

# Large scale agile development of S60 product software

A few hundred synchronized Scrums - setup and experiences

**NOKIA**

## Juha-Markus Aalto

Director, Operational Development

Nokia, S60 SW

2008-11-26

1

## Topics

- What's common with nesting Russian dolls and scaling Scrum?
- Scaling up agile
  - teams
  - sprints
  - backlogs
  - Definition of Done
  - tools
- Challenges
- Experiences

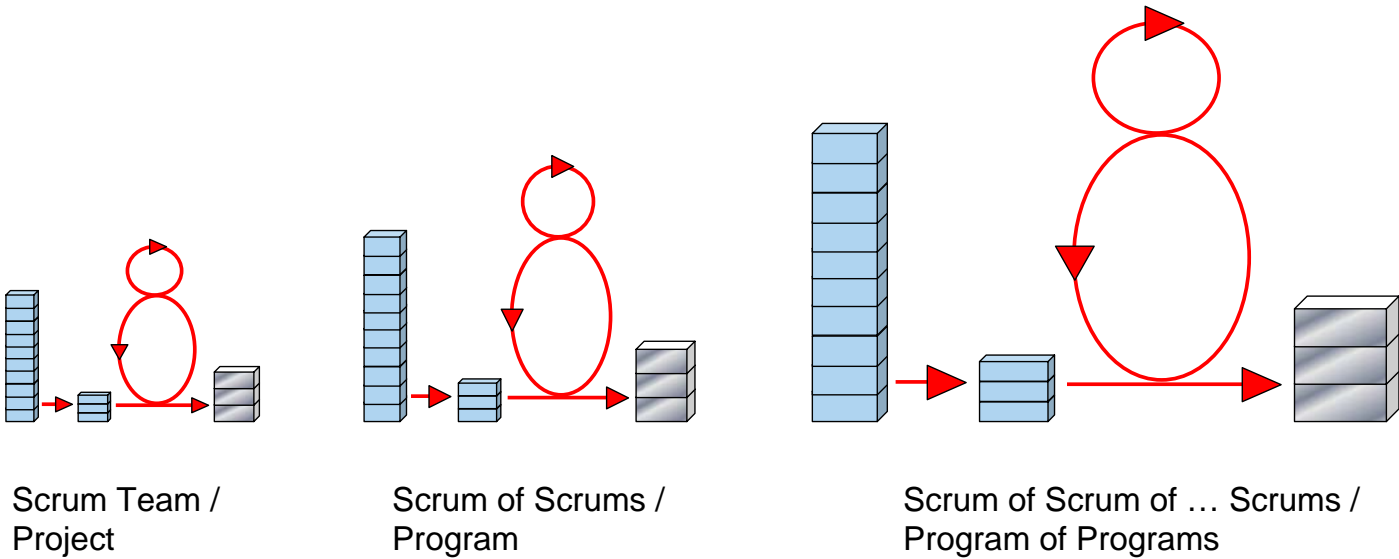
2

**NOKIA**

# Scaling up nesting Russian dolls?

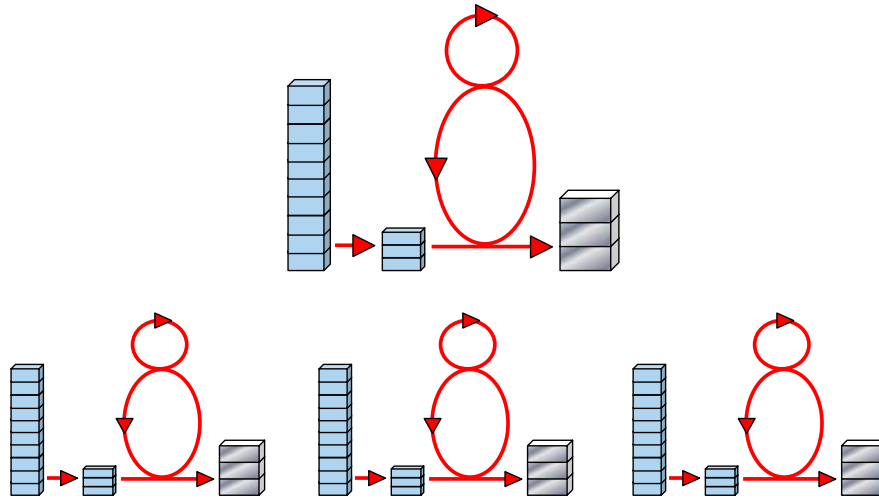


# Scaling up Scrum?



# Agile programs – the obvious next step after agile teams

- Agile Scrum teams instead of projects
- Program requirements prioritized and managed as Program Backlog
- Hardened incremental deliveries integrate the results of Scrum teams
- By definition, a program has a program manager that interfaces agile teams



