Philips Industry Consulting

Where consulting gets hands on

Agile and software architecture

Anton Jansen & Klaas Wijbrans

Philips Industry Consulting

7th October, 2015

innovation + you

PHILIPS

About Anton Jansen

- A system/software architecture consultant at Philips Innovation Service Industry Consulting.
- Lived and worked for 6 years in Sweden working for ABB Corporate Research as a senior scientist software architecture.
- Holds a PhD in software engineering from the University of Groningen on the topic of architectural design decisions
- Written over 20 peer reviewed articles on the topic of architecture.
- Wrote the most influential paper on software architecture in the last 10 years.

PHILIPS

About Klaas Wijbrans

- Senior consultant architecture at Philips Innovation Service Industry Consulting.
- Master's degree in Electrical Engineering and Doctorate in the Technical Sciences from the University of Twente
- > 25 years experience in systems architecture, process improvement and technology management

PHILIPS

Agenda

Philips Industry Consulting

Where consulting gets hands on

- Context
- Agile Software development
- Software Architecture & Agile
- Scaled Agile Framework (SAFe)
- SAFe Challenges
- Open questions
- Recommendations / take aways
- Questions?

PHILIPS

Context

Manifesto for Agile Software Development (2001)

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

Fundamental underpinnings

- Software is unique in several ways;
  - Production/manufacturing is trivial, i.e. copy!
  - Carefully following a process has does not guarantee the quality of the resulting software.

  => A purely process oriented organization cannot create/manage software

- Making software, including coding, is a collaborative **design** activity

⇒ Maximize communication bandwith among relevant actors

- Nobody knows the real requirements, not the developers, not product management, not the customer.
  - "*It's really hard to design products by focus groups. A lot of times, people don't know what they want until you show it to them*" – Steve Jobs

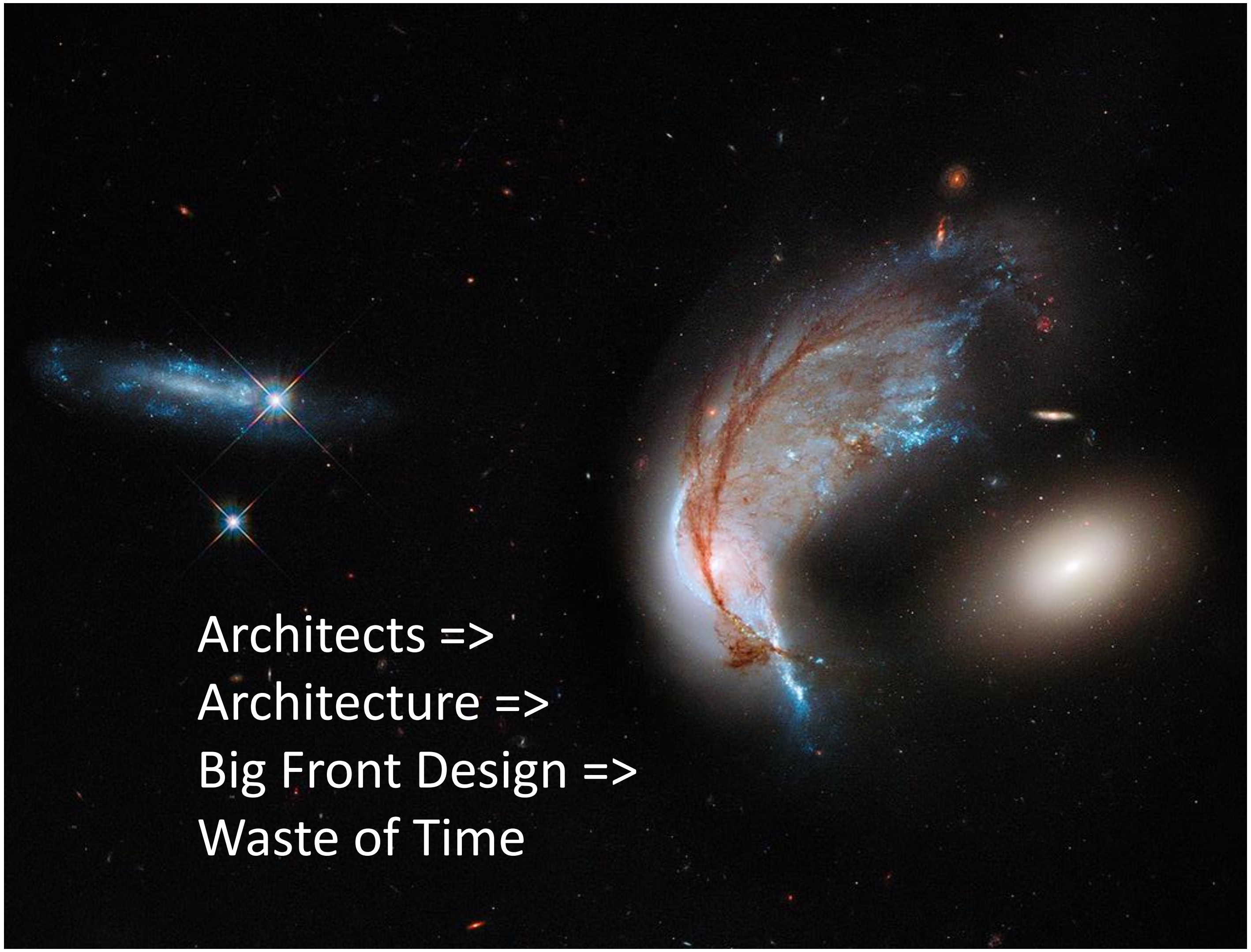
⇒Get customer feedback often

→Release often!

→Can you release your software now?

