

DAVE'S NOTES—WHAT'S THE BUSINESS CASE FOR SCALED AGILE FRAMEWORK (SAFe) SYSTEM DEMOS?

What is the Scaled Agile Framework (SAFe) System Demo?

- The SAFe system demo is a significant event that provides an integrated view of NEW features for the most recent iteration (sprint) delivered by all the teams in the Agile Release Train (ART).
- The SAFe system demo occurs at the end of every iteration (sprint). It provides an integrated, comprehensive view of the NEW features delivered by the ART over the past iteration (sprint).
- SAFe system demos offer ART stakeholders an objective fact-based measure of current system-level PROGRESS within the Program Increment (PI) or quarter—It's the true measure of ART velocity and progress.
- Planning and presenting a useful SAFe system demo requires only minor work and preparation by the teams (but it's the ONLY way to get the fast feedback needed to build the right solution).

Source of SAFe Demo Definition. <https://www.scaledagileframework.com/system-demo>

What are the common reasons the SAFe System Demo is avoided?

- Many SAFe leaders refuse to apply SAFe's system demo because it takes too much time to integrate all products (i.e., SAFe is often applied to brittle monolithic legacy systems that don't lend themselves to frequent DevOps deployments).
- As a result, SAFe leaders prefer to apply inferior practices such as integrated master schedules, EVM, daily burndowns, sprint metrics, PI metrics, velocity, planned vs. actuals, monthly, quarterly, and annual reporting, and other vanity metrics.
- So, how do we get SAFe leaders and managers to apply SAFe system demos as a superior objective measure of system progress measurement, and wean them off of traditional reporting based on vanity metrics (what questions could we ask)?

Questions that may cause SAFe leaders to consider using SAFe System Demos?

- Do SAFe leadership teams and other major stakeholders want a quick, easy, and inexpensive way to visually measure the progress of a SAFe-based value stream, product suite, individual product, acquisition program, or other related project?
- Would your customers, end-users, and budgeting authorities prefer to see live operational (verifiable) evidence of SAFe progress (other than traditional reports, schedule performance, earned value management, and velocity metrics)?
- Do customers, end-users, market representatives, and technical authorities want to see live operational demos of the latest upcoming system changes, enhancements, and capabilities (other than monthly, quarterly, or annual status reports)?
- Does your SAFe leadership team wish to correctly leverage built-in SAFe practices and checkpoints as a continuous business case to justify ongoing funding (other than vanity metrics with little overall business, market, end-user, or technical value)?

Quick Pointers and considerations to get SAFe System Demo conversations started?

- It's okay to have quick-and-dirty SAFe system (component) demos (not necessarily the whole system).
- It's okay to demo key system (components) that are NOT (necessarily) on the SAFe program board.
- SAFe system demos are primarily designed to show (working) system (component) demos to the ART.
- Remember the Agile Manifesto—**"We value Working Software over Comprehensive Documentation!"**
- You don't want to demo the whole system if you could—You only want to demo CHANGES, progress, and improvements.
- Think of Microsoft Office—There isn't time to demo ALL of its features every two weeks—Only the (component) CHANGES.
- You may also want to demo architectural runways, tools, shared services, UX designs, documentation, roadmaps, etc.

SAFe leaders also struggle to apply SAFe Program Boards (key to System Demos)?

- Leaders prefer iteration boards, stories, points, load, and capacity (as a proxy for integrated master schedules and EVM).
- Agile teams are accustomed to think and plan in terms of user stories vs. features (especially if that's what leaders want).
- It's difficult to identify features (especially if integrated master scheduling is applied instead of agile product management).

Detailed considerations and rationale for justifying and applying SAFe System Demos?

- System demos are a bit misunderstood and neglected in SAFe (primarily because we're used to traditional vanity metrics).
- People are consumed by overloaded integrated master schedules, requirements backlogs, and periodic performance reports.
- This is a symptom of a deeper cause—Monolithic legacy systems requiring manually intensive bi-annual system releases.
- SAFe leaders and architects are consumed by measuring the glacial effort of delivering brittle code monoliths with EVM.
- Nevertheless, a SAFe ART may be developing an INTEGRATED system (or a SUITE of interrelated or individual products).
- The purpose of SAFe system demos is to demonstrate individual features completed on an iteration-by-iteration basis.

