VA Mobile App Response

7/18/2018

C: (703) 597-6482

O: (954) 353-5599

DUNS: 080003744

CAGE: 7GS61

1. Recommend an approach VA should take towards delivering a high-quality mobile experience, with the reasoning for why this approach is recommended and an explanation of what alternative approaches were considered and why they were not recommended.

SurfBigData recommends that the VA build native Android and iOS apps, to fully utilize the modern mobile device capabilities. The current smartphone is a remarkable feat of engineering, its half a dozen gadgets packed into a single pocket sized machine. Many of the most amazing features are done with a wide range of sensors, such as accelerometer, gyroscope, magnetometer, GPS, barometer, proximity sensor, ambient light sensor, etc. Those sensors open a lot of opportunities to native apps to interact with app users.

Creating native apps from scratch may take more effort compared to directly using VA's current responsive webpages, but as we understand the VA's intention to build an API framework, native apps will focus on the front side, share those APIs, and in the long run the cost of the app compared to the whole eco system, will be remarkably small. It is a front end focusing on the best user experience and the best data collection sensors, providing reliable data to VA APIs.

Benefits of cross-platform mobile development:

- Cost-effectiveness. It enables investing just once and in a single team.
- One technology stack. Developers can use a single technology stack for a broad variety of engineering tasks.
- Reusable code. Up to 90 percent of codebase can be reused from one platform to the other, instead of designing the same functionality in another language.
- Easy maintainability. It's easier to maintain and deploy changes because there is no need to maintain applications on each platform separately.

We'll consider the basic selection criteria for Xamarin, React Native, and Ionic frameworks to help you make the right decision:

Xamarin is a Microsoft-supported framework for cross-platform mobile app development that uses C# and native libraries wrapped in the .NET layer. You can find more detailed information about Xamarin in our article The Good and The Bad of Xamarin Mobile Development.

Ionic – is the framework that aims at developing hybrid apps using HTML5 and AngularJS.

React Native is the framework that allows for building close-to-native mobile apps (not "HTML5 apps" or "hybrid apps") using JavaScript and React.JS.

Xamarin vs Ionic vs React Native Comparison				
	Xamarin		Ionic	React Native
Code	C#		HTML CSS Type Script JavaScript	JavaScript+Java Objective-C Swift
Compilation/Interpretation	iOS	AOT	JIT with WKWebView	Interpreter
	Android	JIT/AOT	JIT	JIT
UI Rendering	Native UI Controllers		HTML CSS	Native UI Controllers
GitHub Stars	3.6K		33.3K	59.6K
Cross-Platform Portability	Xamarin iOS/Android	Xamarin Forms	Up to 98% of source code reuse	Adapting code to each platform
	Business Logic/Data Access/Network Communication	Up to 96% of source code reuse		
Performance	Close to Native	Moderate-Low	Moderate-Low	Close to Native
Use Cases	All Apps	Simple Apps/Corporate Apps	Simple Apps/Corporate Apps	All Apps
Price	Open Source		Open Source	
Additional Costs	Visual Studio IDE $539-$2,999 for commercial		Ionic Pro $29-$199 for additional features	NA

Recommendation: **Xamarin** is a Microsoft-owned software company founded in May 2011, with modern **C#**-shared codebase. Developers can use Xamarin tools to write native Android, iOS, and Windows apps with native user interfaces and share code across multiple platforms, including Windows and macOS. As of last year, over **1.4 million developers** were using Xamarin's products in 120 countries around the world.

In comparison React Native requires two different sets of code, one for Android and iOS, essentially doubling the manpower required for development and maintenance. While Ionic Framework does allow code reuse across operating systems, it doesn't allow for fine tuning between them. This means small performance gap issues, which would normally not be an issue.

However, due the volume of intended use for the VA Mobile app, we foresee those small performance gap issues accumulating into larger problems down the road when more advance features are implemented.

2. Document which features of Vets.gov would be the best to implement in a native application, if any, and what the specific benefits are of having those features be native. This could include things like "fingerprint login" or ability to take pictures to submit them as evidence for a claim which could not be easily achieved using webviews or a mobile responsive website.