=> Making software is a journey of exploration!

Software Architecture & Agile



Do you need an architect to create software?

It depends…..

- How complex is your system and it’s surrounding?
- How big is the part that you need to develop?
  - Can you re-use an architecture, e.g. a technology stack, and just add the “small” missing parts?
  - How many people would you need to develop the software on time?
- What is the cost to society if things go wrong?

=> Most software does not need an architect

⇒ Most software is written by people who never received formal education for it

Software Architecture

Software Architecture = {Architectural design decisions}

Architectural design decision = Those decisions that are costly to change later

When traditional agile approaches hit the wall

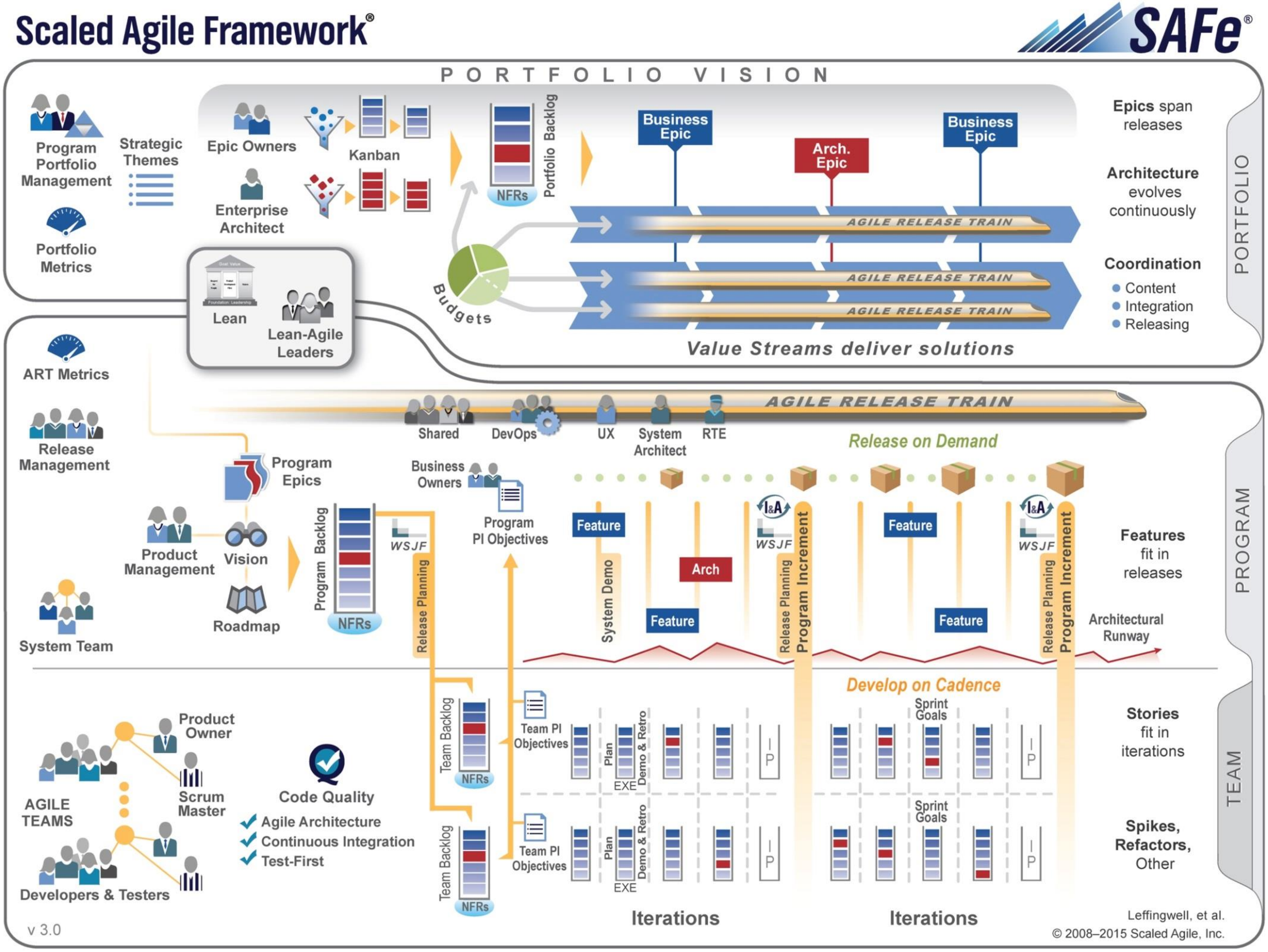
People’s communication bandwith does not scale!