- For instance, in the example BELOW, you MAY demo the 4 features by the Unicorns, Dolphins, and Eagles (after Sprint 1):

	Sprint 1	Sprint 2	Sprint 3
Milestones			→ [Orange Box]
Unicorns	[Light Blue Box] [Light Blue Box]	[Light Blue Box]	[Light Blue Box]
Dolphins	[Light Blue Box]	[Red Box]	[Light Blue Box]
Bears	[Red Box]	[Light Blue Box]	[Light Blue Box]
Eagles	[Light Blue Box]	[Light Blue Box]	[Light Blue Box]
Iguanas	[Red Box]	[Light Blue Box]	[Light Blue Box] [Light Blue Box]

- The 4 features in Sprint 1 in-blue ABOVE may or may not be INTEGRATED (so it's okay to demo them individually).
- SAFe focuses on WORKING SOFTWARE (but features may be designs, spikes, models, runways, infrastructure, tools, etc.).
- It's okay to demo NON-WORKING CODE which developers are AFRAID to do (believing only WORKING CODE is demo'd).
- People only add stories to backlogs that result in WORKING CODE (although 80% of the outputs are NOT working software).
- That's an HONORABLE goal, but it's unrealistic (SAFe recognizes spikes, architectural runways, analytical products, etc.).
- In [dual-track development](#) parallel UX teams make wireframes, mockups, and other artifacts (you'd want to demo these too).
- Agile product managers develop roadmaps, architectures, designs, etc. in parallel for future PIs (these should be demo'd).
- Future visions, roadmaps, architectures, and designs IMPACT current PIs (so save time now or pay unnecessary churn later).
- You don't have to demo every completed feature so only demo the important features completed after sprints or iterations.
- Focus on working software or systems on SAFe Program Boards (but don't ignore important background components).
- SHORT time-boxed demos of about 5-10 minutes are great (it should take 15-20 minutes to demo the 4 features above).
- Most people are not fans of long boring labor-intensive demos, so keep them short, sweet, to-the-point, and highly impactful.
- System demos promote transparency, communication, information and knowledge sharing, and dependency management.
- There are very long demos, but you should only demo the important elements, changes, and improvements of your features.
- Make video recordings if possible, fast forward across the key elements, and highlight the changes, fixes, and value-added.
- Agile teams focus on having working software at the end of each Sprint and prefer LONG demos (so be patient with them).
- Developers should talk through their demos, not just kick off a 60-minute video recording and stay silent (that's not valuable).
- Developers are shy, so encourage them to narrate demos in a positive way (or have other stakeholders narrate for them).
- Agile teams may refine architectural runways that may not be on program boards (so perform demos of periodic snapshots).
- Sprints result in small user stories and tests, but customers want to see cross-cutting OPERATIONAL PRODUCT DEMOS.
- Customers, end users, and stakeholders may not want to see 150 automated acceptance tests screaming by at top speed.
- Have testers run through manual cross-cutting operational user scenarios after PIs (in addition to acceptance test reports).
- Customers want operational demos so have business analysts, systems engineers, or testers demo useful real-life scenarios.
- Legacy system developers have to deliver to staging servers (so be prepared for live customer demos of staged systems).
- Demos off of staging servers are a good proxy for actual system releases if you can't easily deploy to your production fabric.
- Demos don't always have to be something on program boards (as customers may want to see live data centers in action).
- Schedule period demos of data center servers, storage solutions, network switches, and other live interoperability demos.
- Customers want to see their multi-million-dollar budgets result in WORKING OPERATIONAL SYSTEMS (vs. vanity metrics).
- SAFe leaders may force unscheduled demos (to show business value of their investments and build trust with customers).
- Schedule quarterly demos for key stakeholders of SAFe portfolios and larger solutions trains composed of multiple ARTs.
- Insist upon two or three demos after every sprint, program increment, or quarterly portfolio review for key stakeholders.

Footnote. Many companies are investing in business experimentation platforms and web services that allow UX designers to construct and DEMO live rapid prototypes, mockups, and spikes to real customers (it's the wave of the future) – Top Dot Coms actually modify their production systems in real time (for subsets of users), gather feedback, and then rollout the changes to everyone (if the experiments yield business value) – These include Amazon, Microsoft, Google, Tesla, Apple, etc. – Few government and other commercial firms have the ability, culture, infrastructure, etc. to modify their customer facing production fabric in real-time to a few end-users, gather feedback, and rollback/rollout results in real time – But, again, that's the wave of the future – If we were all cloud based, we could have a production server and a mirrored staging server, make changes to the staging server (and demo these changes), and switch our customers to the staging server (if it passes all tests) – That is, make the staging server the production server in a fraction of a second (and rollback to the prior server if something goes wrong).

Email. dave1@dauidfrico.com • **Background.** <http://dauidfrico.com/daves-background.pdf> • **YouTube.** <http://www.youtube.com/c/DavidRico/videos>
 • **Slideshare.** <http://www.slideshare.net/dauidfrico/presentations> • **Business Card.** <http://dauidfrico.com/bcard.jpg>
 • **Agile Book.** <http://dauidfrico.com/agile-book.htm> • **Website.** <http://dauidfrico.com>