michael Schrage says hidden solutions can be discovered with as few as five, $5,000 business experiments using his X-Team approach—Jake Knapp do do the same in two weeks with a Design Sprint!

16. Es·tab·lish • In·fra·struc·ture • Fast (*ĭ-stăb'lĭsh • ĭn'frə-strŭk'chər • făst*) Network, platform, lifecycle tools, communication systems; To establish a fast, lightweight, inexpensive, and open source information technology operating platform

- ✓ Use commercial cloud services
- ✓ Use commercial productivity tools
- ✓ Use commercial development pipelines

Establishing the program's operating infrastructure fast is also a critical success factor (CSF) in SAFe. It basically refers to the platform the SAFe program will use to construct the bold new enterprise transformation—Network, productivity tools, application life cycle management tools, DevOps pipelines, and even the target (cloud) operating environment—Remember, DevOps means to COMBINE or tightly integrate development and operations (platforms). Unfortunately, its difficult enough to find enough (experienced) lean and agile experts to run bold new SAFe transformations. Lay on top of that gauntlet, the fact that enterprises want them at half the pay of a traditional project manager, want them to serve as a one person lean and agile

project management office (PMO) to do the work of 10 to 15 people for 80 hours a week, AND, modernize the infrastructure too with a bold new development infrastructure in the form of modern lean and agile application lifecycle management tools and DevOps pipelines! When you do the economics, the gullible SAFe coach is supposed to do the work of about 25 people for half the pay, while single-handedly implementing dozens of SAFe ceremonies at the Portfolio, Large Solution, and Program levels—Serving as Solution Train Engineer, Solution Manager, Release Train Engineer, Product Manager, Scrummaster, Product Owner, etc.—And, implement all of the CMMI and ISO 9000-oriented linear waterfall ceremonies and documents too! If this is the case, then each of these groupings of activities is an Epic Enabler all by itself—Lay down infrastructure, implement SAFe ceremonies, implement traditional practices, implement agile application lifecycle management tools, implement DevOps pipelines, etc. What does all of this mean? Modern enterprises not only want to build a bold new enterprise transformation system, BUT, they want to modernize their infrastructures, implement modern agile application lifecycle management systems and DevOps pipelines, modernize their delivery practices with SAFe, AND not upset the status quo by failing to implement the outdated waterfall practices too! The answer of course is to keep it simple, don't bite off more than you can chew, and don't hide multiple MVPs in a shell game of multiple enterprise transformations for the price of one—That is, 3 or 4 development and operations transformations at one time—While maintaining 75 year backwards compatibility to 1955! Once again, DON'T fail to overlook the complexity of the development platform, keep it as simple as possible, and establish it first, or at least a basic operating model—And, take responsibility for its complexity, teach and train people how to use new tools, and don't torment them because buyers who are immersed in unnecessary complexity.

17. Lay·down • Ar·chi·tec·tur·al • Run·way • First (*lā-doun • är'kĭ-tĕk'chər-əl • rŭn'wā' • fûrst*) Rails, track, platform, structure, foundation; To establish simple, key, critical, or important architectural dependencies well before development begins

- ✓ Just-in-time architecture
- ✓ Emergent or evolutionary design
- ✓ Critical, risk-intensive impediment or blocker

A closely related critical success factor (CSF) in SAFe is to lay down the architectural runway first. Let's say the epic minimum viable product (MVP) is to build a scalable customer relationship management (CRM) system in a public cloud! It's comprised of a configured off-the-shelf CRM, the CRM middleware itself, and the public cloud infrastructure—Not to mention the network interfaces between your legacy network and the public cloud. This should be tackled by a single, small feature team with ALL of the requisite skills—Network, cloud, CRM, etc. The epic should be implemented as a vertical slice by a feature team rather than linear horizontal layers! To do so, network engineers would have to open firewalls, the public cloud would have to be architected and implemented, the CRM system would have to be installed, and the CRM system would have to be configured. The essence of an epic is that it is not a 50-year enterprise or system architecture, implementation of 15-year-old business requirements all at once, nor development of a final solution. A very small feature team can easily stand up this MVP in a matter of hours, days, and weeks! So, what are the conditions to successfully doing so—1. The team has to be assembled with the requisite technical skills and motivation, 2. The enterprise has to be willing to open its firewalls between the legacy network and public cloud, 3. The basic CRM solution has to be available even as a demo model (oftentimes cloud providers have virtual on-demand instances of middleware CRM systems already available), 4. Public cloud design and implementation skills have to be present, 5. The SAFe solution and product management and architecture teams have to identify a few salient prototype requirements, 6. The Feature team has to install or configure the CRM and program it to simulate some basic end-user (demo) functions, and 7. This vertical slice has to be demonstrated to key stakeholders in hours, days, and weeks. The basic assumptions in this scenario are that there are micro slices of mini-architectural runways that are quickly established by the feature teams—Modern public clouds have these micro architectural runways built in. And, of course, these early MVPs not only have to demonstrate basic functions such as end-user-oriented CRM capabilities, but non-functional characteristics as well, such as scalability, performance, security, etc., which can also be done with test data and load testing. Once early MVPs are demonstrated—Inexpensively and quickly without a multibillion dollar, multidecade waterfall lifecycles—The functional and non-functional capabilities have to be grown over time in an emergent and evolutionary style. The basic lesson is small vertical feature teams quickly evolve MVP slices vs. long expensive independently contracted horizontal layers!

18. Keep • Au·to·mat·ed • Tools • Sim·ple (*kēp • ô'tə-mā'təd • to͞olz • sĭm'pəl*) Utensil, device, machine, appliance, instrument; To apply simple, visual, intuitive, easy-to-use, valuable, and non-cumbersome management, development, and operations tools

- ✓ Visual information radiators
- ✓ Manual and automated combination
- ✓ Easy, useful, intuitive, productive, and helpful

An emerging, but subtly insidious critical success factor (CSF) in SAFe is to keep automated tools utterly simple! What exactly do we mean by this? Namely, lean and agile application lifecycle management systems (ALMs)! This is an old silver bullet from the 1960s and 1970s, when ALMs were known as Software Factories, Computer Aided Software Engineering (CASE) tools, and even Computer Aided Drafting (CAD), Computer Aided Engineering (CAE), Integrated Development Environments (IDEs), and, now, Computer Aided Systems Engineering toolsets. For some reason, ALMs are all-the-rage, and implementing them on bold new SAFe enterprise transformations is a two for one MVP—That is, not only save the world with an all-encompassing enterprise or systems architecture, BUT, implement a bold new ALM system across the enterprise no-less. It's difficult enough to implement a new lean and agile ALM system across a single program or project, much less an entire portfolio or solution train, because of dozens of participating suppliers and contract vehicles. Now add to that, the complexity of standardizing a bold new lean and agile ALM system across an enterprise's portfolio of programs, not just a buyer's or customer's portfolio. Unfortunately, many lean and agile ALM systems are extremely complex, require SAFe participants to spend hours entering dozens if not hundreds of epics, capabilities, features, user stories, and tasks into these ALMs, and, of

course, master the target system technology stacks. Well, there are several problems with this scenario to begin with—1. ALMs are expensive, 2. ALMs don't implement SAFe ceremonies very well, 3. SAFe is rapidly evolving obsoleting ALMs in 6 month cycles, 4. Most managers and developers have no direct experience with the myriad of ALMs, 5. ALM experts don't part with their secret mystical ALM wisdom and experience very easily, 6. Spending 80% of a your week entering and updating tasks in an ALM is simply waste, AND, worse yet, 7. Heads down ALM data entry is antithetical to heads up visual lean and agile face-to-face collaboration, keeping it simple, and adapting to change as hidden tacit market, customer, and end-user business intelligence suddenly emerges—That is, once an ALM data element is entered into the tool, it is obsolete. What's the bottom line on lean and agile ALMs? Keep it simple, focus on heads up live face-to-face visual collaboration over heads down ALM data entry, show people how to use ALM tools right away, cross pollinate ALM skills early and often (don't keep ALM mystical gnostic skills secret), don't turn SAFe projects into traditional integrated master schedules (IMSs) with ALMs, make using ALM tools fun, enjoyable, and motivating, and don't use ALM tools to create cultures of FEAR!