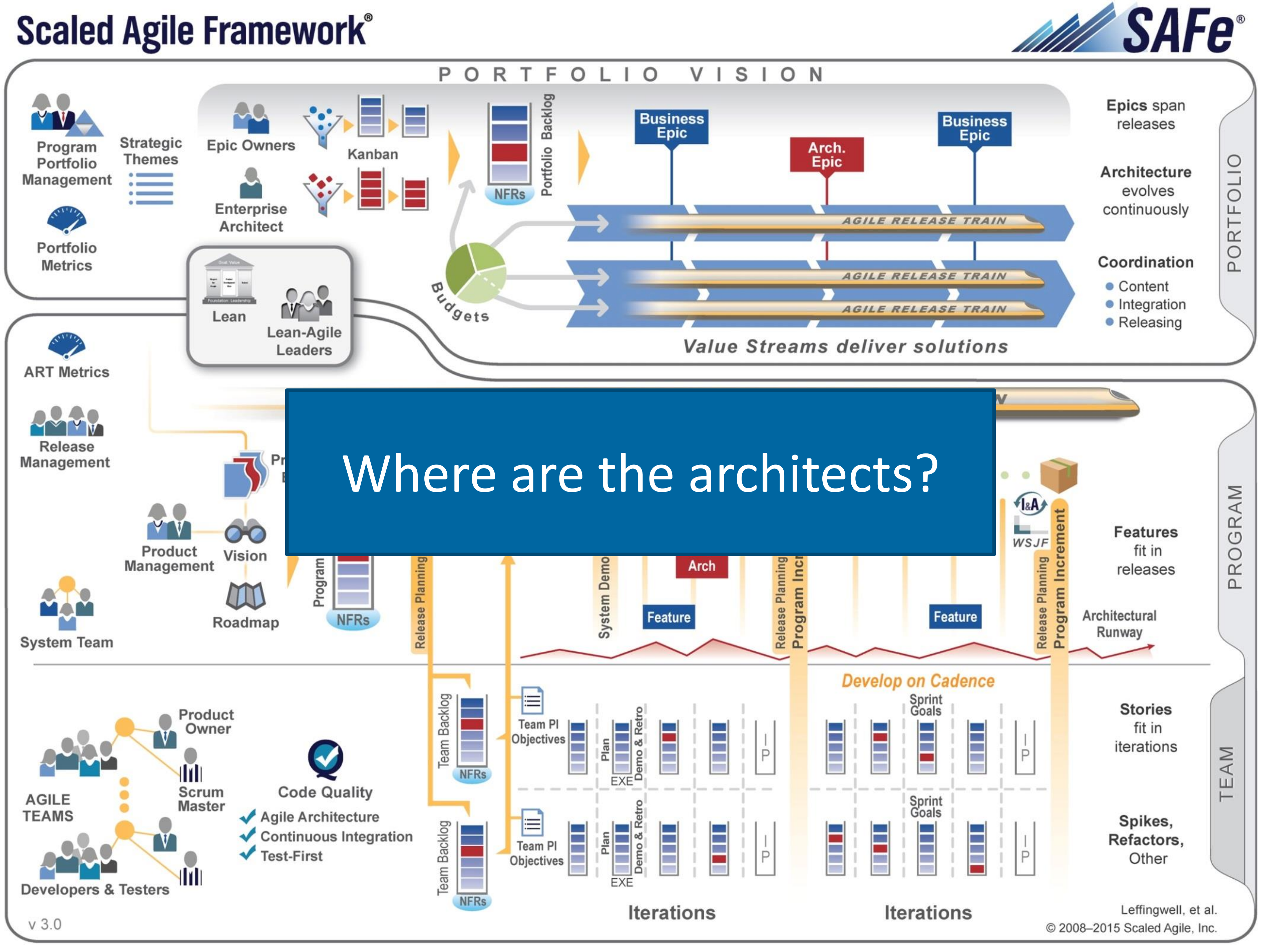
Scaled Agile Framework (SAFe)

- Background
  - Created by Dean Leffingwell (Rational Unified Process (RUP), ClearQuest, RequisitePro)
  - One of several attempts (e.g. DAD, LeSS, Nexus) to scale up agile methods.
  - Combines ideas from lean and agile
- Philosophy
  - Push decision making down as much as possible
