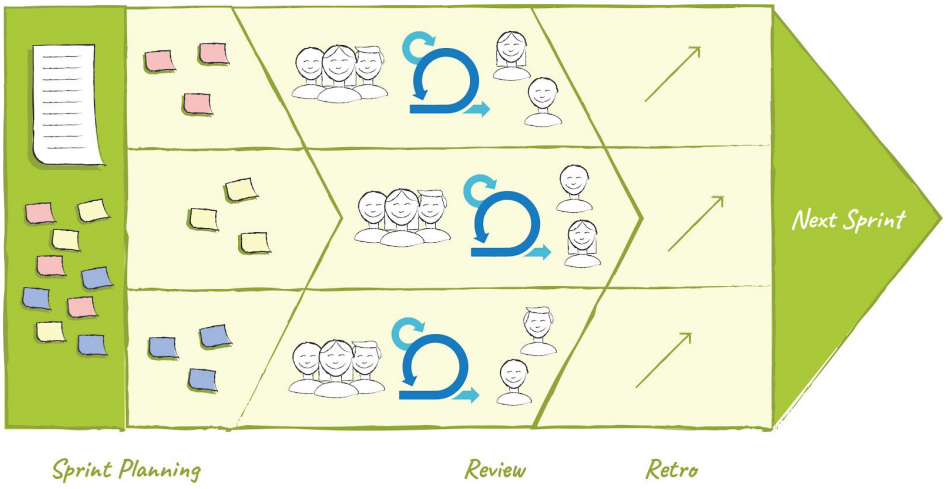
Scaled agile in de praktijk: welke modellen zijn er en wat werkt het beste in jouw situatie?

‘Nothing beats an agile team!’ Lang leve het agile team dat zich tijdens elke sprint verder verbetert. Maar wat nu als meerdere agile teams moeten samenwerken? Bijvoorbeeld omdat ieder teams een stuk van het totale informatiesysteem bouwt of omdat er koppelingen nodig zijn. Om het totale proces te kunnen laten werken moeten de teams dan samenwerken. Hoe regel je team overschrijdend samenwerken goed met agile teams? Daarop krijg je antwoord in dit artikel.

Scaled Agile modellen vergeleken

De meest eenvoudige manieren van scaled agile zijn LeSS en Nexus. Met deze raamwerken werken meerdere teams zo eenvoudig mogelijk parallel aan elkaar en zijn ze op elkaar afgestemd door gezamenlijke *sprint plan sessies*, r*eview meetings* en *retrospectives*. Ervaren scrumteams kunnen deze raamwerken in zeer korte tijd uitstekend toepassen.