19. Ap·ply • Dev·Ops • Prac·tic·es (*ə-plī' • dĕv'ŏps • prăk'tĭs'əz*) Integrated, automated, development, deployment, operations; To create a lean, cloud-based continuous integration and delivery pipeline to rapidly field small high-quality microservices

- ✓ Integrated development and operations
- ✓ Fully automated deployment pipeline
- ✓ Virtual, cloud-based infrastructure

A modern critical success factor (CSF) is to apply liberal portions DevOps thinking and practices across all SAFe levels. DevOps is the combination of two words—Development and Operations—And means for information technology developers and (target system) operators to collaborate, communicate, and integrate their policies, practices, tools, and platforms. The handoff between development and operations is a major bottleneck in traditional, lean, and agile frameworks, extending lead and cycle times. The goal of lean and agile thinking to streamline the end-to-end value stream so that minimum viable products (MVPs) can be quickly pulled from development into field operations, so enterprises can measure the value of market, customer, and end-user feedback. Short lead and cycle times enable enterprises to cycle through many simultaneous business experiments faster and more frequently to tease out hidden, inexpressible, and tacit market, customer, and end-user needs. Waterfall lifecycles place all of the emphasis—work in progress, work in process, or WIP—On the front development part of the lifecycle—Vastly underestimating the resources necessary for back end operations. Although, 80% to 90% of total lifecycle costs are tied up in operations and maintenance, the basic manufacturing theory was that these downstream costs could be AVOIDED by spending 80% to 90% of the cost and effort on the front (development) end of the lifecycle. Thus, heavyweight process and document lifecycles reached their golden age in the 1970s, 1980s, and 1990s. Finally, largely with the advent of rudimentary agile methods like Extreme Programming (XP) in the 1990s, someone finally came up with the idea to automate the handoff between development and operations—Starting with testing and then the deployment itself in the early 2000s. Basic DevOps thinking led to the notion of "shift-left" thinking—That is shift the effort to the left by automating and simplifying the front end development portion of the lifecycle—An old idea from the 1970s starting with Software Factories and Computer Aided Software Engineering (CASE). However, with the advent of (virtual) public cloud computing services, DevOps is rapidly becoming a "shift-right" paradigm, where everything begins and ends in the (target) cloud platform--That is, (virtually) develop the code on the (public) target platform, pull development into the operations portion of the lifecycle—minimizing, decimating, or making development (platform) resources infinitesimal—Test it on the target cloud platform and virtually release it to millions or billions of end users simultaneously. In doing so, enterprises can cycle through many quantum and inexpensive business experiment MVPs until market, customer, and end-user satisfaction is achieved.

20. Ag·gres·sive·ly • Man·age • Con·tract • Com·plex·i·ty (*ə-grĕs'ĭv'lē • măn'ĭj • kŏn'trăkt' • kəm-plĕk'sĭ-tē*) Legal terms, clauses, agreements, acquisitions; To minimize using multiple, firm, fixed-priced end-item contracts with unnecessary dependencies

- ✓ Apply single simple short-term contracts
- ✓ Use lean and agile collaborative contracts
- ✓ Minimize long-term capital-intensive contracts

Aggressively managing contract complexity is an important critical success factor (CSF) in SAFe. This is especially true in extremely large enterprises, portfolios, programs, and public sector agencies that have multi-billion-dollar information technology budgets. Acquirers or buyers will often charter new SAFe transformations comprised of multiple contracts. They'll have one contract for cloud or infrastructure services, middleware services, end-user applications, portfolio, program, or project (PMO) services, and dozens of small business contracts. This contractual complexity is often the result of Conway's Law, where the portfolio, large solution, or program system architecture mirrors the acquirer's bureaucracy. Acquirers are divided along system lines along with their contracts. In order to acquire or buy a new system, a complex portfolio, solution train, or agile release train must be comprised of multiple functionally organized acquisition contracts that defy lean, agile, and SAFe physics. In other words, its hard to specify and form a cross cutting epic minimum viable product, vertical capability or feature slice, and vertical feature team that crosses multiple disparate (legal) contractual lines. To add insult to injury, multiple contracts can rarely be awarded at the same time, so horizontally or functionally delineated contracts may not be available or existent at the same time. In other words, its hard to build an application if the cloud infrastructure or middleware contract has not been awarded yet or the money is not available to fund the contract. Add to this, that suppliers will often protest contract awards, immediately misaligning solution trains that were dependent upon supplier alignment. Contractors understand that large enterprises suffer from vendor lock and getting left out of a large acquisition contract's technology stack means getting left out forever, conversely getting included on the ground floor means a 30 to 50 year multi-billion operations and maintenance contract for vendors. Unfortunately, innovatively new information technologies emerge in 18 months cycles, lean and agile paradigms such as SAFe are sensitive to minor 90-day fluctuation in contract awards, and composing a portfolio,

solution train, or agile release train from multiple acquisition contracts simply upsets the flow of SAFe alignments, rapid fielding of vertical MVPs, frequent feedback cycles to tease out uncertainty from requirements hypothesis tests (or business experiments), and the ability to rapidly field hundreds or thousands of DevOps deployments like commercial dot coms, mobile phone vendors, and state-of-the-art electric vehicle manufacturers. The bigger and more complex the contract aggregate is, the harder it falls, and "all the king's horses and all the king's men simply can't put humpty dumpty together again!"

21. Form • Strong • Buy·er • Sup·pli·er • Re·la·tion·ships (*fôrm • strông • bī'ər • sə-plī'ər • rĭ-lā'shən-shĭp'z*) Bond, trust, rapport, affinity, connection, association; To create close-knit informal and social bonds, teams, and attitudes with key suppliers

- ✓ Treat everyone like family
- ✓ Openly share information and data
- ✓ Involve everyone in making all decisions

Forming strong buyer-supplier relationships is an extremely important critical success factor (CSF) in SAFe. Oftentimes, a consulting or subcontracting firm is hired, retained, or contracted to perform IT services delivery, portfolio, program, or project management office (PMO) services, or SAFe transformations. This refers to the relationship between someone who forms a legal relationship for products or services. In a highly legalized relationship, oftentimes the buyer, acquirer, or prime contractor has an incontrovertibly, unshakable, or unbreakable "US vs. THEM" mentality, psychology, or attitude. Prime contractors or acquirers treat subcontractors like legalized slaves who are there to do their bidding, take orders, and perform ill-conceived behaviors, no matter how ignorant they are (the acquirer's wishes). The prime contractor simply says, "walk off a cliff," and the subcontractor unquestioningly does so—especially if they wanna keep their contract! Furthermore, the prime contractor (acquirer) suboptimizes, works for their own benefit and overtly and even subconsciously works AGAINST the subcontractor. It's sort of insanity, why would an acquirer consciously try to undermine the performance of a supplier building their IT system. Many times, an executive will command middle management adopt a new lean and agile paradigm like SAFe and command middle managers to hire a subcontractor to implement it. Today's middle managers often rose to middle management by adopting traditional manufacturing era paradigms like project management and systems engineering. Therefore, middle managers try to force traditional practices on the SAFe supplier or deliberately destroy them in order to undermine the success of the visionary executive so they can rise from middle management and assume his or her responsibilities. Part of this degenerative us vs. them mentality involves flooding SAFe subcontractor backlogs with mountains 10 to 15 year old uncoded business requirements, outdated enterprise architectures, or, if they're somewhat enlightened themselves, hundreds of UX wireframes that are more like capabilities or features than user stories or tasks. Each business requirement or UX wireframe, when unpacked, is like dozens if not hundreds of user stories that would take an entire SAFe PI to implement, and there are thousands or hundreds of these in the backlog. Lean thinking involves viciously limiting WIP—That is the number of business requirements or UX wireframes. Agile thinking involves cooperation, collaboration, and trust between all stakeholders, especially buyers and suppliers, breaking down barriers, and sharing responsibility for both the success or failure of a traditional, lean, agile, or SAFe transformation. There is NO "us vs. them," except in the Western hemisphere!