  - Steer on scope, budget/time and quality fixed.
  - Develop on a fixed cadence
  - Sharing knowledge is primarily done through socialization

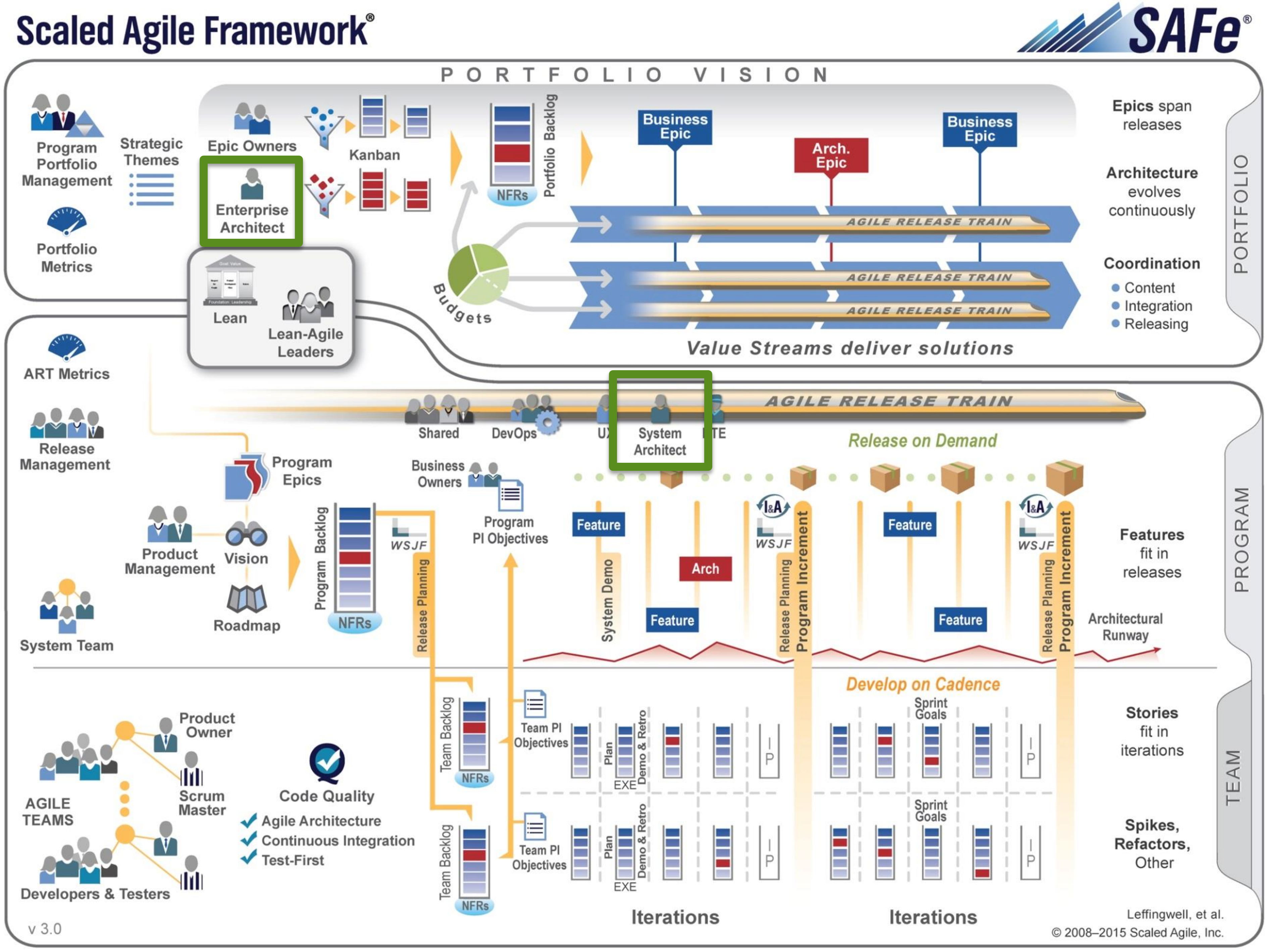
$\Rightarrow$No more project leaders
$\Rightarrow$Requires deep and full engagement of product management
$\Rightarrow$Management has to give up on large parts of (illusionary) control