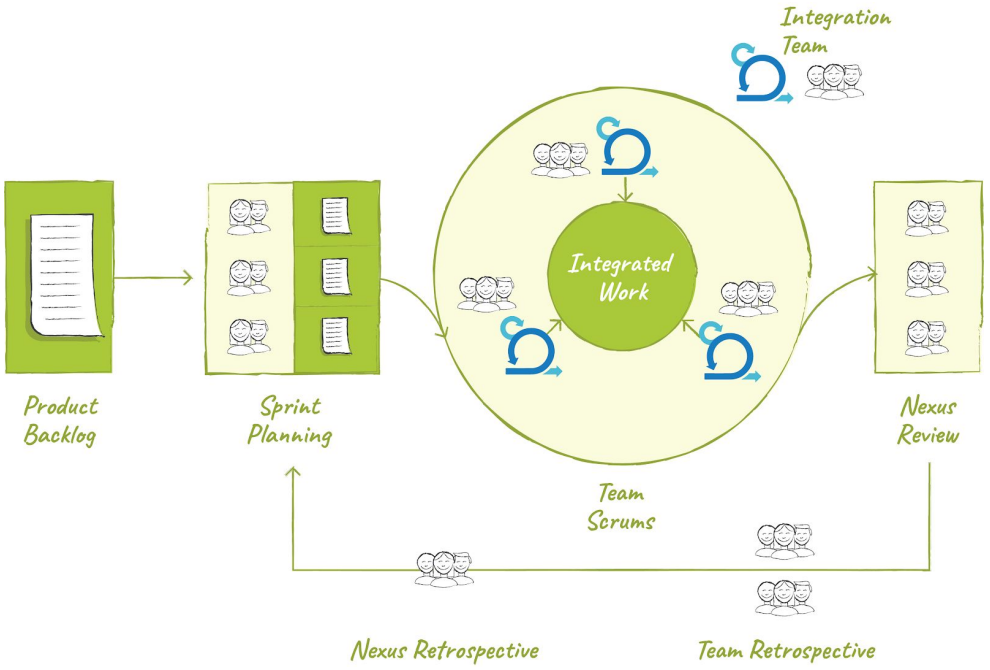
LeSS



Met het LeSS framework werken meerdere scrumteams parallel aan dezelfde product backlog. Na een gezamenkijke plansessie plant ieder scrumteam zijn eigen sprint.

Aan het eind van de sprint is er een gezamenlijke demo, een team retro en gezamenlijke retrospective meeting. LeSS heeft als groot voordeel dat de teams goed van elkaar weten waar zij mee bezig zijn en er gezamenlijke feedback van de stakeholders wordt verkregen. LeSS is een krachtig en eenvoudig framework voor teams die samen aan één programma met een gezamenlijk totaalproduct werken.

Nexus



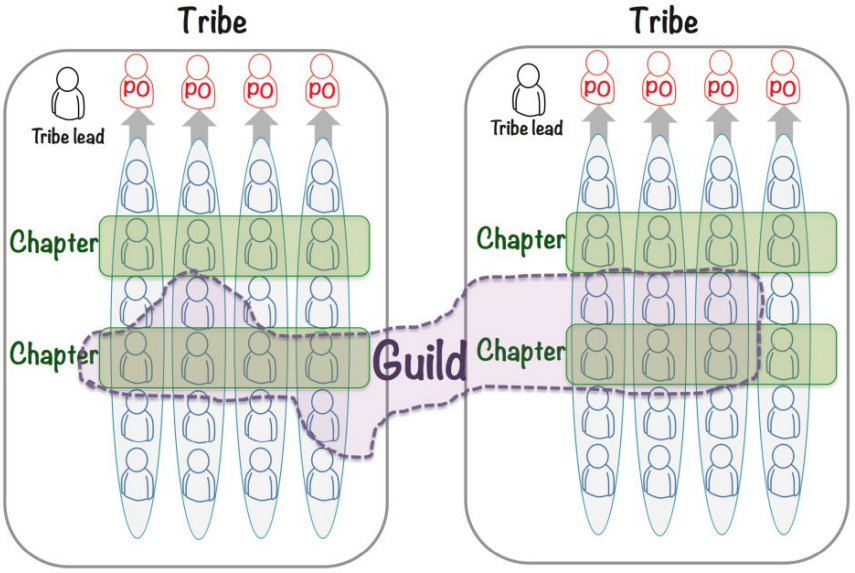
Nexus ziet er op het eerste gezicht anders uit dan het LeSS framework maar lijkt er toch erg veel op. Ook hier werken de teams vanuit één gezamenlijke product backlog en één Nexus sprint planning waar afgevaardigden uit de teams de team sprint backlogs bepalen.

De teams hebben ook hier na hun sprint een gemeenschappelijke sprint review meeting, retro en team retrospective meeting.

Het integreren van werk tot één gemeenschappelijk product is hier wat prominenter aangegeven maar kan ook prima met Less. Nexus is met deze focus ook heel geschikt om bijvoorbeeld met een team op locatie en een team op afstand te werken, die samen hun product afstemmen met stakeholders en naar productie brengen.