22. Cre·ate • One-Team • Col·lab·o·ra·tive • Cul·ture (*krē-āt' • wŭn-tēm • kə-lăb'ə-rə-tĭv • kŭl'chər*) Unified, joined, combined, integrated, synchronized; To create strong vertical and horizontal unity, teamwork, cooperation, collaboration, and synergy

- ✓ Foster vertical and horizontal teamwork
- ✓ Eliminate vertical and horizontal barriers
- ✓ Encourage cooperation and collaboration

Part and parcel to forming strong buyer-supplier relationships is the SAFe critical success factor (CSF) of creating a one-team collaborative culture. That is, the SAFe "team" should behave as ONE TEAM—A single, undivided, fully integrated TEAM! There should be no conceptual, logical, or physical barriers between teams. Unfortunately, the world is full of barriers, both visible and invisible, and is getting much worse! In the 1900s, teams were often collocated in a single office space, floor, or building. This was the answer to a one-team collaborative culture and still applies. Projects got in trouble when the team was on different floors, buildings, or cities! It was difficult to have program managers in one city, architecture teams 2,000 miles away, and component suppliers scattered in dozens of other regional cities. Well, it's much worse today, because teams are scattered among the world's 38 time zones! Today's teams have different cultures, ethnicities, genders, languages, customs, functions, technologies, infrastructures, etc. In today's downsizing culture, people are overallocated or multitasked amongst multiple simultaneous projects, fierce Western individualism comes into play, and consensus oriented Far Eastern cultures are experiencing an emancipation era of fierce Western individualism! Lack of teamwork is now a global pandemic, like the Black Plague or Death that killed 60% to 70% of Europe's population or nearly 200 million people in the 1300s! One answer is to legislate a policy or culture of seamless cross functional (vertical) buyer-supplier teams, but more importantly, seamless (horizontal) supplier teams—Which were often called "badgeless" teams! Many firms have different badges for buyers and suppliers—For power-status, us vs. them, master-slave, and command-and-control purposes! It gets much worse, WHEN buyers and suppliers are actually in the same city—An assumption that is rapidly evaporating in today's seamless global village—Suppliers are often forced to work in their own cities and buildings, lease space from acquirers, and prevent from using the acquirer's IT infrastructure (networks, printers, copiers, etc.). Many process models—Be it traditional, lean, agile, or SAFe—Are built upon the basic assumption that the team is collocated, they can participate in routinely scheduled face-to-face ceremonies—and are willing to do so, and they are single tasked and always available! Modern collaborative workplaces often boast of OPEN workspaces and phone booths for private conversations—Which are useless if your team is globally distributed, people telework, or people lock themselves in a phone booth because they have no people skills (and don't have time for informal conversations and collaboration because they are overallocated to 5 to 10 different global projects)!

23. Re·ward • Team·work • Be·hav·iors (*rĭ-wôrd' • tēm'wûrk' • bĭ-hāv'yərz*) Synergy, harmony, fellowship, cooperation,

collaboration; To encourage leadership to recognize, reinforce, incentivize, and attribute all accomplishments to entire team

- ✓ Teach teamwork values and principles
- ✓ Encourage teamwork behaviors and practices
- ✓ Attribute all accomplishments to greater teamwork

A closely related critical success factor (CSF) in SAFe is rewarding teamwork behaviors instead of individual behaviors! In today's global pandemic of hyper individualistic, hyper narcissists, this is particularly difficult. Western society is based upon the ideal of the solo superhero—Nobel prize winner, marathon winner, quarterback, basketball superstar, Cy Young pitcher, Pulitzer prize winning author, movie star, or world's single richest person! A program manager's favorite technique is to immediately reward the behavior of his or her favorite star performer—Explicitly or implicitly. People often promote or reward their own ethnic or gender clone, and, many times do so subliminally! They'll even go as far as to reward an individual when it was the concerted effort of the entire team that enabled their favorite ethnic or gender clone to succeed! Unfortunately, today's Western teams are top-heavy with an army of middle managers who tightly scope and bound a project in order to enable a single computer programmer to succeed—In these cases, a single middle manager is promoted to an executive leadership position when the programmer succeeds OR, worse yet, the programmer is promoted to middle management and the others are fired. It's takes a concerted effort to reprogram Western leaders, middle managers, and even programmers to recognize the critical importance of the entire team vs. a single superstar—Generally, their favorite Western clone. God help you if you're the wrong ethnicity or gender—You'll be immediately marginalized no matter what you do! When teams are rewarded, buyers will often publicly decorate the entire buyer team to humiliate and subjugate the suppliers, or the principal suppliers will publicly decorate their own teams to humiliate and subjugate their outsourced consultants or subject matter experts (SMEs). Another degenerative behavior is to single out a single team for its performance—Leadership team, portfolio, program, or project management office (PMO), architecture team, business analysts, SAFe coaches, programmers, testers, etc. Remember, if one individual or team succeeds, then everyone succeeds, and if one individual or team fails, then everyone fails. It's frustrating to notice that the more things change—Advent and adoption of SAFe—The more things stay the same—Immediate recognition of a single individual's or subteam's performance out of 100 to 200 agile release train members! Exactly, when are Western managers gonna learn this fundamental lean, agile, and SAFe value, principle, practice, and behavior? We have to eat the elephant one bite at a time, rejoice that Western managers promulgate SAFe, and hope to have true lean, agile, and SAFe leaders that recognize the incontrovertible principle of rewarding team vs. individual behavior!

24. Form • Small • Cross • Func·tio·nal • Teams (*fôrm • smôl • krôs • fŭngk'shə-nəl • tēmz*) Crew, group, squad, assembly, collective; To create a small team of people with complementary technical skills, abilities, and talents

- ✓ Keep team size as small as possible
- ✓ Rotate membership early and often
- ✓ Crosstrain and crosspollinate skills

An essential critical success factor (CSF) in SAFe is forming small cross functional teams. The operative term here is SMALL! It simply does NOT take an army to move a mountain and small groups or pairs often do GREAT things! Engineering lore is chock full of the accounts of 2 or 3 engineers who devised breakthrough innovations that employed thousands of people, resulted in billions of dollars in revenues, and saved failing corporations. Likewise, engineering lore is chock full of many more stories of large teams trying to move a mountain that failed miserably. Western managers are psychologically hardwired to believe success hinges upon large bureaucratic teams—Especially in the U.S. public sector where ample resources are available in the form of an annual $2 trillion economy. Modern studies reveal that small cross functional teams can easily devise solutions resulting in billion-dollar windfalls in one or two weeks. Historically, these were called tiger teams! There is some controversy about the level of these teams, whether it is an executive team, a bottom level developer team, or a cross cutting vertical team. This sounds a lot like (vertically sliced) feature teams? Some believe the chief executive officer (CEO) should drive the team, especially if it is a small startup, since it is often the entrepreneurially CEO who has the firm's vision in mind! Other people believe an influential technical expert or engineer should lead the team—In Japan these were called heavyweight bosses, while these were called chief programmer teams in the early 1970s! Sometimes, these are simply "business development (BD)" or proposal teams in the Western hemisphere, that work night and day in a matter of weeks or months to vision a solution, develop a strategy and tactical plan for moving mountains, and convince a buyer to hire them to execute multi-billion dollar projects. A modern practice among BD teams is to position themselves as "loss-leaders"—Severely underbid the competition, promise to make bricks without straw, and prey on the vulnerability of cost-conscious buyers! At some point, the supplier has to execute when promising vaporware to buyers, miracles, and operating at a loss! Small overallocated, multitasked, international teams are often formed to implement thousands of overscoped business requirements in 6 to 12 months, because that's all of the patience modern enterprises have for transformation initiatives! It's great for supplier executives but sad news for overallocated cross functional teams that have to work night and day at half the pay in 38 time zones! The key is to involve the lower-level technical experts in the teams, underpromise a little, viciously apply lean thinking, keep the team as small as possible, single task vs. multitask, build in a little excess capacity, and overdeliver!

25. Form • Bot·tom • Heav·y • Tech·ni·cal • Teams (*fôrm • bŏt'əm • hĕv'ē • tĕk'nĭ-kəl • tēmz*) Hands-on, coders, programmers, developers, producers; To form a hands-on technical development team vs. of an army of managers and one coder

- ✓ More developers than managers
- ✓ Developers perform all functions
- ✓ Minimalize functional specialists

A controversial, provocative, and counterintuitive critical success factor (CSF) in SAFe is forming bottom heavy technical

teams. In the early 1970s, these were called “chief programmer teams”—One “experienced computer programming team leader” leading a small team of computer programmers. Notice the terms of this equation—It’s small, there is one and only one leader, all are programmers, and there are no middle managers, executives, non-programmers like architects, business analysts, testers, etc.! So, perhaps, (extremely small) “chief programmer team” is a far better term than “bottom heavy technical team?” Even (quantumly tiny) “pair programming teams” exhibit productivity, throughput, and business value levels beyond those of “chief programmer teams!” The POINT is to unravel, unpack, and demythologize the necessity of top-heavy Western teams! Many Western businesses have a ratio of 10 managers to one programmer and some agile release trains have a ratio of 199 managers to 1 computer programmer! What IF we reduced the size of the team and inverted the ratio of managers to computer programmers, developers, or other technical hands-on personnel. We’ve emphasized the role of SAFe for computer programmers, but SAFe isn’t just for computer programmers anymore, teams can be business developers, product planners, lawyers, marketers, lawyers, hardware developers, human resources, recruiters, sales people, advertising, manufacturing, packaging, shipping, facilities, etc. The point is that if a SAFe ART can be successful with 199 managers and 1 computer programmer, then IMAGINE how much more successful it is with a vastly smaller number of managers and more computer programmers than managers. Remember the old adages, “there are too many chiefs and not enough Indians,” “too many cooks spoil the soup,” etc. The bottom line is to apply lean thinking in the form of lean portfolio management—Apply SAFe principles PROPERLY vs. as a means of promulgating traditional thinking, viciously reduce the backlog, size, complexity, and WIP, viciously reduce the number of managers, and increase the technical, development, or computer programming capacity—But don’t overallocate or overutilize them which is very common because industrial age business requirements backlogs and legacy systems still linger in the background like mountains of refuse (waste). In other words, don’t block the upstream queues with overscoped backlogs, reduce the WIP flowing through the value stream pipeline, reduce the workload, allocation, and utilization of the outprocessing development queue and watch throughput, morale, productivity, lead time, cycle time, customer satisfaction, profit, revenue, and overall enterprise performance SOAR like an eagle.