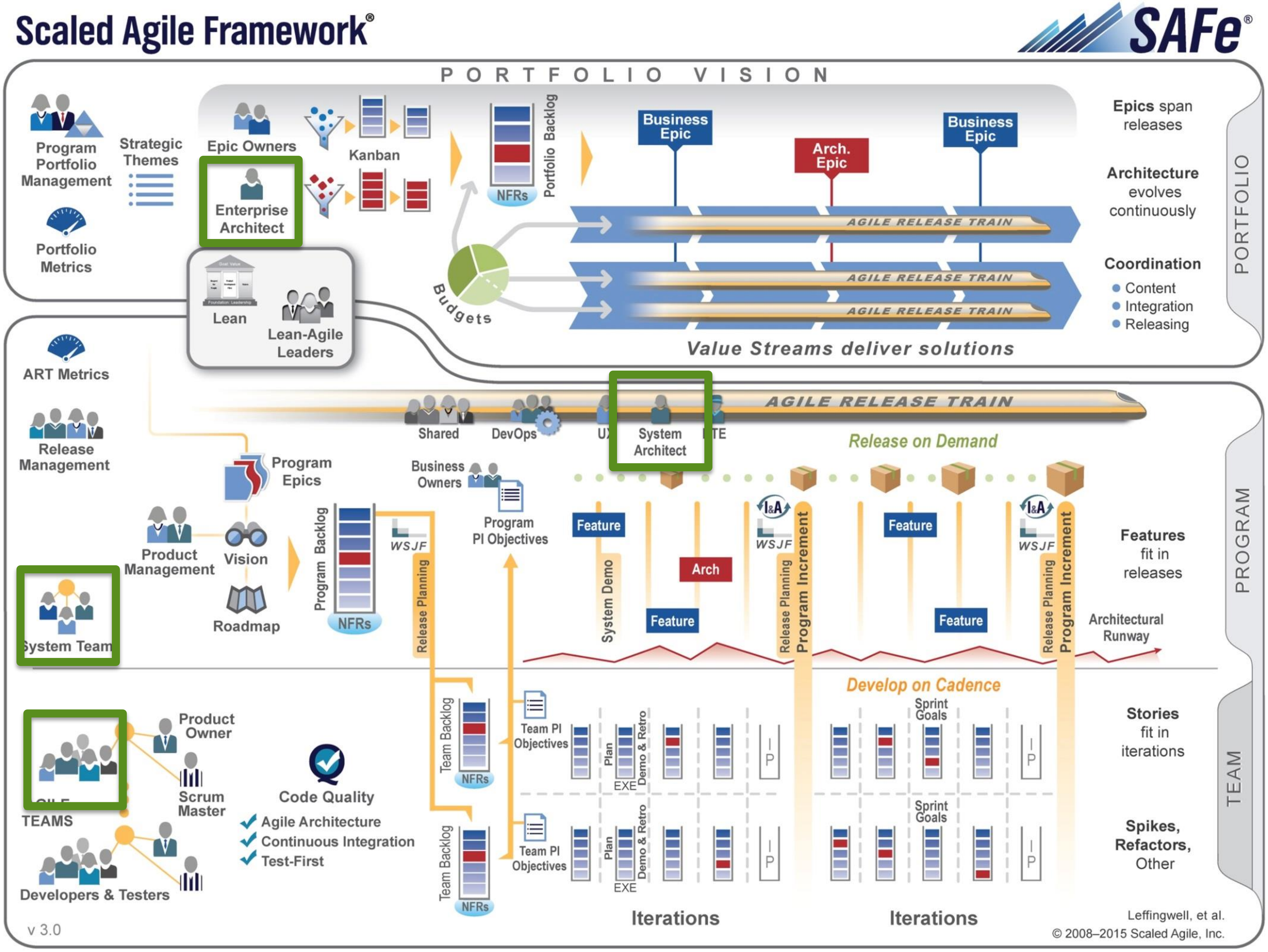
Scaled Agile Framework®



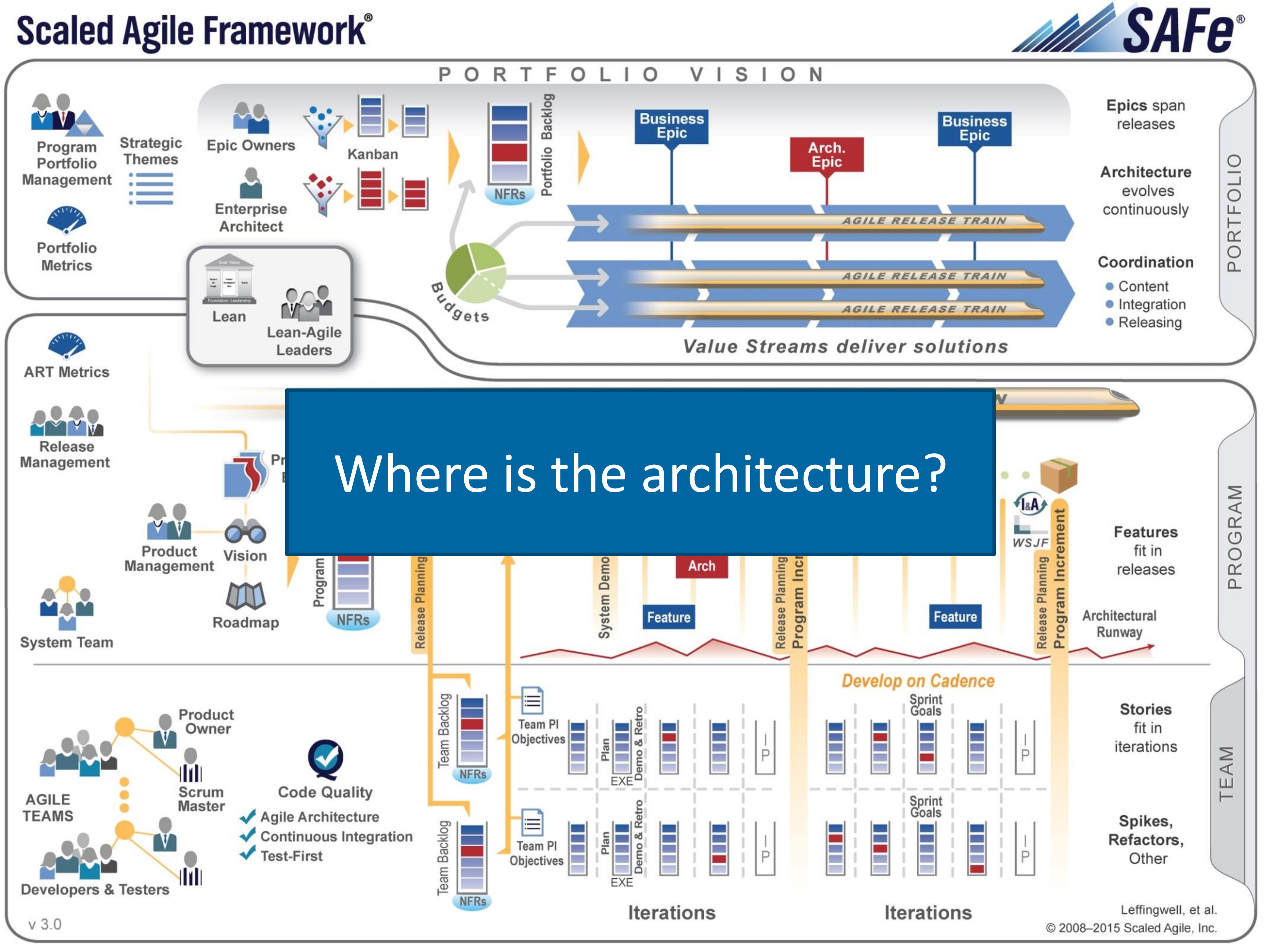


Where are the architects?