To document the features of the Vets.gov site that would be best to implement as native applications we would first ask all of the key stakeholders which features matter to them most. We would also include any future stakeholders not currently supported by the current system as well as capturing any interdependencies and/overlap. We would then have all stakeholders prioritize the features. The highest prioritized feature would then be selected for a feasibility study and proof-of-concept (POC). The highest prioritized feature that can deliver the greatest value having the largest positive impact to the mobile app as a whole would then be selected for a feasibility study and proof-of-concept (POC).

For example, if identification authentication is selected as a top feature we would build out a POC on the AWS GovCloud to experiment with various types of authentication (e.g. facial recognition, fingerprint login, etc.). It is best to build these features as Cloud native applications because they can continuously be improved in a timely manner following continuous improvement/continuous deployment(CI/CD) by any developer on the team following the development logs and DevOps process.

We would then be able to have the team(s) test, develop, integrate and perform a system demo for all stakeholders after each sprint of 2 week intervals. When the set of features delivered meet with stakeholders definition of done then this POC with be used to define the scope, vision and roadmap for the most viable product (MVP). The development of new features would all take place in a test environment in parallel to the current production environment so there will be no disruption to the current production environment.

From a technical standpoint SurfBigData recommends Xamarin to build VA mobile app. Xamarin is a comprehensive mobile app development platform supporting the full mobile development life cycle, including development, testing and monitoring. Xamarin uses C#-share codebase, so developers can use this tool to write native Android, iOS, and Windows apps with native user interfaces and share code across multiple platforms, including Windows and macOS, with 100% of the native APIs exposed, giving developers full access to device capabilities.

All smartphones have built-in web browsers, native apps also have the capability to embed a browser to present web contents or use JavaScript to access limited phone resources. But we do not recommend for the VA to use webviews as the primary function delivery channel in the app. Compared to native functions, webviews are less secure and lower performance. An app with primarily webviews is "bolted on" rather than built in, it is less consistent and less impressive. Organizations have focused on mobile as a key part of their digital transformation strategies. Choose tools that specifically meet the portfolio of mobile apps by prioritizing and planning for the use cases that apps will support. Focus on what's needed now and keep an eye on future interactions and multiple experiences needed. Create a layered approach to providing multiple experience solutions by choosing tools that offer flexibility and scalability.

We recommend developing the most important or most interesting/ attractive features in the first release, even if it has only a single, impressive feature to help VA users. Carefully planning for features in each release, using agile software development methodology, continuous testing and delivery based on planning and feedback is the roadmap to a successful app that benefits its users.

3. Create a roadmap for both an MVP native application as well as the next highest priority items after an MVP. What would need to be included in an MVP to have an acceptable product for VA to go to market with?

We recommend the Scaled Agile Framework (SAFe) to establish the roadmap for MVP native application and agile mechanism to continuously identify and implement the highest priority items. The DevOps/Agile mindset suggests we should not define an MVP upfront, but we should engage the stakeholders at regular intervals to define the roadmap to continuous deployment success.

Users and customers' feedback is key to evolving the MVP into the final product. To align closely to veterans and stakeholders' value, the team must include continuous customer feedback loops when working on an MVP. The entire team should be ready with an Agile approach, a discovery mindset, and regular feedback from the stakeholders. By remaining customer-centered throughout the effort, we are most likely to have an MVP that is considered valuable to the veterans and stakeholders. SAFe for VA is as Figure 1.