26. Cross • Pol·li·nate • Tech·ni·cal • Skills (*krôs • pŏl′ə-nāt′ • tĕk′nĭ-kəl • skĭlz*) Share, train, tutor, teach, fertilize; To proactively share technical skills within and among technical teams to improve team synergy productivity, performance, and value

- ✓ Self-motivated cross-training
- ✓ Skills and knowledge sharing
- ✓ Increase team-level competency

A hidden, invisible, and insidiously subtle critical success factor (CSF) in SAFe is to cross pollinate technical skills! Enterprises recognize the necessity of assembling cross functional teams composed of a variety of interrelated technical skills necessary to successful execute an information technology project or build a product. The team may require a project manager, architect, analyst, computer programmer, technical writer, etc. Don’t be fooled, because Western firms are expert magicians when using the word TEAM, because their real intention is to hire individuals to allocate to multiple managers and projects—Each individual is a team of exactly one person. The basic notion is that with a small army of interchangeable individuals, their skills can be mixed and matched at will to solve any enterprise problem. Therefore, many Western firms hire generalists to serve cross-cutting enterprise needs vs. the specialized needs of a single customer, project, or value stream. In the 1900s, many firms hired generalists from the top universities with degrees in physical sciences, based on the theory that their proven neuroplasticity can be repurposed to any enterprise problem, challenge, and resulting solution in the past (legacy system), present (project), or future (research and development). In other words, they were generalizing specialists. Although this looks good on paper, and often worked in the slow moving 1900s with 30 to 50-year projects, this doesn’t work well in the 20th century with a small global interchangeable workforce of 90-day projects, system releases, or SAFe Program Increments (PIs). Firms will pay any price for a technical specialist, quickly insert them into a SAFe PI, and then release them at the end of the PI (along with their skills). The key here is to “explicitly” cross pollinate teams. Yes, form a small cross functional, multi-disciplinary team, BUT, have each member cross train at least one other member of the team, if not all members of the team. If the expert gets hit by a bus, moves on to another project, or is simply released, then part of the expert’s skillset remains with the greater cross functional team. Once the individual’s skillset has been injected into his or her team, then teams should cross train one another, along with the rest of the team. Generally, this happened somewhat osmotically (implicitly or tacitly), but the trick is to intentionally, explicitly, or tangibly share one another’s skills with the rest of the team. Human beings are experts at sharing some of their knowledge, while withholding critical skills. Today’s projects have a dearth of agile lifecycle management tool expertise, hire people to perform these duties, and tool experts are experts at withholding their knowledge—This is true for any technological expertise! Most individuals can perform any job with only 20% of the knowledge of the SMEs.

27. Cre·ate • Pos·i·tive • En·vi·ron·ment (*krē-āt′ • pŏz′ĭ-tĭv • ĕn-vī′rən-mənt*) Joy, happy, safety, optimistic, encouraging; To stimulate team communication, collaboration, and performance by creating an environment of trust, safety, and enjoyment

- ✓ Positive and constructive
- ✓ Encourage safety and trust
- ✓ Communication and collaboration

Perhaps the single most important critical success factors (CSFs) in SAFe is to create a positive (working) environment. No one likes a dark, negative, mean, critical, caustic, fearful, hateful, and toxic working environment—Except for most people in the Western and Eastern hemispheres! Traditional management is based on harassing, haranguing, threatening, and otherwise forcing people to do more for less—Usually work off an overscoped backlog with an infinitesimal team capacity in the form of a big upfront loss leading business proposal, statement of work, enterprise architecture, project plan, integrated master schedule, business requirement specification, etc. Buyers are experts at forming ambiguous statements of work and then locking the services of a dopey supplier into a binding legal contract to implement a mountain of business requirements

for one-tenth the cost. It's not exactly clear who's dopier—Buyers for constructing a mountain of ambiguous business requirements or suppliers for agreeing to implement them for free—Because both stakeholders want to succeed! Creating an environment of fear is the usual method of torture, death, and punishment to force small development teams to do the bidding of 200 middle managers. However, people (developers) don't perform very well when you scare them to death—On a daily basis no less! IRONICALLY, people (psychologically) perform very well when they're happy, joyful, and optimistic—That is, the buyer, project manager, or SAFe leadership team actually creates a POSITIVE vs. NEGATIVE working environment—What a surprise! None-the-less, project managers when faced with overscoped projects, few resources, and traditional buyers, still resort to screaming, yelling, threatening, cursing, harassment, and firing people when things go awry—It's just plain old human nature (instinct). SAFe is not immune to these historical counterproductive degenerative behaviors. In today's globally distributed workforce, much of the agile release train is brought together every 90 days or so for a "Big Room" Program Increment (PI) planning event AND the "screaming" begins—The extroverted program manager simply begins screaming at all of the team leads one by one—Let the flogging begin until morale improves! Conversely, SAFe PI planning was meant to be a positive, collaborative, happy, and joyful experience for everyone to openly communicate, visibly share tacit knowledge, and constructively perform 90-day replanning—NOT get screamed at and humiliated by an extroverted program manager with a 150 IQ! Sadly, there is an inverse correlation between IQ and creating a positive working environment—One can catch more flies with honey than vinegar! POSITIVITY is the secret sauce to SAFe success!

28. Pro·mote • E·mo·tion·al • In·tel·li·gence (*prə-mōt' • ĭ-mō'shə-nəl • ĭn-tĕl'ə-jəns*) Social, friendly, communal, congenial, empathy; To create a team, culture, and environment of well-managed, constructive, and generative social interactions

- ✓ Teach and practice empathy
- ✓ Positive and encouragment
- ✓ Listen, learn, and improve

A closely related critical success factor (CSF) is to promote emotional intelligence in SAFe. Promoting emotional intelligence is also primarily a leadership behavior but is comprised of teaching team members how to exhibit positive behaviors (horizontally) towards their teammates. Since Western culture—And now Far Eastern culture—Is a society of fiercely individualistic narcissists who are all about "me, myself, and I," they are experts at what is known as "Kissing Up and Kicking Down" or "Kicking Sideways!" That is, people intuitively smile and exhibit (positive) pleasant behaviors towards their superiors—Most of the time—In order to get promoted, but are EXPERTS at exhibiting (viciously negative) unpleasant behaviors towards their peers, teammates, subordinates, and suppliers—In order to keep them from getting promoted! Unfortunately, the side effect of all of this hyper-competitive behavior is to undermine team success, project performance, and ultimately SAFe success! Do you want SAFe to succeed?—Then exhibit positive emotional intelligence upwards, sideways, and downwards! Conversely, do you want SAFe to fail?—Then exhibit negative emotional intelligence upwards, sideways, and downwards! Most (sane) people want to succeed, they want their enterprises to succeed, and they want their (SAFe) projects to succeed—They just simply do NOT know how? Likewise, they may even THINK that SAFe success largely depends upon degenerative (negative) emotional intelligence behaviors! Therefore, SAFe leaders must create policies expecting (positive) emotional intelligence, train and certify people in (positive) emotional intelligence behaviors, exhibit (positive) emotional intelligence themselves, reward and reinforce (positive) emotional intelligence behaviors, and take corrective actions when necessary. These are often called "soft skills" in the traditional vernacular, but are selectively practiced, as people learn how to exhibit soft skills towards executives and superiors, exhibit soft skills when they need something, and then digress, backslide, and degenerate into exhibiting negative behaviors (after the honeymoon phase). Once again, humans are experts at emotional intelligence during the relationship forming stage (courtship or honeymoon phase), but resort towards animalistic, cannibalistic, and other subhuman behaviors after you've been duped into joining the team (getting married). Toxic, caustic, and other nihilistic cultures are the norm, especially in the Western hemisphere, toxic cultures are tolerated in Far East where consensus is expected of societal members, and cultural toxicity is the global norm. The trick to SAFe success is to undo this global phenomenon by promulgating cultures of (positive) emotional intelligence.