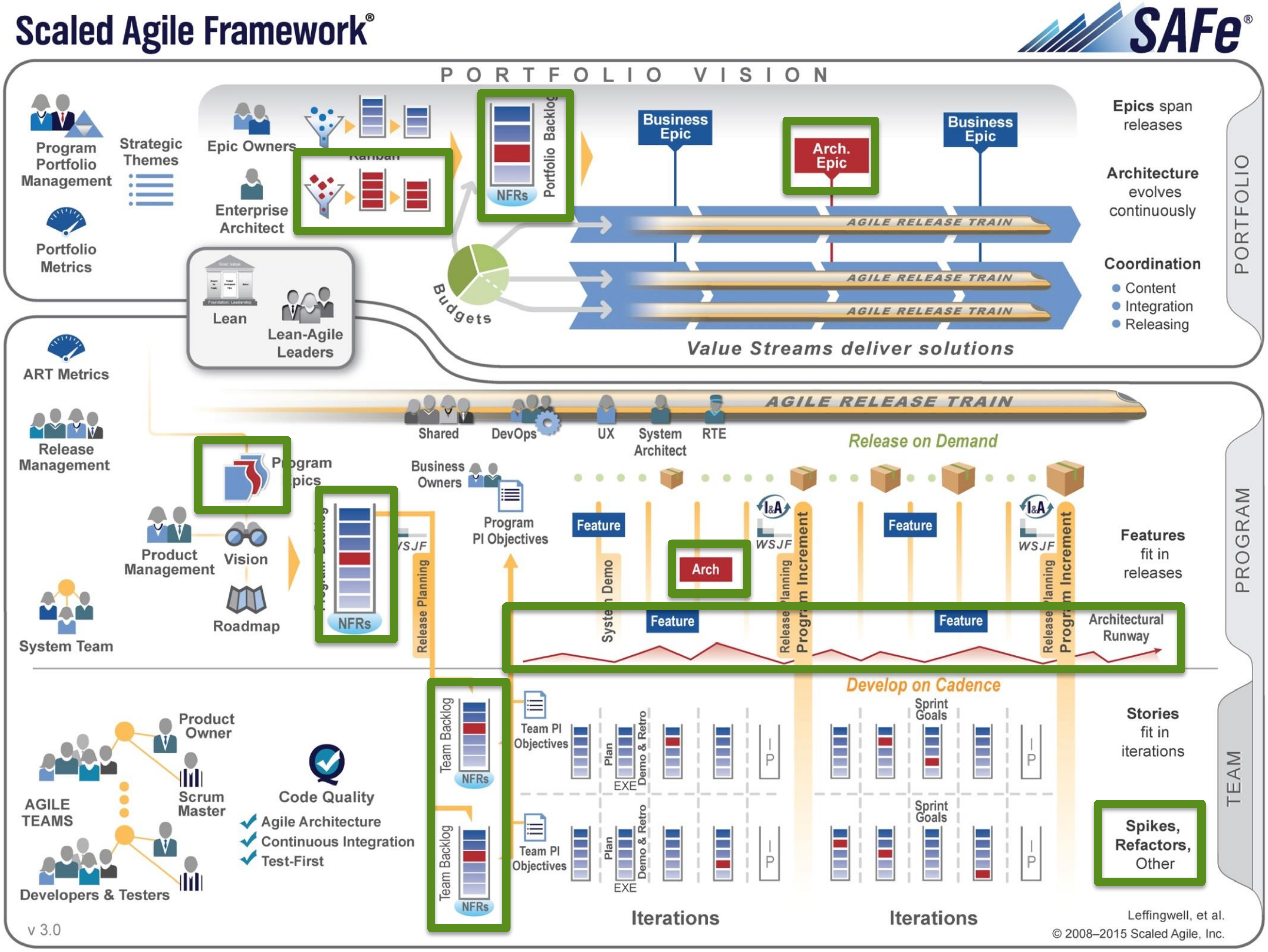

Scaled Agile Framework®
SAFe®
PORTFOLIO VISION
Program Portfolio Management
Strategic Themes
Epic Owners
Kanban
Enterprise Architect
Portfolio Metrics
NFRs
Portfolio Backlog
Business Epic
Arch. Epic
Business Epic
AGILE RELEASE TRAIN
AGILE RELEASE TRAIN
AGILE RELEASE TRAIN
Budgets
Lean
Lean-Agile Leaders
Value Streams deliver solutions
Epics span releases
Architecture evolves continuously
Coordination
Content
Integration
Releasing
PORTFOLIO
ART Metrics
Release Management
System Team
AGILE RELEASE TRAIN
Shared
DevOps
UX
System Architect
RTE
Release on Demand
Program Epics
Product Management
Vision
Roadmap
Program Backlog
NFRs
WSJF
Release Planning
Business Owners
Program PI Objectives
Feature
Arch
Feature
Feature
Feature
System Demo
I&A
WSJF
Release Planning
Program Increment
I&A
WSJF
Release Planning
Program Increment
Features fit in releases
Architectural Runway
PROGRAM
Develop on Cadence
Product Owner
Scrum Master
AGILE TEAMS
Developers & Testers
Code Quality
Agile Architecture
Continuous Integration
Test-First
Team Backlog
NFRs
Team PI Objectives
Plan
EXE
Demo & Retro
IP
Sprint Goals
Team Backlog
NFRs
Team PI Objectives
Plan
EXE
Demo & Retro
IP
Sprint Goals
Iterations
Iterations
Stories fit in iterations
Spikes, Refactors, Other
TEAM
v 3.0
Leffingwell, et al.
© 2008–2015 Scaled Agile, Inc.