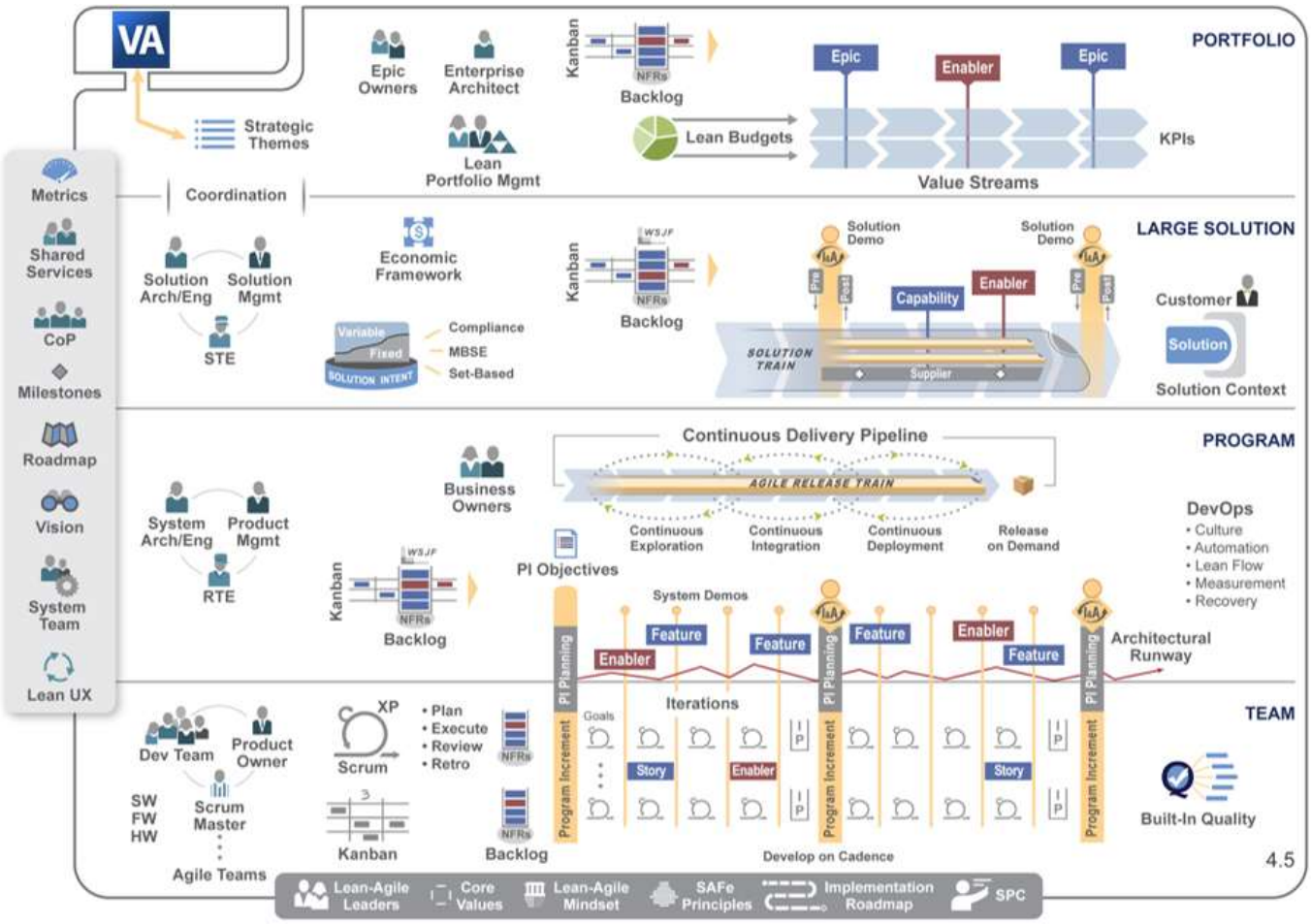

Figure 1. Full SAFe for VA Flagship Mobile Application

Portfolio Level:

The Portfolio Level contains the principles, practices, and roles needed to initiate and govern a set of development Value Streams. This is where strategy and investment funding are defined for value streams and their Solutions. This level also provides Agile portfolio operations and Lean governance for the people and resources needed to deliver solutions.

The portfolio level aligns enterprise strategy to portfolio execution by organizing the Lean-Agile Enterprise around the flow of value through one or more value streams. Delivering the basic budgeting and necessary governance mechanisms, it assures that investment in solutions will provide the Return on investment (ROI) the enterprise needs to meet its strategic objectives. In VA, there may be multiple SAFe portfolios.