29. Pro·ac·tive·ly • Man·age • Di·ver·si·ty (*prō-ăk'tĭv'lē • măn'ĭj • dĭ-vûr'sĭ-tē*) Heterogeneous and multi-cultural, ethnic, gender, racial; To proactively plan for, manage, and mitigate degenerative risks, behaviors, and outcomes of extreme diversity

- ✓ Diversity management planning
- ✓ Teach diversity empathy skills
- ✓ Mitigate risks of diverse team

Proactively managing diversity is a central, hidden, and often neglected or ignored critical success factor (CSF) in SAFe. There is often a multi-tiered ethnic, racial, and gender stratification in most enterprises. The top tier of enterprises is often white males, directors and program managers are often white males too, middle managers and other program management assistants are often white females, development staff are often Indian nationals, and there is very little diversity beyond that! Down in SAFe programs, especially in commercial or civilian public sector industries, there is greater diversity. Many times, programs are stratified with vertical male stovepipes and horizonal female layers. Many times, a single program function such as a shared service or cross functional team will be cloistered by race, ethnicity, and gender. These cloisters or clusters make it difficult to communicate, collaborate, and integrate the program functions into a single cohesive team. Ethnic groups often divide within themselves along religious, gender, or regional lines. White male program managers may be completely blind, insensitive, or unaware of stratification, not only across the entire SAFe agile release train, but within individual shared services, feature, or other cross functional teams. Program managers may create a plan, timeline, and set of operating objectives, but don't understand why throughput is blocked. Sometimes, each racial, ethnic, or gender grouping will create a clan, self-select their own leaders, and struggle for power and status within their clan—And ignore the program manager.

Many times, there are communication, cultural, or ideological barriers too, so direction has to be repeated over and over again. When conflict does occur within a clan, the program manager's first response is to fire someone, rather than resolving the conflict peacefully, constructively, and agreeably. Many times, ethnic clans will stay silent, smile, and exhibit peacefulness on the outside, but they have no intention of following the lead of an outsider, stonewall non-ethnic members, or simply argue with the technical direction that robs them of their ethnic power and status. It makes it very difficult to communicate, collaborate, and cross pollinate technical skills across racial, ethnic, and gender lines if a particular clan does not recognize the power and status of an outsider, or an insider who is not the politically correct gender, religion, or regional origin. SAFe program managers must appoint cultural attaches to identify the racial, ethnic, and gender challenges, identify risk mitigation plans, dissolve clans, and negotiate with clan leaders peacefully. Even high performers may belong to an underrepresented ethnic group and become psychologically depressed when starved with interaction with their own ethnic clan.

30. Break·down • Si·los • and • Cliques • Ear·ly (*brāk-doun • sī'lōz • ănd • klĭkz • ûr'lē*) Pit, hole, gang, pack, faction, function; To foster intergroup trust, communication, cooperation, collaboration, and performance by minimizing suboptimizing bottlenecks

- ✓ Create diverse pairs and teams
- ✓ Dynamically rotate operating teams
- ✓ Minimize naturally occurring homogeneity

A closely related but distinct critical success factor (CSF) within SAFe is to breakdown silos and cliques early, rather than tolerating them. Sometimes, silos, cliques, and clans cross racial, ethnic, and gender lines, and stratify along buyer, supplier, management layer, organizational function, or seniority—those that have been together a long time. Since modern short-lived agile teams are transient in nature, 80% to 90% of SAFe program teams are quickly constituted and reconstituted in frequent cycles, while larger enterprises, buyer teams, and other more stable organizational elements form silos, cliques, and clans among long standing members. Silos, cliques, and clans become self-managing enterprises, organizations, departments, and functions within the enterprises with informal but formidable power and status. They are like "chief programmer teams" where the chief programmer becomes the chief executive officer of the silo, clique, or clan. They behave like loosely networked terrorist clans that defy the real CEO of the enterprise, program manager's authority, direction of the lean and agile program management office (PMO), collaboration and cooperation with SAFe agile release trains, and success of the SAFe adoption. They'll pretend to go along with a SAFe adoption and seek SAFe training and certification to understand the new ideological mindset, but continue operating as their own de facto agile release train (ART) independent of the main ART. Long standing silos, cliques, and clans often have the deepest customer, market, and enterprise knowledge, communicate directly with buyers or customers outside of formal lines of communication, dig a foxhole, and stay in it while the newer transient shared services, feature, or cross functional teams struggle to gain traction. Members of long-standing silos, cliques, and clan know where all of the bodies are buried, mines and minefields lay, bear traps and barriers exist, and how to successfully navigate the customer's or enterprise's immense bureaucracy to get things done. They even know where and how to hide while new SAFe ARTs are forming, norming, storming, and performing to avoid mandatory integration into the new ART. Program managers, be it old or new, are often oblivious to long standing silos, cliques, and clans, simply choose to ignore them, or respect their independent power, authority, and deep domain knowledge of the market, customers, and enterprise. In other words, program managers often take the attitude of "if it ain't broke, don't fix it!" Unfortunately, ignoring and worshipping the independent power and status of self-managing silos, cliques, and clans leads to suboptimization of SAFe transformations, value streams, solution trains, and agile release trains. Ironically, silos, cliques, and clans are often high performing teams.

31. In·volve • Eve·ry·one • In • SAFe • Cer·e·mo·nies (*ĭn-vŏlv' • ĕv'rē-wŭn' • ĭn • sāf • sĕr'ə-mō'nēz*) Rite, ritual, service, meeting, process; To involve key stakeholders in SAFe meetings to optimize whole, create one-team culture, and enhance decisions

- ✓ Integrate all key suppliers in ART(s)
- ✓ Include internal and external teams
- ✓ Include shared services and enablers

A critical success factor (CSF) is to involve everyone in SAFe ceremonies. This means people at all levels of a SAFe portfolio, large solution, and program should participate in SAFe ceremonies, no one is exempt, and no one sits on the sidelines or operates in a vacuum. SAFe adds a little confusion to this matter by assigning epics to epic owners, capabilities to solution managers, features to product managers, user stories to agile teams, code and tests to developers and testers. There are a lot of other people like portfolio, large solution, and agile release train executives, directors, managers, and management offices—Including their staff. SAFe is a bit silent on how the work of executives, directors, managers, management offices, portfolio managers, solution managers, product managers, release train engineers, product owners, and scrum masters define and track their work. For instance, everyone knows that user stories and tasks are self-selected by developers and testers, but how does everyone else track their contributions to business value? Do management and administrative staff at the portfolio, large solution, and agile release train level have epics, capabilities, features, user stories, and tasks—Work item types in the lean vernacular? Do these non-developers have management and administrative backlogs, are they assigned to solution or agile trains, do they iterate, and, more importantly, is their management and administrative work synchronized with the rest of the solution and agile release trains? Do managers and administrators do daily standups, iteration demos, system demos, inspect and adapt, or participate alongside development teams in solution train pre-program increment planning and agile release train program increment planning? Do managers and administrators have their work item types on management, administrative, solution, or program boards ALONGSIDE development teams? Why not? More and more SPCs now treat SAFe transformation services as an agile release train (ART). Why shouldn't lean portfolio management, solution management, and ART management itself—along with shared services, communities of practice, user experience teams, system teams, and other technical and administrative teams—treat their work as an ART and participate in solution and agile

trains? If 80% of the personnel belonging to a SAFe portfolio, large solution, or program are NOT principal developers, why should the PI planning and Program Board spotlight be placed on developers, while managers and administrators hide in the dark? The ultimate goal is to keep non-developers to a minimum, make their work visible, and have them participate openly and visibly in solution and agile trains—And subsequent boards and information radiators.