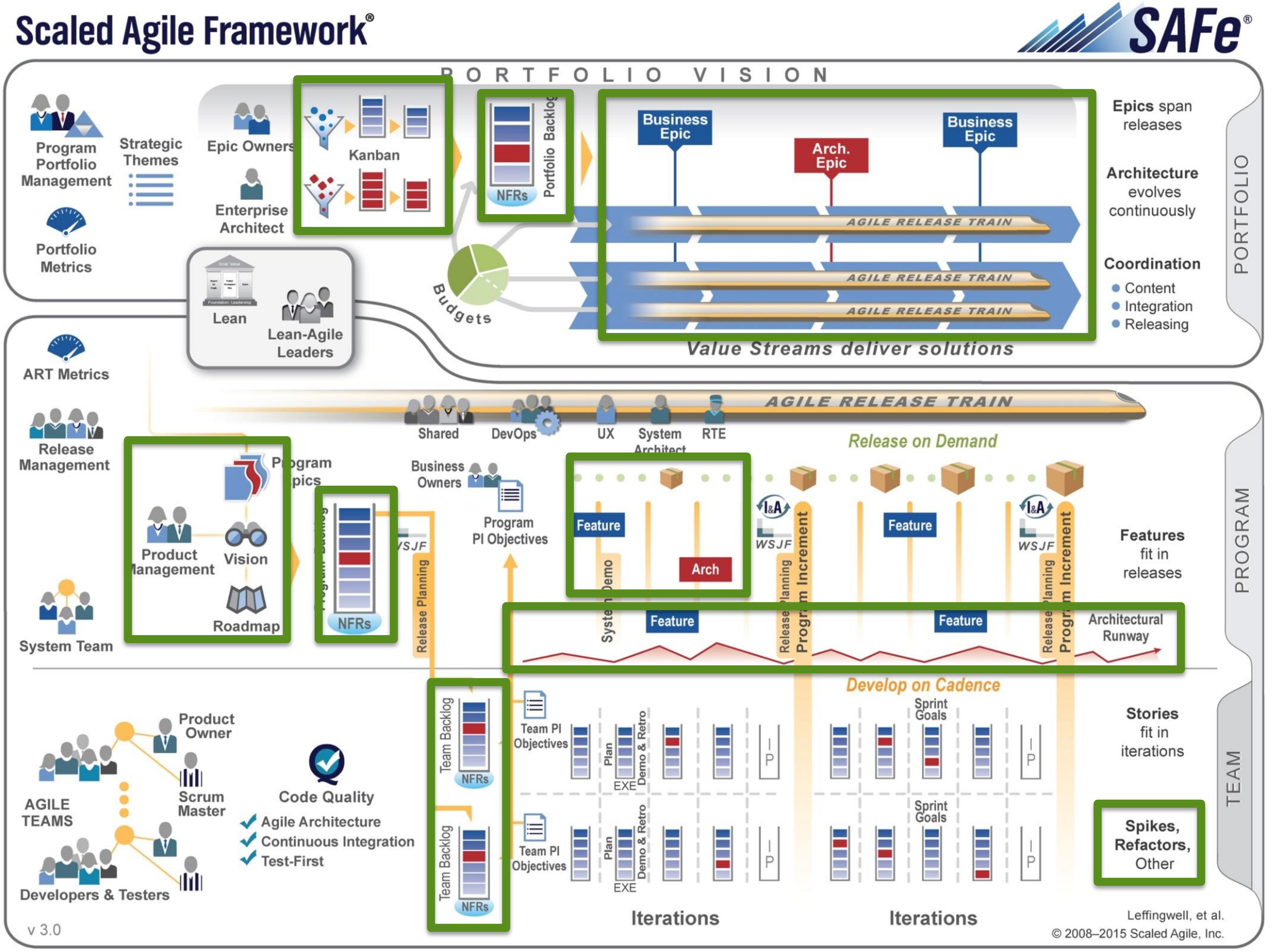
Scaled Agile Framework®
SAFe®
PORTFOLIO VISION
Program Portfolio Management
Strategic Themes
Epic Owners
Enterprise Architect
Kanban
Portfolio Backlog
NFRs
Business Epic
Arch. Epic
Business Epic
Portfolio Metrics
Lean
Lean-Agile Leaders
Budgets
AGILE RELEASE TRAIN
AGILE RELEASE TRAIN
AGILE RELEASE TRAIN
Value Streams deliver solutions
Epics span releases
Architecture evolves continuously
Coordination
Content
Integration
Releasing
PORTFOLIO
ART Metrics
Release Management
AGILE RELEASE TRAIN
Shared
DevOps
UX
System Architect
RTE
Release on Demand
Program Epics
Business Owners
Program PI Objectives
Product Management
Vision
Roadmap
Program Backlog
NFRs
WSJF
Release Planning
Feature
System Demo
Feature
Arch
I&A
WSJF
Release Planning
Program Increment
Feature
Feature
I&A
WSJF
Release Planning
Program Increment
Features fit in releases
Architectural Runway
PROGRAM
System Team
Develop on Cadence
Agile TEAMS
Product Owner
Scrum Master
Code Quality
Agile Architecture
Continuous Integration
Test-First
Developers & Testers
Team Backlog
NFRs
Team PI Objectives
Plan
EXE
Demo & Retro
IP
Sprint Goals
Team Backlog
NFRs
Team PI Objectives
Plan
EXE
Demo & Retro
IP
Sprint Goals
Stories fit in iterations
Spikes, Refactors, Other
TEAM
Iterations
Iterations
v 3.0
Leffingwell, et al.
© 2008–2015 Scaled Agile, Inc.



Where is the architecture?


Scaled Agile Framework®
SAFe®
PORTFOLIO VISION
Program Portfolio Management
Strategic Themes
Epic Owners
Enterprise Architect
Kanban
NFRs
Portfolio Backlog
Business Epic
Arch. Epic
Business Epic
Epics span releases
Architecture evolves continuously
Coordination
Content
Integration
Releasing
PORTFOLIO
Portfolio Metrics
Lean
Lean-Agile Leaders
Budgets
AGILE RELEASE TRAIN
AGILE RELEASE TRAIN
AGILE RELEASE TRAIN
Value Streams deliver solutions
ART Metrics
AGILE RELEASE TRAIN
Shared
DevOps
UX
System Architect
RTE
Release on Demand
Release Management
Program Epics
Business Owners
Program PI Objectives
Product Management
Vision
Roadmap
Program Backlog
NFRs
WSJF
Release Planning
Feature
System Demo
Arch
I&A
WSJF
Release Planning
Program Increment
Feature
Feature
Feature
Architectural Runway
Features fit in releases
PROGRAM
System Team
Develop on Cadence
Team Backlog
NFRs
Team PI Objectives
Plan
EXE
Demo & Retro
IP
Sprint Goals
Stories fit in iterations
TEAM
Product Owner
Scrum Master
AGILE TEAMS
Developers & Testers
Code Quality
Agile Architecture
Continuous Integration
Test-First
Team Backlog
NFRs
Team PI Objectives
Spikes, Refactors, Other
Iterations
Iterations
v 3.0
Leffingwell, et al.
© 2008–2015 Scaled Agile, Inc.

Scaled Agile Framework®



Leffingwell, et al.