Large Solution Level:

The Large Solution Level contains the roles, artifacts, and processes needed to build large and complex solutions. This includes a stronger focus on capturing requirements in Solution

Intent, the coordination of multiple Agile Release Trains (ARTs) and Suppliers, and the need to ensure compliance with regulations and standards.

When building large and complex systems, theoretically, the simplest thing that could possibly worked would be a single team. But we know that even teams with more than 11 people are usually problematic. And a single 'team' of hundreds, or even thousands, of people just isn't feasible. Instead, we need teams-of-teams (ARTs) and teams-of-teams-of-teams (Solution Train). Coordinating these activities requires additional roles, events, and artifacts, which is the purpose of the large solution level.

Program Level

The Program Level contains the roles and activities needed to continuously deliver solutions via an Agile Release Train (ART). The program level is where development teams, stakeholders, and other resources are devoted to some important, ongoing solution development mission. The ART metaphor describes the program level teams, roles, and activities that incrementally deliver a continuous flow of value. ARTs are virtual organizations formed to span functional boundaries, eliminate unnecessary handoffs and steps, and accelerate value delivery by implementing SAFe Lean-Agile principles and practices.

Although it is called the program level, ARTs are long-lived and, therefore, have a more persistent self-organization, structure, and mission than a traditional program. Usually, a program has a definitive start and an end date, as well as temporarily assigned resources

Team Level:

The Team Level contains the roles, activities, events, and processes which Agile Teams build and deliver value in the context of the Agile Release Train (ART).
While depicted somewhat separately in the "Big Picture", the SAFe Team Level is a vital part of the Program Level. All SAFe teams are part of an ART-the primary construct of the program level.

The Roadmap for the MVP native application and the highest priority items after MVP follows Pipeline in the Agile framework.

3.1 Pre-MVP Business Understanding and Researching

Often as business analysts we are expected to dive in to a project and start contributing as quickly as possible to making a positive impact. Sometimes the project is already underway, other times the project is at its inception. We face a lot of ambiguity as business analysts and it's our job to clarify the scope, requirements, and business objectives as quickly as possible.

We take the required time, whether that's a few hours or at the very most a few weeks, to get oriented to the customers goals. This ensures we are not only moving quickly, but also able to be an effective and confident contributor on the project.

Our key responsibilities in this step include:

- Clarifying our role as the business analyst so that we are sure to create deliverables that meet VA stakeholders' needs.

- Determining the primary stakeholders to engage in defining the project's business objectives and scope, as well as any subject matter experts to be consulted early in the project.

- Understanding the project history so that we don't inadvertently repeat work that's already been done or rehash previously made decisions.

- Understanding the existing systems and business processes so we have a clear picture of the current state of the effort. By researching the most visited functionalities and the services being requested, it will give a picture of priorities in terms of statistics. This is where our team will learn how to learn what we don't know. This step gives us the information customers need to be successful and effective in the context of VA's Mobile Application project.

- Discovering expectations from your primary stakeholders – essentially discovering the "why" behind the project. For VA Flagship Mobile App, reasons of mobility service access are different. For veteran use case, Health Care Benefits require timelier and more frequent access than Education Benefits and Housing Assistance.

- Reconciling conflicting expectations so that the business community begins the project with a shared understanding of the business objectives and are not unique to one person's perspective. Ensuring the business objectives are clear and actionable to provide the project team with momentum and context while defining scope and the detailed requirements.

- Discovering the primary business objectives sets the stage for defining scope, ensuring that you don't end up with a solution that solves the wrong problem or, even worse, with a solution that no one can even determine is successful or not.

3.2 SAFe Managed-Investment Pre-Commitment Phase

Pre-commitment Prior to engaging in any significant investment contract for developing a complex system with many unknowns, due diligence is required. The customer (VA)

and supplier work together to come to terms for the basis of the contract. This is the pre-commitment phase, illustrated in Figure 2.