32. De·liv·er • Busi·ness • Value • Fast (*dĭ-lĭv'ər • bĭz'nĭs • văl'yo͞o • făst*) Cost, price, profit, revenue, market share; To rapidly deliver valuable, high-quality products and services to market with the shortest possible lead and cycle times

- ✓ Customer or market value-focused
- ✓ Just-in-time, waste-free product or service
- ✓ Focus on small, fast, and high-quality outcomes

The single most basic, fundamental, or elemental critical success factor (CSF) in SAFe is to deliver business value fast. SAFe is not an add-on to an already bureaucratic traditional waterfall lifecycle from the 1970s—More work! Furthermore, SAFe is not intended to make information technology portfolios, large solutions, and programs a LITTLE faster, it is intended to make them a LOT faster! One of the most basic things a SAFe transformation can do is to shed the traditional lifecycle and resist becoming a HYBRID traditional lean-agile lifecycle—This is just self-defeating no matter how palatable this seems. Another key principle is to viciously apply lean and agile thinking principles and practices, keep it simple and small, resist architectural and contractual complexity, and keep teams small and focused on rapidly fielding vertical slices or minimum viable products (MVPs). Also, keep it visual, transparent, face-to-face, collaborative, and positive—And watch out for self-defeating suboptimizations like high power-distance buyer supplier relationships, debilitating competition between supplier teams, and pseudo-terrorist cells like silos, cliques, and clans divided along seniority, race, ethnicity, gender, regional origin, and even religion. Focus on small, tightly scoped epic MVPs with viciously small capability, feature, and user story sets, and pull these through the end-to-end value streams as quickly as possible. Dot coms and other state-of-the-art firms like electric automobile manufacturers and mobile phone platforms pull hundreds, thousands, and tens of thousands of epic MVPs through their enterprise to billions of global markets, customers, and end users every day! Remember the essential role of DevOps and public cloud computing platforms. Push the development pipeline to the right—Into the cloud—Pull the microservices through the virtual infrastructure and into the hands of billions of global end users fast! There's absolutely no reason why SAFe portfolios, large solutions, and programs need to wait years or decades to pull epics, capabilities, features, user stories, and code through their value stream—There's simple no excuse! SAFe is not about codifying the waterfall lifecycles of manufacturing era bureaucracies and governance boards—Conway's Law—And there's no reason to turn a SAFe portfolio, large solution, or program into a 21st century bureaucratic gauntlet of manually-intensive top-heavy decision-making boards. To be agile is to viciously push down, delegate, empower, ease, and automate the process of pulling epics, capabilities, features, user stories, and code (microservices) through end-to-end enterprise value streams at the speed-of-light to billions of global markets, customers, and end users. In other words, deliver business value fast—Today, with SAFe!

SAFe Summary

So, what have we learned on this short treatise about the principles and practices of successfully implementing SAFe? We've learned that the 21st century now belongs to lean and agile enterprise, portfolio, large solution (system of systems), and program management frameworks. We've also learned SAFe is the preferred lean and agile enterprise framework for global firms after a relatively short period of time—10 to 15 years—Riding on the shoulders, tidal wave, and global movement of lean and agile giants. In addition, we've learned SAFe began as an agile program management framework, but evolved into a large solution, portfolio management, and enterprise business agility framework. More importantly, we've learned that SAFe is a LEAN and AGILE framework, not simply a small addition to the pantheon of traditional waterfall process and document heavy enterprise military frameworks from the 1970s. As such, we've learned that lean and agile values, principles, and practices must be viciously applied to limit WIP across the entire end-to-end enterprise value stream to decimate lead and cycle times. Therefore, it's a mistake to jam unimplemented business requirements specifications and enterprise architectures into SAFe backlogs and expect lean and agile performance outcomes—Short lead and cycle times. Also, we've learned that SAFe has a rather simple and highly recommended implementation roadmap that SHOULD be followed in order to extract the maximum benefits from SAFe. That is, simply bootstrapping SAFe onto traditional frameworks, piecemealing SAFe, or mixing SAFe and waterfall practices is self-defeating. Probably the single most important thing we've learned is to dissolve and blur divisive vertical and horizontal buyer supplier relationships to achieve the benefits of teamwork, collaboration, communication, trust, performance, throughput, and quality.

32 Principles of Highly-Successful Scaled Agile Framework (SAFe) Implementations

1. **Establish Topdown SAFe Leadership**—Select a powerful leader who will champion SAFe from very top
2. **Form SAFe Leadership Team**—Form a powerful coalition for planning, monitoring, and improving SAFe implementation
3. **Apply Lean and Agile Thinking**—Deliver valuable, high-quality products and services with fast lead and cycle times
4. **Resist Hybrid Traditional Practices**—Resist refining heavy non-value adding processes, documents, and overprocessing
5. **Instill SAFe Thinking Early and Often**—Continuously learn, practice, and master SAFe values, principles, and ideals
6. **Follow SAFe Roadmap**—Observe and comply with the 12-stage SAFe implementation roadmap
7. **Train and Certify Everyone**—Subject stakeholders for formal classroom instruction and evaluation
8. **Form SAFe Product Management Team**—Charter a team or group to translate epics into selected capabilities and features
9. **Form SAFe Architecture Team**—Charter team or group to define solution or system architecture for capabilities/features
10. **Keep It Simple**—Keep the solution or product scale, scope, and risk as simple, narrow, and easy as possible
11. **Keep It Small**—Keep solution or product size, magnitude, and dimensions as small, diminutive, and pintsized as possible

12. **Keep It Visual**—Keep solution or product roadmaps, kanbans, program increment plans, and iterations plans openly visible
13. **Minimize Complexity**—Keep solution or product architectures simple, minimalistic, modularized, and narrowly-scoped
14. **Viciously Minimize WIP**—Apply lean thinking, just-in-time, demand-based waste-free production, and workflow principles
15. **Minimize Architectural Complexity**—Resist creating big up front architectures and designs with long lead times
16. **Establish Infrastructure Fast**—Establish a fast, lightweight, inexpensive, and open source IT operating platform
17. **Laydown Architectural Runway First**—Establish simple, key, critical, or important architectural dependencies early
18. **Keep Automated Tools Simple**—Apply simple, visual, intuitive, easy-to-use, valuable, and non-cumbersome tools
19. **Apply DevOps Practices**—Create lean cloud-based continuous integration/delivery pipeline to rapidly field microservices
20. **Aggressively Manage Contract Complexity**—Minimize using multiple, firm, fixed-priced, unnecessary end-item contracts
21. **Form Strong Buyer Supplier Relationships**—Create close-knit informal and social bond, team, and attitude with suppliers
22. **Create One-Team Collaborative Culture**—Create strong unity, teamwork, cooperation, collaboration, and synergy
23. **Reward Teamwork Behaviors**—Encourage leadership to recognize, reinforce, incentivize, and attribute teamwork
24. **Form Small Cross Functional Teams**—Create small team of people with complementary technical skill, ability, and talent
25. **Form Bottom Heavy Technical Teams**—Form hands-on technical development team vs. of manager army and one coder
26. **Cross Pollinate Technical Skills**—Proactively share technical skills within and among technical teams to improve synergy
27. **Create Positive Environment**—Stimulate team communication, collaboration, and performance with trust, safety, and fun
28. **Promote Emotional Intelligence**—Create team, culture, and environment of constructive, and generative social interactions
29. **Proactively Manage Diversity**—Proactively plan, manage, and mitigate risks, behaviors, and outcomes of extreme diversity
30. **Breakdown Silos and Cliques Early**—Foster intergroup trust, communication, cooperation, collaboration, and performance
31. **Involve Everyone in SAFe Ceremonies**—Involve key stakeholders to optimize whole, create team, and enhance decisions
32. **Deliver Business Value Fast**—Rapidly deliver valuable, high-quality products and services to market with short lead times

However, we've also learned a few other critical success factors (CSFs) along the way too. Namely, training and certifying key stakeholders and then all of the members of portfolios, large solutions, and programs. Applying lean and agile portfolio, large solution, and program (Kanban-based) pipeline management is absolutely essential. Thus, in order to limit WIP, enterprises must keep epics, capabilities, solution architectures, features, system architectures user stories, tasks, code designs, and software modules (microservices) simple, small, and complexity free. We've also learned that lean and agile thinking is a fundamentally a live, face-to-face, visual, and transparent enterprise, portfolio, large solution, and program planning system. Therefore, skipping live face-to-face solution and program increment planning events, burying and hiding people's attention and outputs in complex agile life cycle management tools, and allowing key stakeholder groups to be nonparticipative and hidden is antithetical to SAFe. Cross functional teams should be small, multi-disciplinary, vertically integrated, and cross-pollinating to maximize the skills and productivity of the team over the individual. Sadly, we've learned that fierce Western individualism, narcissism, overallocation, multitasking, and non-collaboration is a global pandemic that is difficult to overcome with 20th century ceremony (meeting) intensive management and technical frameworks—Even SAFe! Surprisingly, we've learned that public sector agencies—Civilian, military, and intelligence—Are flocking to SAFe to rapidly field new transformational enterprise systems, they often have too much money due to multiple trillion dollar national economies, they tend to overscope acquisitions—defying lean and agile physics, and the use of multiple, legalistic phased (unsynchronized) firm fixed-priced contracts to build horizontal layers doesn't work too well.