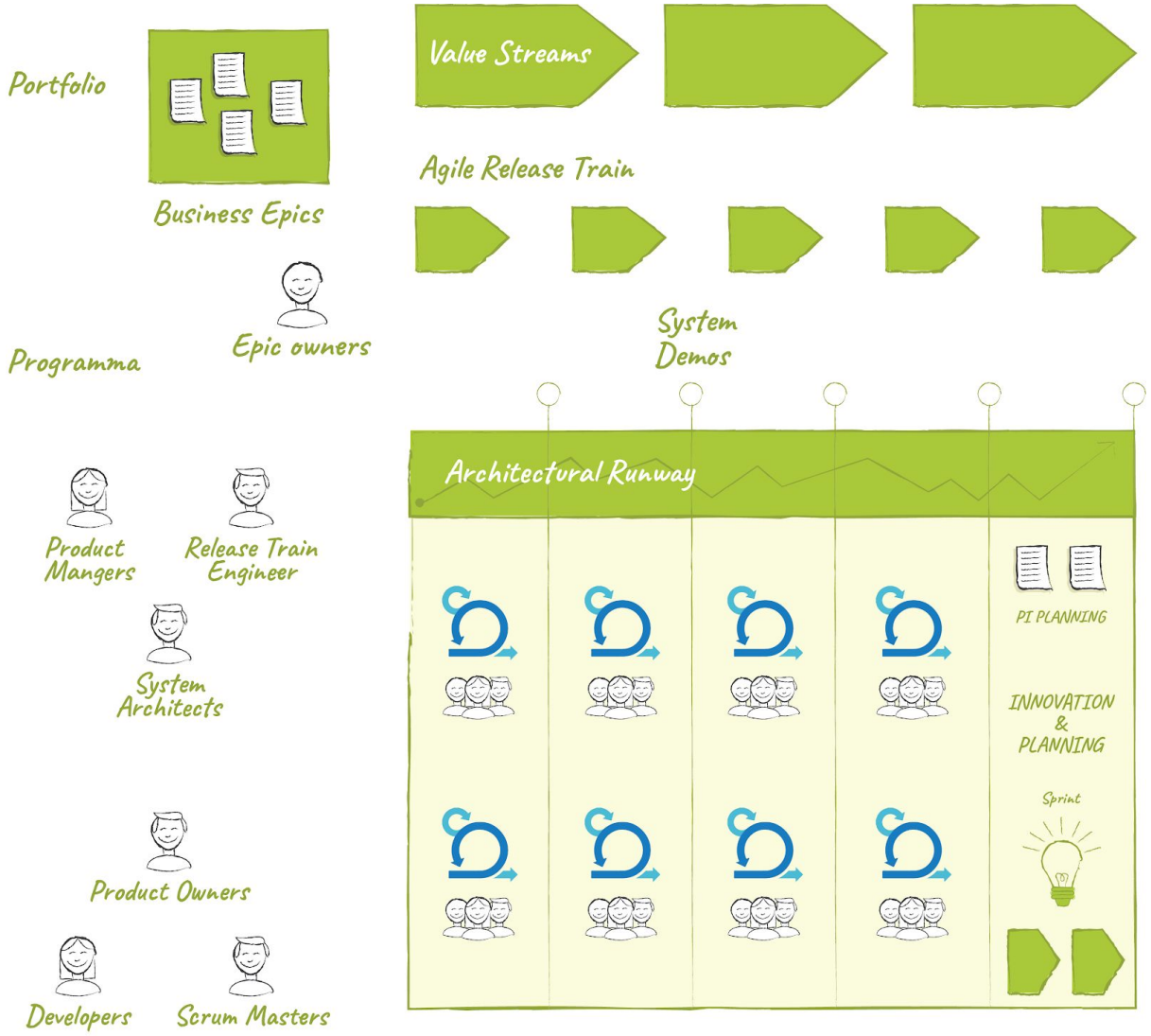
Het Spotifymodel

In het Spotifymodel ontstaan de structuren vanuit de teams zelf (!), Het hele idee is dat er net genoeg structuur is om een effectieve samenwerking te hebben. En ook dat structuren weer worden opgeheven zodra ze niet meer nodig zijn.