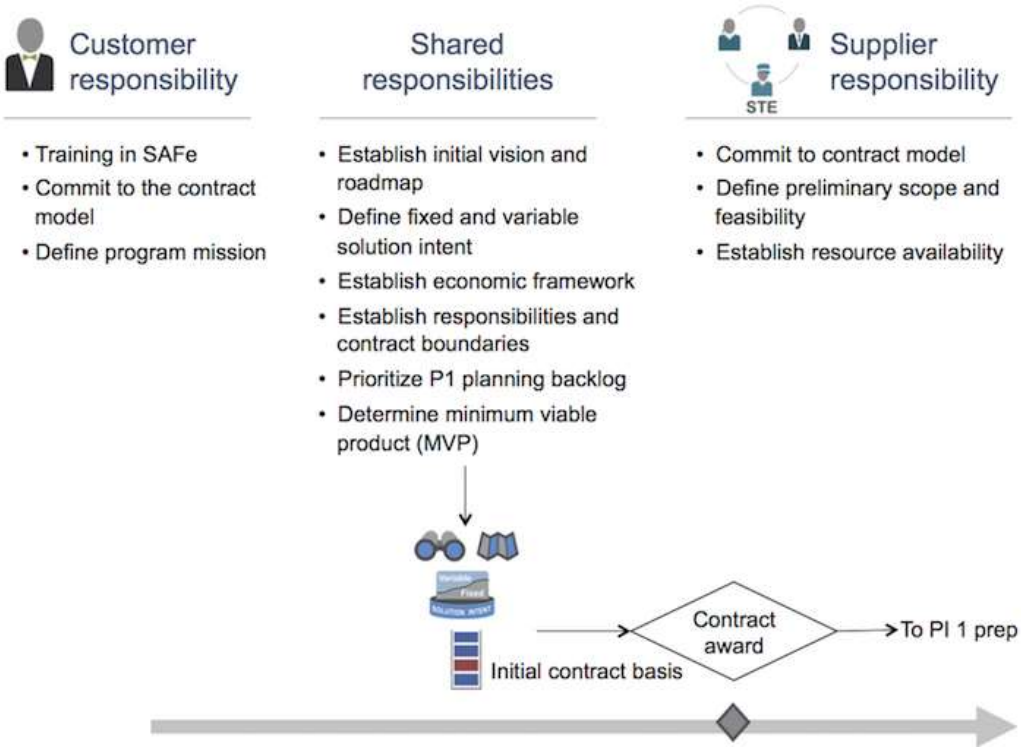

Figure 2 Safe Pre-commitment Phase

During pre-commitment, the customer has specific responsibilities, including understanding the basic constructs and obligations of this form of Agile contract, and defining and communicating the larger program mission statement to the potential supplier(s).

We the supplier/contractor do our homework as well. This often includes the first analysis of potential feasibility and alignment of the buyer's solution priorities. It also demands some understanding of the potential resources that will be required over the initial contract periods and a rough cost estimate.

The shared responsibilities, illustrated in Figure 2, start the customer and supplier/contractor toward a more measured investment, supported by continuous objective evidence of fitness for use. These responsibilities include:

- Establishing the initial Vision and Roadmap
- Identifying the Minimum Viable Product (MVP) and additional Program Increment (PI) potential Features
- Defining the initial fixed and variable Solution Intent
- Prioritizing the initial Program Backlog for PI Planning
- Establishing execution responsibilities
- Establishing the Economic Framework, including economic trade-off parameters,

3.3 Feasibility Study: MVP-POC based on architecture identified during the feasibility study

The Lean Startup Cycle shown in Figure 3 is also used to illustrate how major product development investments are managed with sound economics.