Most importantly, we've learned that the key to successful enterprise SAFe transformations is creating and maintaining a positive working environment, dissolving vertical and horizontal collaboration barriers, and teaching and rewarding emotional intelligence, teamwork, collaboration, and communication skills. Likewise, traditional management behaviors such as screaming, yelling, reprimanding, and using tactics of fear, harassment, and threats are antithetical to the success of lean and agile frameworks such as SAFe. Unfortunately, poor behavior is all too human even among the best of us, but this historic pandemic can be easily be overcome by exhibiting TRUE servant leadership behaviors—vs. the instinctual urge to hurt someone fast as a form of motivation. Another, perhaps even more insidious counterproductive human behavior is the instinctual desire to quickly amalgamate into silos, clans, and cliques along lines of seniority, gender, ethnicity, race, language, color, religion, and even regional origin. It's easy to spot a silo, clique, and clan acting like a self-managing counterproductive terrorist cell, but sometimes, multiple simultaneous overlapping vertical and horizontal silos, cliques, and clans will exist that are hard to spot. Today's SAFe program managers are faced with numerous challenges—Instituting SAFe, instilling lean and agile thinking, exhibiting servant leadership, following the SAFe roadmap, resisting traditional hybridism, training and certifying people, stopping enterprise architectures and business requirements from getting dumped into SAFe backlogs, and preventing SAFe from becoming a waterfall. However, the greatest challenge a SAFe leader faces is coordinating the efforts of global part-time, multi-tasking distributed narcissists—And dissolving naturally occurring barriers of silos, cliques, and clans along race, ethnicity, gender, religion, language, color, and region.

SAFe got off to a slow start in 2007, some people immediately associated SAFe with degenerative lifecycles from the 1990s—A reputation that's been hard to shake! However, SAFe is now leaner and meaner and an unstoppable global phenomenon! In spite of SAFe's immense popularity and extremely large number of trainees eclipsing traditional certifications from the 1970s, SAFe is still new, practical hands-on SAFe experience is rare—But on the rise, there are still many SAFe skeptics, and SAFe adoptions are a little rough (suboptimal). The worst part is jamming too many left over business requirements from the early 2000s into SAFe backlogs—Each the size of a Mack Truck, skipping Program Increment (PI) Planning, doing PI Planning in advance by individual Scrum team, and jamming dozens of user stories and tasks into agile lifecycle management tools instead of highly-visible solution and program boards, failing to communicate during PI planning (ignoring people), and using PI planning to promulgate cultures of fear—Flogging people until morale improves. There are still many traditional middle managers lurking in the background like dark matter, holding their teams back from participating in SAFe rollouts, encouraging other middle managers to abstain from SAFe ceremonies, and convincing directors and other executives to back away from SAFe transformations—This was also true of traditional transformation initiatives. Transformation is an executive prerogative, not a developer's, top-down SAFe leadership is imperative, and SAFe leaders must seek out and mitigate the fears of traditional non-participants as well—That's another story!

Further Reading

- Campbell-Pretty, E. (2016). *Tribal unity*: *Getting from teams to tribes by creating one team culture*. Richmond, Vic: Pretty Agile.
- Campbell-Pretty, E., & Wilson, A. L. (2019). *The art of avoiding a train wreck*: *Practical tips and tricks for launching and operating SAFe agile release trains*. Pretty Agile Pty.
- Dalton, J. (2019). *Great big agile*: *An os for agile leaders*. New York, NY: Apress Media.
- Hopp, W. J. (2008). *Supply chain science*. Long Grove, IL: Waveland Press.
- Kane, L. (2018). *SAFe PI planning*: *A step-by-step guide*. Torrance, CA: Agile One Media.
- Kaye, B. & Jordan-Evans (2014). *Love em or lose em*: *Getting good people to stay*. San Francisco, CA: Berrett-Koehler.
- Knaster, R., & Leffingwell, D. (2018). *SAFe distilled*: *Applying the scaled agile framework for lean enterprises*. Boston, MA: Pearson Education.
- Leffingwell, D. (2007). *Scaling software agility*: *Best practices for large enterprises*. Boston, MA: Pearson Education.
- Leffingwell, D. (2011). *Agile software requirements*: *Lean requirements practices for teams, programs, and the enterprise*. Boston, MA: Pearson Education.
- Leffingwell, D. (2018). *SAFe reference guide*: *Scaled agile framework for lean enterprises*. Boston, MA: Pearson.
- McConnell, S. (2019). *More effective agile*: *A roadmap for software leaders*. Bellevue, WA: Construx Press.
- Pearson, C., & Porath, C. (2009). *The cost of bad behavior*: *How incivility is damaging your business and what to do about it*. New York, NY: Penguin Group.
- Pound, E. S., Bell, J. H., Spearman, M. L. (2014). *Factory physics*: *How leaders improve performance in a post-lean six sigma world*. New York, NY: McGraw-Hill Education.
- Reinertsen, D. G. (2009). *The principles of product development flow*: *Second generation lean product development*. New York, NY: Celeritas.
- Ries, E. (2017). *The startup way*: *How modern companies use entrepreneurial management to transform culture and drive long-term growth*. New York, NY: Crown Publishing.
- Rico, D. F. (2018). *Lean & agile contracts*: *21 principles of collaborative contracts and relationships*. Retrieved June 29, 2018, from http://davidfrico.com/collaborative-contract-principles.pdf
- Rico, D. F. (2018). *Lean & agile organizational leadership*: *Establishing vision, guidance, & trust to unleash enterprise competitiveness*. Retrieved February 20, 2018, from http://davidfrico.com/rico18g.pdf
- Rico, D. F. (2018). *Using SAFe 4.5 to transform a $200 million U.S. healthcare portfolio*. Retrieved November 19, 2018 from http://davidfrico.com/safe-case-study-ii.pdf
- Rico, D. F. (2019). *32 attributes of successful continuous integration, continuous delivery, and devops*. Retrieved September 27, 2019, from http://davidfrico.com/devops-principles.pdf
- Rico, D. F. (2019). *Business value of lean leadership*: *22 Attributes of successful business executives, directors, and managers*. Retrieved April 27, 2019, from http://davidfrico.com/rico19a.pdf
- Rico, D. F. (2019). *Business value of lean thinking*: *Capitalizing upon lean thinking principles to rapidly create innovative products and services*. Retrieved November 11, 2019, from http://davidfrico.com/rico19d.pdf
- Rico, D. F. (2019). *Evolutionary architecture*: *24 principles of emergent, organic, and highly-adaptive design*. Retrieved September 3, 2019, from http://davidfrico.com/evolutionary-architecture-principles.pdf
- Rico, D. F. (2019). *Lean & agile enterprise frameworks*: *Using SAFe 4.6 to manage U.S. government agencies, portfolios, and acquisitions*. Retrieved June 12, 2019, from http://davidfrico.com/rico19b.pdf
- Rico, D. F. (2019). *Lean leadership*: *22 attributes of successful business executives, directors, and managers*. Retrieved August 28, 2019, from http://davidfrico.com/lean-leadership-principles.pdf
- Rico, D. F. (2019). *Piloting the scaled agile framework (SAFe) in a top 10 u.s. energy firm*. Retrieved May 15, 2019, from http://davidfrico.com/x-safe-case-study-iii.pdf
- Rico, D. F. (2019). *Top SAFe videos*, Retrieved April 7, 2019, from http://davidfrico.com/top-safe-videos.htm
- Rico, D. F. (2020). Business value of CI, CD, & DevOps(Sec): Scaling up to billion user global systems using containerized microservices. Retrieved January 26, 2020, from http://davidfrico.com/rico20a.pdf or http://youtu.be/qrWRoXSS9bs
- Rico, D. F. (2020). *Business value of lean thinking*: *Capitalizing upon lean thinking principles to rapidly create innovative products and services*. Retrieved January 29, 2020, from http://davidfrico.com/rico20b.pdf or http://youtu.be/wkMfaPAxO6E
- Rico, D. F. (2020). *SAFe 5.0 videos*, Retrieved January 21, 2020, from http://davidfrico.com/safe-5.0-videos.htm
- Rico, D. F. (2020). *Using large solution SAFe in a high-profile U.S. civilian public sector agency*. Retrieved January 29, 2020, from http://davidfrico.com/y-safe-case-study-iv.pdf
- Rico, D. F., Sayani, H. H., & Sone, S. (2009). *The business value of agile software methods*: *Maximizing ROI with right-sized processes and documentation*. Ft. Lauderdale, FL: J. Ross Publishing.
- Rothman, J. (2016). *Agile and lean program management*: *Scaling collaboration across the organization*. Victoria, Canada: Practical Ink.
- Smith, P. G. (2018). *Flexible product development*: *Agile hardware development to liberate innovation*. San Francisco, CA: Jossey-Bass.
- Soukath-Ali, M. M. (2017). *Get SAFe now*: *A lightning introduction to the most popular scaling framework on agile*. Chennai, India: Ali Publications.
- Thompson, K. W. (2019). *Solutions for agile governance in the enterprise* (*SAGE*): *Agile project, program, and portfolio for development of hardware and software products*. Vancouver, CA: Sophont Press.
- Yakyma, A. (2016). *The rollout*: *A novel about leadership and building a lean-agile enterprise with SAFe*. Boulder, CO: Yakyma.