De kracht van het Spotifymodel zie je vooral in een combinatie van verschillende werkwijzen. Enerzijds werken medewerkers met *verschillende* expertises gegroepeerd in scrumteams (ook wel squads genoemd) samen om zelfstandig een product of dienst op te leveren. Anderzijds delen medewerkers met *dezelfde expertise* (chapters) of interessegebieden (guilds) onderling kennis en ervaringen uit zodat zij zich op dat onderwerp kunnen doorontwikkelen. Deze werkstructuren heffen zich weer op als ze niet meer nodig zijn. Dit werkt uitstekend als de vereiste organisatiecultuur hiervoor al bestaat en er in de organisatie ‘net genoeg regels bestaan om chaos te voorkomen’. Het Spotify model kent ook best practices als het werken met feature toggles en gefaseerde uitrol die in alle situaties waardevol zijn.

Het SAFe-model



Scaled Agile Framework (SAFe)

Voor veel organisaties voelt het SAFe-model als een warm bad. Het biedt veel meer structuur dan eerdergenoemde modellen. SAFe investeert ook om die structuur en sturing voor elkaar te krijgen. Bijvoorbeeld door middel van innovatie- en planningssprints. Vergeleken met de eerdergenoemde modellen sluit het beter aan bij bestaande (niet agile) processen. Maar daarin schuilt ook meteen het gevaar! Je bouwt daardoor namelijk onbewust niet-agile patronen in (te veel controle en zijsturing op de eigenlijke opleveringsprocessen). Zo kunnen de regels weer belangrijker worden dan het samenwerken. In het slechtste geval verlies je hierdoor zelfs 20% tot 30% van je ontwikkelsnelheid. Weet je het wel agile genoeg te houden, dan krijg je er een gestructureerde portfolio sturing, architectuur en innovatie voor terug, waarmee je veel efficiënter bent. Vooral voor complexere projectprogrammas is dit van toegevoegde waarde.

Hoe gaan de verschillende modellen om met het afstemmen van de afhankelijkheden tussen de teams?

Bij LeSS en Nexus werken de teams vanuit één gezamenlijke *product backlog*. Het Spotify-model organiseert het werk zo dat de teams los van elkaar producten kunnen produceren. Dit kan door de onderdelen *loosely coupled* te maken en met *feature toggles* te werken. En SAFe besteedt juist veel aandacht aan portfolio sturing en het gezamenlijk managen van afhankelijkheden.

Voor- en nadelen

Onderstaande tabel zet de verschillen tussen de scaled agile modellen op een rij.

	LeSS, Nexus	**Spotify**	**SAFe**
Inspanning	makkelijk	makkelijk	gemiddeld - zwaar
Agile	100%	100%	bewaken
Portfolio sturing	licht	indien nodig	sterk
Schaalbaarheid	2-9 teams	tot ca 30 teams	4-9 teams per onderdeel
Training	ervaren scrumteams kunnen direct starten	2 youtube videos	certificeringen en coaches
Focus	samen opleveren van wat al bij elkaar hoort	innovatie en community boven structuur	samen portfolio managen

Welk model is voor jouw situatie het beste?

Dat hangt er maar net vanaf. In elk geval is het altijd beter om het aantal teams te minimaliseren als dat kan. Dan hoef je immers minder afhankelijkheden te managen. Maar, hoe doe je dat dan?

A) Door de complexiteit te verkleinen
B) Door de productiviteit te verhogen

Complexiteit verkleinen

1) **Standaardiseren**
   Veel productiestraten zitten gevangen in hun bestaande structuur doordat ze een lappendeken aan applicaties in stand houden.