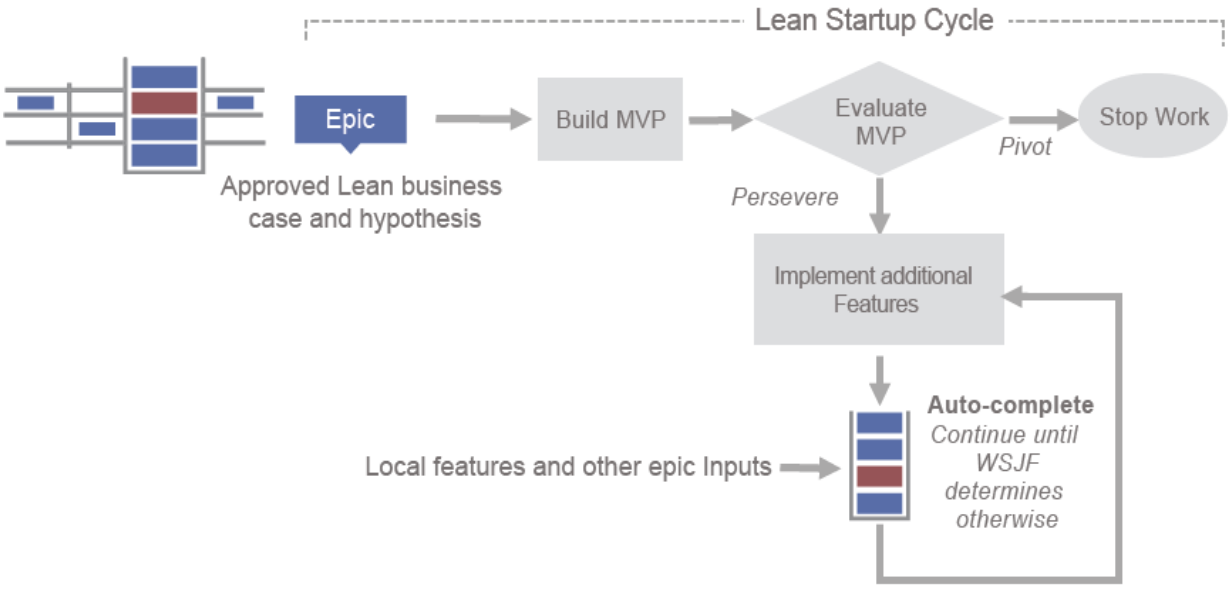

Figure 3. Iterative and incremental approach for Epics in the Lean Startup Cycle

As Figure 3 implies, approved Epics deserve additional investment—but not a fully committed investment up-front. After all, the work up to that point has been analytical and exploratory. The proof is in the pudding, and for that, we need to apply the hypothesize-build-measure-learn cycle, including the steps below:

- **Hypothesize** – Each epic has a Lean business case, which includes the hypothesis that describes the assumptions and potential measures that can be used to assess whether an epic will deliver business value commensurate with the investment.

- **Build an MVP** – Based on the hypothesis of the epic, the next step is to implement a Minimum Viable Product (MVP)—the minimum effort necessary to sufficiently validate or invalidate the hypothesis. In SAFe, this translates to the minimum Feature set required to deliver some holistic, but minimal, solution.

- **Evaluate the MVP** – Once the feature set is implemented, teams evaluate the MVP against the hypothesis. However, this evaluation is not based on Return on Investment (ROI), as that's a trailing economic indicator. Instead, teams apply 'innovation accounting,' and design the systems to provide fast feedback on the leading indicators of future success. These might include usage statistics, system performance, or anything else that would be a useful metric.

- **Pivot or persevere** – With the objective evidence in hand, teams and stakeholders can decide: pivot—stop doing that work and start doing something else; persevere—define features to further develop and refine the innovation.

- **Implement additional features** – Choosing to persevere means work continues until new epic-inspired features that hit the backlog can't compete with other features. This will occur naturally via 'Weighted Shortest Job First'(WSJF). WSJF also has the unique advantage of ignoring sunk costs on the epic to date, as the job size includes only the work remaining.

This Lean Startup model decreases time-to-market and helps prevent the system from becoming bloated with unnecessary features that may never be used. It also enforces the 'hypothesize-build-measure-learn' cycle. The implication is that Agile contract language is modified to reflect a combination of fixed and variable components. The MVP identified at the pre-commitment stage can establish a high-level definition of fixed scope to be delivered over a proposed number of PIs.

Beyond the delivery of the MVP, the contract can also specify the number of option periods consisting of one or more PIs--. The goal is to optimize the delivery of prioritized features within each PI. This process continues until the solution has delivered the value the customer requires. This provides the best of both worlds to customers: Better predictability of estimates associated with a far smaller MVP than the full list of all requirements, and total control over the spend required for additional incremental feature based on economic outcomes.