**NOKIA**

## WaterScrum dilemma: Agility may be drowned by waterfall-like programs – and vice versa

- *Programs* have expectations for agile projects to give waterfall-like commitments
- Programs push aggressive targets and manage dependencies through plans with command and control mentality
- Immature agile teams, in turn, may interpret agility incorrectly as “no commitments, no plans” beyond the next sprint
- True agile teams are self-organizing, pull the work and manage their own dependencies

Presenter’s personal intermediate conclusion end of November, 2008:  
Program-like approaches, preferably lean ones, are still needed in complex SW-HW product integration, but SW development can and should be agile.

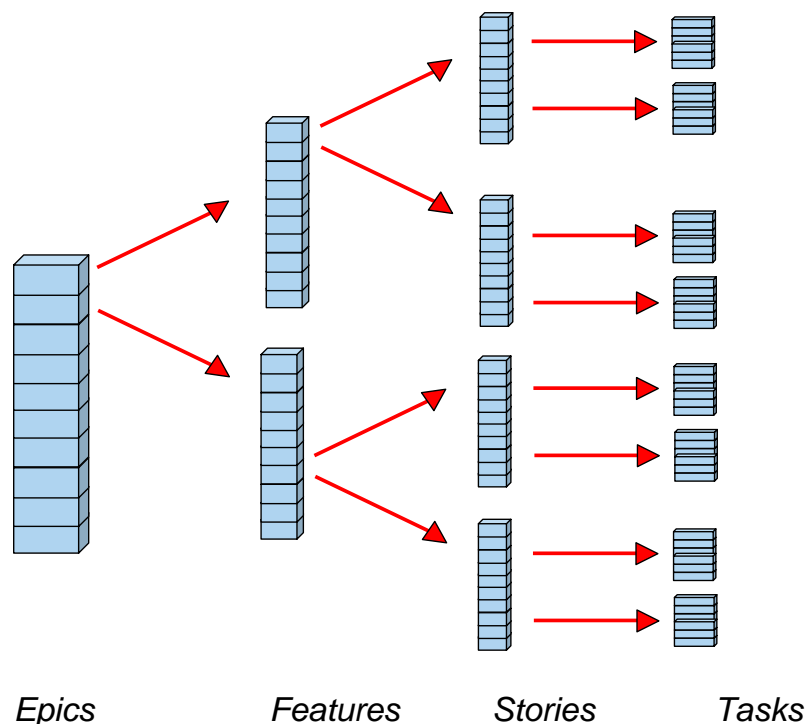
# Scaling up the backlogs

- Our first try: team, program and portfolio backlogs
  - team level backlogs as user stories, program content backlogs as requirements
  - portfolio backlog as a list of prioritized programs
  - results: OK to start team level agile deployment but sub-optimal in many ways
- The new approach being deployed: agile requirements model based on Epics, Features and Stories
  - all functional requirements managed as product backlog items that have explicit containment hierarchy
  - portfolio backlog expressed in terms of **Epics**, not as a prioritized list of programs
  - release backlogs introduced with **Features**
  - **Stories** used for team level backlogs as before, Scrum at team level