Door meer te *standaardiseren* kunnen teams beter elkaars werk overnemen en is het geheel beter schaalbaar.

2) **Hergebruik**
Met dit scenario leggen de teams de nadruk op *hergebruik*. Bijvoorbeeld door dezelfde api's te gebruiken voor dezelfde processen in verschillende kanalen. Het risico hiervan is dat teams afhankelijk blijven van elkaar en dat ze 'op elkaar zitten te wachten'. Om dit goed te doen is een krachtige organisatie en eigenaarschap voor het API management vereist. Je moet het samen echt willen om dit goed te laten werken.

Productiviteit vergroten

1) **Paardenmiddelen inzetten**
Om de productiviteit te vergroten kun je twee paardenmiddelen inzetten: een senior ingewerkt team inkopen of productiviteitsverhogende tools inzetten zodat je meer user stories per sprint kunt opleveren. Voorbeelden van productiviteitsverhogende tools zijn low code platformen als Pegasystems, Broadridge, SalesForce, Mendix en Outsystems.

2) **Teams naar bedrijfsproces organiseren**
Een grote productiviteitsverhoging bereik je ook door teams naar bedrijfsproces te organiseren. Bijvoorbeeld door één team zelfstandig (en volledig verantwoordelijk) een bedrijfsproces (hoe klein ook) te laten ontwikkelen, onafhankelijk van andere teams. Dit is vergelijkbaar met het Spotify-model; het team kan alles voor dat bedrijfsproces volledig zelfstandig en onafhankelijk verbeteren en produceren. Voor alle distributiekanalen en apparaten en onafhankelijk van de services van anderen.

De fundamenten echt op orde brengen is lastig, vergt moed en kost energie en tijd. De afhankelijkheden die je hebt kun je maar beter zo efficiënt mogelijk managen. Maar de basis blijft dat niets zo effectief kan werken als een opererende agile team. Keep it simple als dat kan!

Als de teams de overgebleven afhankelijkheden onderling kunnen regelen in gezamenlijke plansessies en reviews (zoals dat bij het LeSS of Nexus-model gaat) heeft dat absoluut de voorkeur. Als de afhankelijkheden groter zijn kan een Release Train Engineer (zoals in SAFe) het afstemmen van de afhankelijkheden faciliteren. Gezamenlijke plansessies versterken dan de afstemming van afhankelijkheden en het werken aan de doelen van het portfolio.

Een sterk punt uit SAFe is de Agile portfolio planning vanuit business epics. Dit helpt om de prioriteiten in de Agile teams een veel duidelijkere focus te geven. Dit is tevens een mogelijkheid om de complexiteit te verminderen.

Stappenplan Scaled Agile

Onderstaande checklist kun je in de praktijk toepassen om het afstemmen van de afhankelijkheden tussen agile teams zo eenvoudig mogelijk voor elkaar te kunnen krijgen:

Hoger management wil weten waaraan wordt gewerkt?	ja	Maak een roadmap per team en richt een ‘Product Owner overleg’ en ‘Project Board’ in
Zijn er grote programma’s die met elkaar concurreren	ja	Maak per programma value streams met eigen teams
Moeten de teams veel samenwerken om doelen te bereiken?	nee	Enjoy en Stay Agile!
	ja	Standaardiseer op technologie Automatiseer tests en deployments en integreer vaak Faciliteer (API) hergebruik Werk DevOps Zet productiviteitsverhogende (low code) tools in. Blijf weg van ‘eigen’ architecturen. Organiseer de teams naar bedrijfsproces over de kanalen heen
Zijn er nog steeds afhankelijkheden te managen?	Ja	Just do it and keep improving Werk met gezamenlijke plansessies en demo’s (LeSS, Nexus) of agile release trains met plansessies als je in een complex programma werkt.

QbResult
Partner in Agile