- Opportunity overview
  - Problem description and expected outcomes from the end customer perspective
  - Estimated workloads (number of applications and servers, server types (Windows, Linux)) to be discovered and migrated

    - TCO estimates (AWS Annual Run Rate post migration-assumed AWS is VA Mobile App cloud service provider)
- Scope and assumptions of the project
- Project phases and list of work items and deliverables in each phase – see *Technical Project Plan template* below
- Summary of milestones and deliverables, which are verifiable and auditable in the future
- Financial information:
    - Total project cost
    - Estimated contributions by Partner, Customer, and AWS respectively

Contract Execution: After the contract is awarded, development begins, as is illustrated in Figure 4.

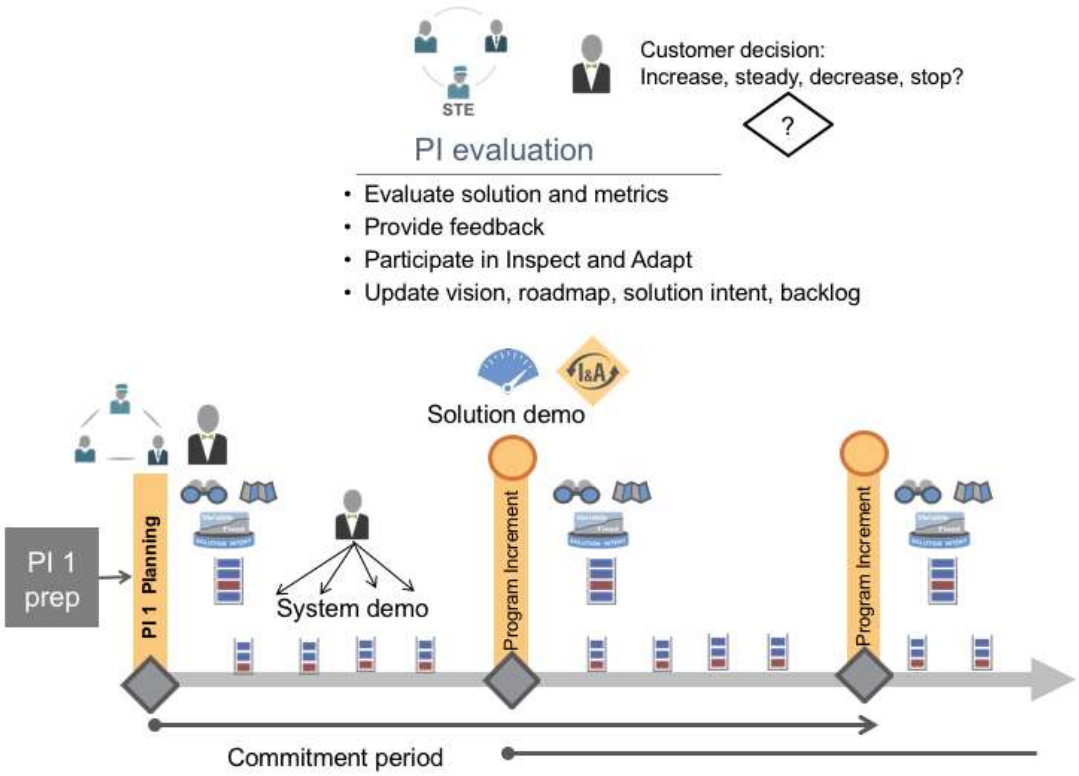

Figure 4. SAFe managed-investment contract execution phase

A description of the activity timeline follows:

- PI preparation – Both supplier and customer will invest some time and effort in preparing content and logistics for the first PI planning session.

- PI planning session. (Note: In some cases, it might be suitable that the first PI planning is part of the pre-commitment phase, this route requires more investment by both parties.) PI planning – The first PI planning event influences the entire program. There, customer and supplier stakeholders plan the first PI in iteration-level detail.

- PI execution – Depending on context, customers participate at various levels in iteration execution. At a minimum, however, direct customer engagement is usually required for each System Demo. For large solutions, however, the multiplicity of system demos may be replaced by a more fully integrated Solution Demo, which can occur more frequently than at PI boundaries.