NOKIA

7

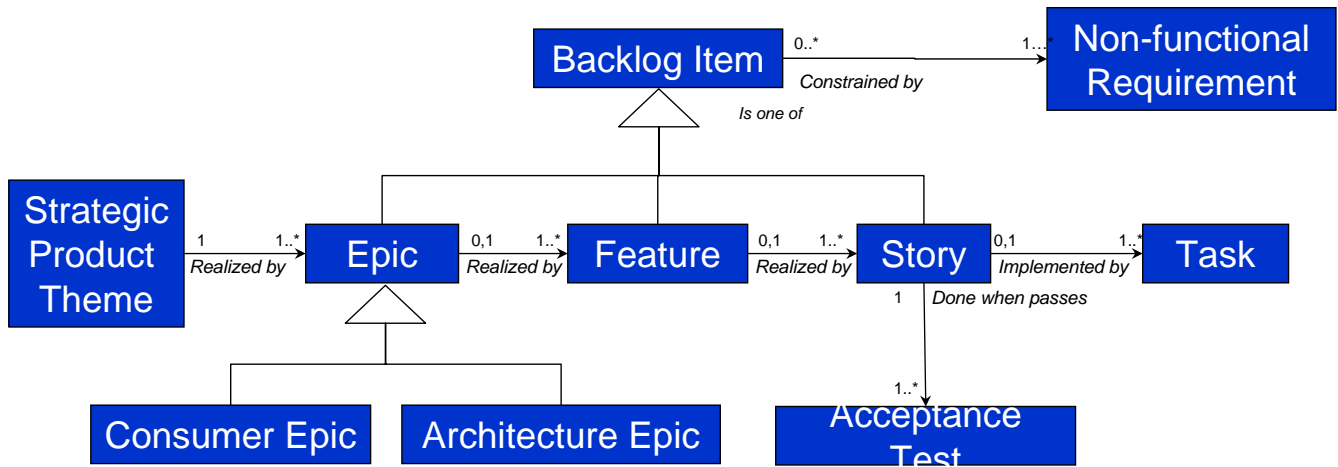
## Epics breakdown into features, elaborated in stories, implemented by tasks



8

NOKIA

# S60 SW Agile Requirements Model



## Epics are elaborated in terms of Features and Stories incrementally, just in time



As an OVI Share user, I want to share my pictures.

As an OVI Share user, I want to upload a photo from my phone.

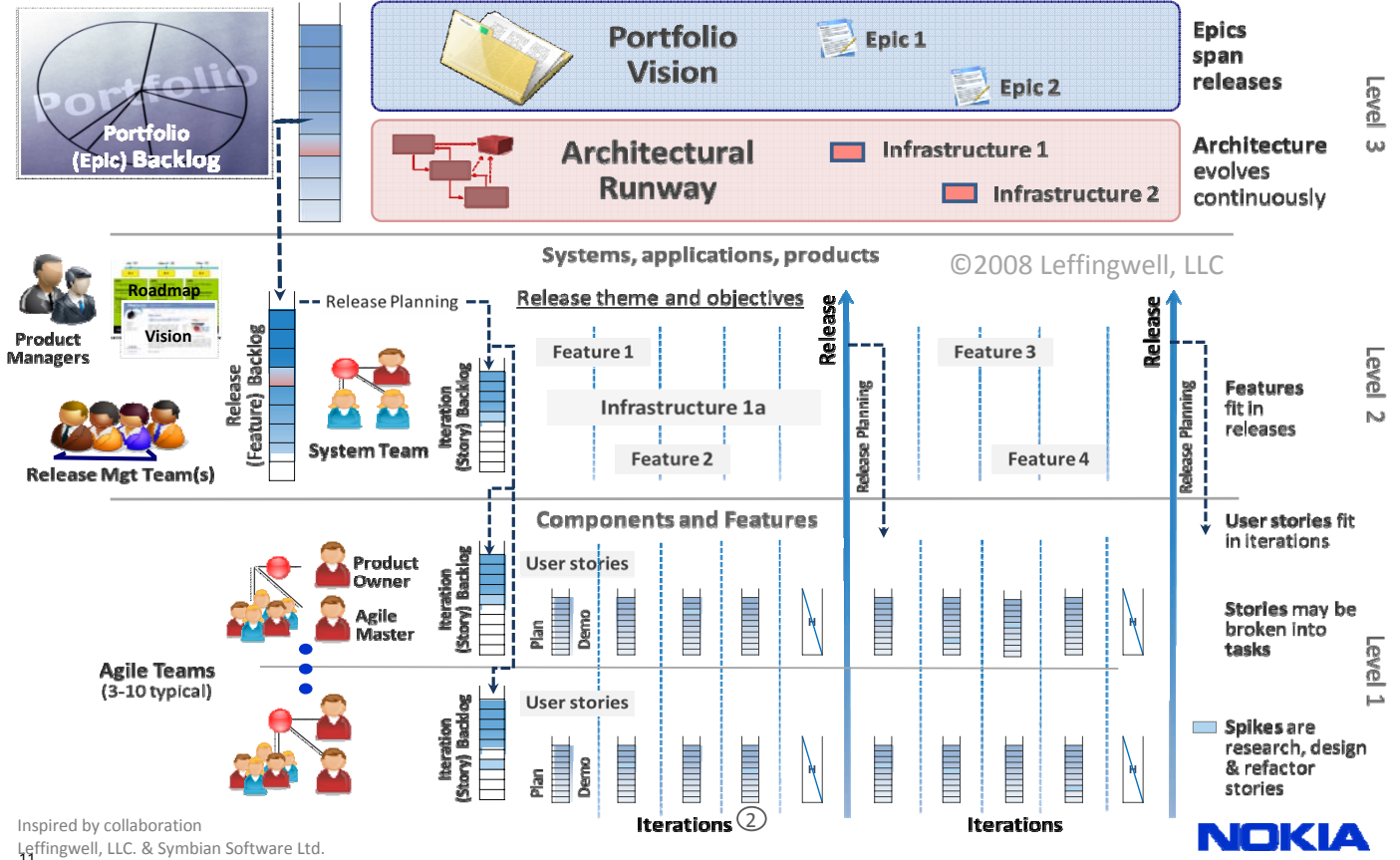
As an OVI user, I want to select the photo to be uploaded.