Case Studies of Highly-Successful Scaled Agile Framework (SAFe) Implementations

- Global Energy Firm. *The first SAFe case study involves a global energy firm. What makes this case unique is that the energy sector is a staunchly traditional industry known for being overly regulated, slow moving due to its monopoly on utility customers, and, of course, the fact that it is dependent upon a mission and safety-critical infrastructure. So much so, that 99.9% of its resources are tied up in operating its immense capital-intensive infrastructure—Buildings, powerlines, generating stations, networks, etc. Therefore, its information technology budgets were infinitesimal in spite of its immense revenues, excess revenues were tied up in capital improvements and payouts to shareholders, and whatever information technology projects it did have were very traditional in-nature. Like most information technology executives, its leaders wanted to push out new information technology capabilities quickly, utilize modern information technologies to replace manually intensive customer support systems, and, of course, catch up with the mobile and Internet revolutions. Therefore, its executives demanded a SAFe-based digital transformation initiative for its consumer facing systems. It's important to note that many of its customers or consumers were other energy firms subleasing its power and infrastructure or energy traders selling its capabilities on the open market, but the focus of this SAFe initiative was on the ordinary household consumer. The first thing it did was to create a modern user experience (UX) department, construct a mountain of beautiful overscoped UX wireframes, and then retain a big six consultant to hire information technology delivery personnel with modern skills, teach them basic agile frameworks, and then use SAFe to integrate the work of multiple teams. Clearly, a success factor was that its top leadership demanded lean, agile, SAFe, and modern information technologies, which was pretty rare among regional enterprises whose executives weren't quite so enlightened. The digital transformation services provider quickly hired capable developers, stood up basic lean and agile teams, and began at once to churn on the overscoped UX wireframes—Each of which was bigger than an Epic, Capability, or Feature, but treated like a User Story. Furthermore, part-time Product Owners assumed the role of Project Managers and demanded that each minute of every developer's day be accounted for to crank on the overscoped UX wireframes. Scrummasters, of course, were ignored, and Product Owners were given full autonomy from collaborating with Agile coaches and Scrummasters. Although, there were six or seven active agile teams, traditional middle managers relieved over half of them from participating in SAFe due to distrust in SAFe, and, of course, they were busy churning on mission critical UX wireframes. In spite of this, SAFe coaches rolled out its first SAFe Program Increment (PI) planning teams quickly, participating middle managers and agile release train (ART) participants immediately realized the value, and productivity experienced an immediate increase. SAFe created sort of a space-race with non-participants to do the energy firm's first software release in decades.*
- Public Sector Financial Agency. *The second SAFe case study involves a large public sector financial services agency. What makes this case unique is that the public financial services sector is staunchly conservative, they are deeply entrenched in waterfall lifecycles, they have thousands of intelligent planners accounting for every possible risk contingency, and moving fast is not their goal. In spite of these obstacles, this agency viewed itself as a thought leader among its 170 or so other peers, and they began dabbling in basic agile frameworks. Oftentimes, they simply bootstrapped an agile team or two on a waterfall lifecycle, but not without much peril as their employees often reported agile behaviors to the inspector general as dangerous methods. By the late 2010s, this agency began a SAFe initiative for publicly facing web services, experienced some success, and decided to use SAFe to rescue a four-year-old enterprise system stuck in first gear using a traditional waterfall. This was also a topdown leadership driven SAFe initiative, the SAFe implementation roadmap was codified into and integrated master schedule down to the very last detail, and SAFe was formally kicked off to rescue their waterfall-based transformation initiative. Multiple disparate contracts were launched, immediately protested, and horizontal epic enablers immediately became unavailable. However, the core web services to extract vital data from legacy systems immediately began using SAFe and experienced much success. Unfortunately, there were hundreds of program participants from multiple disparate functionally organized departments, they were all using waterfall practices, and SAFe development teams were being starved from key features and user stories. SAFe Scrummasters often worked behind the scenes to tease out features from relevant product managers, preplanned SAFe PIs, and easily completed program boards in minutes once PI planning began. The program manager appointed a SAFe leader—Sort of a program level lean and agile program management office (PMO), an agile lifecycle management office also existed to help push SAFe ceremonies and automation, and the program manager also participated and pushed SAFe ceremonies hard from the top-down. Other than the blocked contracts, this was a textbook SAFe rollout, everyone was encouraged to participate in SAFe ceremonies, and it freed itself from the constraints of its waterfall lifecycle processes and documents. In spite of its many challenges, its program manager refused to surrender, revert to traditional practices, nor abandon the information technology transformation initiative—All of which were very tempting—And begrudgingly agreed to keep on moving forward, reboot the initiative six months in, and try to free up the blocked contracts. It's important to note that the enterprise itself had an agile management office (AMO), pushed basic lean and agile frameworks hard—And even competing SAFe frameworks, and was highly skeptical of its two major SAFe initiatives—Both of which were exemplars in the entire portfolio of public sector agencies. In other words, there was an internal space race for competing agile frameworks.*
- Public Sector Healthcare Agency. *The third SAFe case study involves a large public sector healthcare agency. What makes this case unique is that the public healthcare sector is staunchly conservative, they are deeply entrenched in waterfall lifecycles, and they consider themselves guardians of mission and safety critical waterfall methodologies and systems due to privacy and security concerns. Like many traditional public sector agencies, they began dabbling in basic lean and agile frameworks around 2010, bootstrapped these onto waterfall lifecycles, and experienced some success—They were the first projects to deliver new capabilities in decades. From there, they began exploring agile project management frameworks, and eventually SAFe. This was actually two public sector agencies—One for national systems and the other for oversight of state-level distributed systems with some national systems for analyzing state-level data. The second state-level oversight agency kicked off a SAFe initiative first, but its leadership team was more interested in standing up a public cloud service to host its legacy data than bother with SAFe ceremonies, so that SAFe initiative quickly died on the vine—They simply didn't have time for SAFe in their leadership's opinion. The first and larger agency responsible for building a portfolio of federal systems kicked off several SAFe initiatives withing a year or so. It's leadership demanded SAFe implementation, properly, because its suppliers were still pushing waterfalls, it stood up a well-funded lean and agile centers of excellence (LACE), and it kicked off its first SAFe Program Increments (PI) at the agency level in the history of its enterprise. These Agile Release Trains (ARTs) began delivering new capabilities and features, business value was realized very quickly, and it didn't bother with its well ingrained waterfall lifecycle. Ironically, the successful SAFe initiatives were driven from the leadership and government program managers—Not the supplier program managers—The traditional suppliers didn't know what hit them—And the suppliers did what their government leaders and LACE demanded they do—Execute SAFe ceremonies! Unfortunately, this public sector healthcare agency was so entrenched in waterfall lifecycles from the 1970s, it created a cottage industry of staunchly traditional suppliers that were simply not psychologically wired for lean and agile thinking, agile frameworks, nor SAFe! Oftentimes, even when contracts demanded suppliers use basic agile frameworks, traditionally minded suppliers were the first to complain about the pain of using agile frameworks. This wasn't without cause of course, as the agency often jammed 10 to 20-year-old overscoped business requirement specifications into under-resourced agile backlogs and then flogged suppliers to death for having jammed (frozen) backlogs. When suppliers asked business and product owners to be trained in lean and agile thinking—Reduce WIP, downsize business requirements, and use agile properly—The agency refused. None-the-less, after some initial fits and starts with bottoms up SAFe initiatives, this agency succeeded with topdown leadership-driven SAFe, an obvious CSF.*