SAFe Challenges (1/2)

- Assumes a brown-field situation
  - Misses the ramp-up curve needed for green field

- The enterprise architect has it’s own Kanban, with budget
  - Drive architectural run-way
  - Manage technical debt

- NFRs are seen as constraints on the contents of the different backlogs
  - Architecture trade-offs left implicit

- Architecture is incrementally and piecemeal defined.
  - Epic/Features/User stories contain delta’s to architecture and requirements
  - Architecture Exploration performed through “spikes”, which are triggered at portfolio level

SAFe Challenges (2/2)

- Base for delta is implicit!
  - Consolidation of delta’s not defined
  - SAFe tool support only focusses on delta’s
  - Consolidation can be hooked to definition of “done”

- Implicitly assumes agile teams are interchangeable
  - Deep domain knowledge ignored
  - Teams can be aligned with components ➔ allocation of domain knowledge to the right components becomes key!

- Implicit Transformation of Epics to Program Epics
  - Need solid architecture foundation to spread work among the ARTs

- SAFe is more a framework than a methodology -> consultancy friendly ☺

Open Questions

- Would an approach like SAFe also work for software intensive / hardware based products/systems/services?

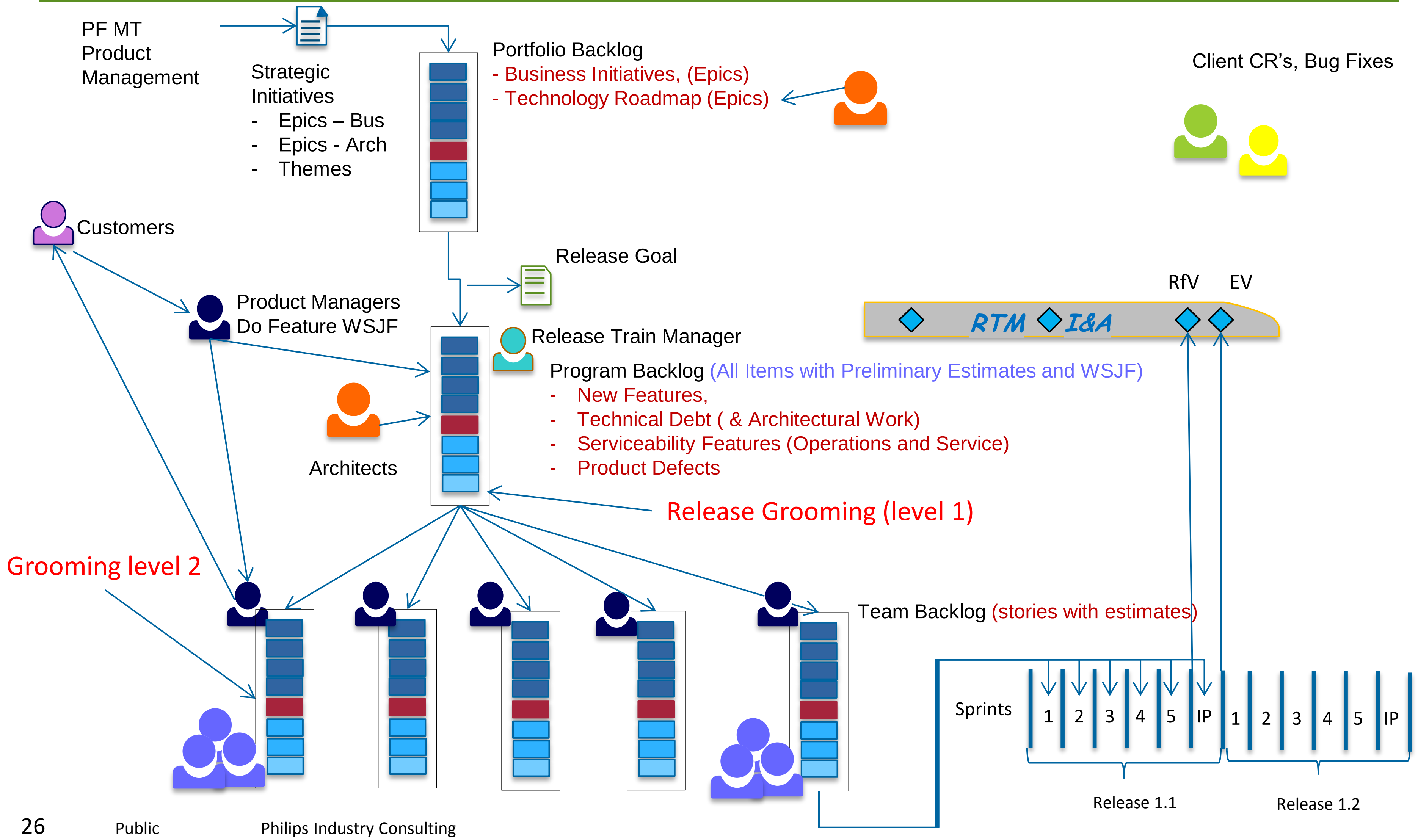
- Using SAFe in a platform environment
  - How can we speed up the customer feedback loop?

PHILIPS

QUESTION EVERYTHING

PHILIPS

Establish backlogs to introduce product features into portfolio and groom to project, releases and iterative implementation



PF MT Product Management
Strategic Initiatives
- Epics – Bus
- Epics - Arch
- Themes
Portfolio Backlog
- Business Initiatives, (Epics)
- Technology Roadmap (Epics)
Client CR's, Bug Fixes
Customers
Release Goal
RfV
EV
Product Managers Do Feature WSJF
RTM
I&A
Release Train Manager
Program Backlog (All Items with Preliminary Estimates and WSJF)
- New Features,
- Technical Debt (& Architectural Work)
- Serviceability Features (Operations and Service)
- Product Defects
Architects
Release Grooming (level 1)
Grooming level 2
Team Backlog (stories with estimates)
Sprints
1
2
3
4
5
IP
1
2
3
4
5
IP
Release 1.1
Release 1.2