...

...

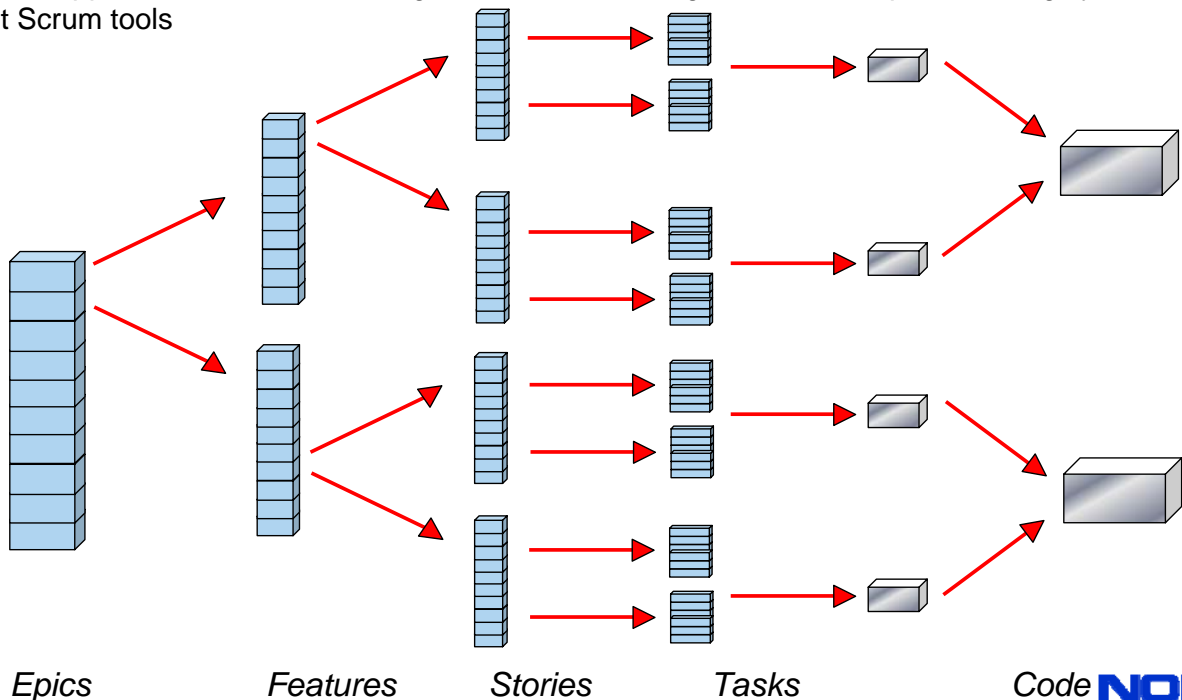
# The Agile Enterprise Big Picture

Used in this presentation with Dean Leffingwell's permission For discussion, see [www.scalingsoftwareagility.wordpress.com](http://www.scalingsoftwareagility.wordpress.com)



## Also the tools need to scale up

- In large scale development, end to end transparency is important for effective collaboration
- The tools need to support requirements traceability, analysis and portfolio estimating without driving any team overhead.
- Proper support for hierarchic backlogs and distributed large scale development is largely missing from most Scrum tools



# Summary

- For us it seemed to be a good choice to start from agile teams, bottom up
- At all levels, there are teams, backlogs, iterations, and *potentially* shippable increments with some Definition of Done
- Program-like approaches, preferably lean ones, are still needed in complex SW-HW product integration, but SW development can and should be agile.
- Useful backlog structure is critical when scaling agile
- Yes, agile scales up!