- PI evaluation – Thereafter, each PI marks a critical Milestone for both the customer and supplier. At each milestone, the solution demo is held, and the solution is evaluated. Agreed-to metrics are compiled and analyzed, and decisions are made for the next PI. Solution progress and program improvements are assessed during the Inspect and Adapt (I&A) event. Thereafter, the next PI planning commences, with the scope based on the outcome of that decision.

Flagship Mobile Application

MVP Template

This minimum viable product (MVP) template will explain the steps involved in determining what a viable version one of the VA mobile product entails. Each step will be explained in detail; at the end of the steps, we have included an editable Template that will allow teams to create the VA's own plan for the VA Mobile Application MVP.

A MVP is a version of VA product which includes the features that will allow VA to release the product to market by solving a core problem for a set of users. The purpose is to provide immediate value, quickly, while minimizing development costs.

This walkthrough and template will provide management with the guidance VA needs in order to build VA Mobile Application MVP.
We have grouped the process to accomplish this into simple steps:

1. Understand the Business Need
2. Find the Solutions
3. Decide What Features to Build (Prioritization Matrix)

All of these steps should be part of product definition for any project, however following these steps in a manner that allows VA to confidently outline what to ship a valuable version one of VA Mobile Product.

1. Understand VA Business Needs

Determine the long-term goal and write it down. The goal should be driven by a problem needed to be solved. We want to answer the simple question: Why are we doing this project?

Long-Term Goal:

User Personas: Who are we building this product for? Define VA end users by creating user personas that outline the needs, motivations, and pain points of people who will use VA Mobile Application.

Success Criteria: We want to identify the success criteria that will demonstrate whether or not the product will be successful. Note that the success criteria can and usually will be more than a single metric.

2. Find the Solutions

The purpose of this exercise is to identify the opportunities and determine how VA can most effectively add value and solve pain points. Here is the step-by-step process:

A. Map out the user journeys:
   1) Map out the users (write down on the left)
   2) Write down the story ending on the right (what we need the user to do to meet the goal)
   3) Write down all actions (jobs) in between.

In the majority of cases, we want to look at which user has the most jobs and focus on that user (Note that this approach works from a logical perspective, but there are sometimes higher priorities which may need to be addressed).

User	**Action**	**Story Ending**

B. Create a pain and gain map for each action
   1) Write down the action (job) the user completes when using the product
   2) Write down the pain points for each job
   3) Write down the gains for each job

List and count the number of pains and gains for each action. Ideally, when it makes sense, you want to assign a value to help signify importance (for example, if a gain reduces a financial cost to your business then it is worth 3 points, whereas a smaller gain is worth 1 point). This exercise lets you determine where you have the greatest potential to resolve pains and add gains; focus on building features that address that area for your MVP (other areas can be added as items for later in the product roadmap).

Pains	Actions	Gains
Trouble enrolling health benefit program	Automating eligibility, document proof, ID verification (Facial Recognition/Finger Print/Voice Recognition/Military ID)	Accessing health benefit programs, and eligibility to given benefit program immediately approved or informed with specific timeline.

C. Summarize the pains and gains into opportunity statements
   There are a number of methods to summarize pains and gains, but we like to use opportunity statements that follow a "How Might We" format.
   For Example:
   • How Might We expedite the application process?
   • How Might We help users find things more easily?
   • How Might We make it easier to upload ID and EHR documents securely?
   Opportunity statements will reflect the pains and gains you have identified for your product. Opportunity Statements
   How Might We _____?

3. Decide What Features to Build

Using your opportunity statements, finalize what features you want to build out. In this stage, we would create the feature sentence, for example: “How Might We expedite the application process?” would become "Reduce your application processing time by 35%.”

List the user and the specific opportunity statements, and provide a breakdown of the features to include in the product roadmap. Next, you want to prioritize features. This step helps you to identify where you can make the most impact with your product in relation to the urgency of the feature. For this, we recommend using a prioritization matrix. From the prioritization matrix, you make the final decision on what absolutely needs to be included in your MVP.

Prioritization Matrix