Improve Your Team's Project Sizing with Scrumban RBS

Dimitar Bakardzhiev
Managing Director
Taller Technologies Bulgaria
@dimiterbak

Ajay Reddy
Founder, CodeGenesys
@ajrdy

@dimiterbak

Dimitar Bakardzhiev is the Managing Director of Taller Technologies Bulgaria and an expert in driving successful and cost-effective technology development. As a Lean-Kanban University (LKU)-Accredited Kanban Trainer (AKT) and avid, expert Kanban practitioner, Dimitar puts lean principles to work every day when managing complex software projects with a special focus on building innovative, powerful mobile CRM solutions. Dimitar has been one of the leading proponents and evangelists of Kanban in his native Bulgaria and has published David Anderson's Kanban book as well as books by Eli Goldratt and W. Edwards Deming in the local language.

Who We Are

A quick peak at our products, services & affiliations

Scrumban

http://scrumban.io and http://scrumban.com Research, Case Studies & Articles on Scrumban and Lean / Agile principles.

CodeGenesys

Boutique IT Solutions & Services firm that builds high-performing teams & organizations.

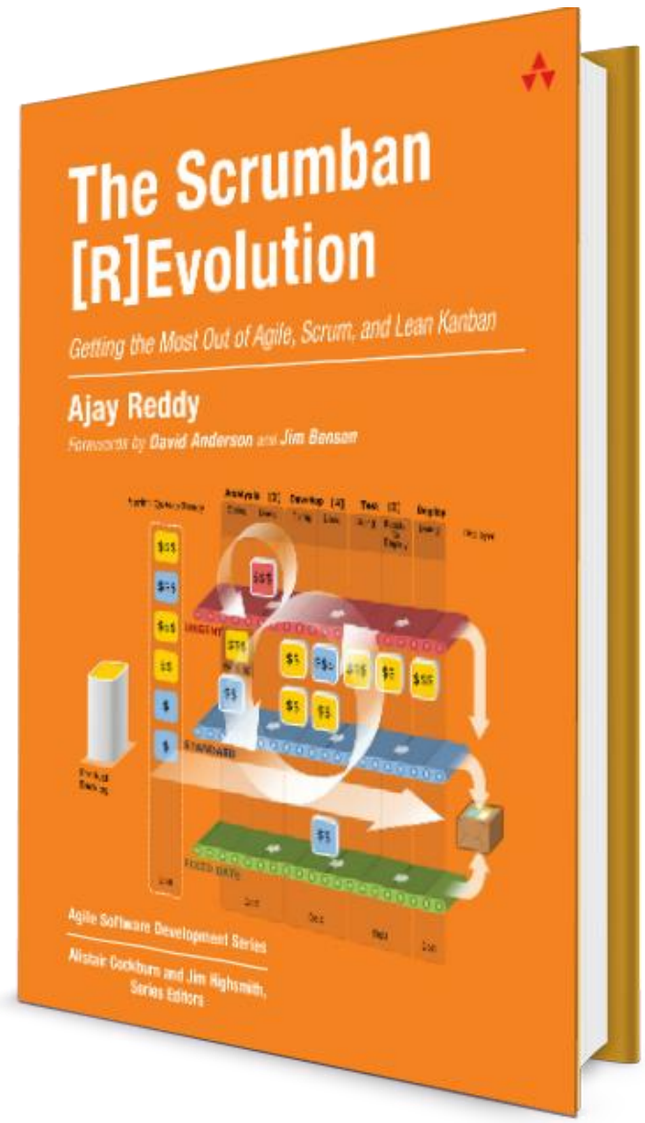
GetScrumban

The game trusted by 130+ trainers and coaches

ScrumDo

65,000+ users and counting
Your friendly neighborhood Scrum and Scrumban tool

LEAN KANBAN UNIVERSITY • CERTIFIED KANBAN TRAINING •



The Scrumban [R]Evolution
Pearson Education (2015)

Who We Help

A snapshot of organizations we've helped

State Street

ca technologies

Kroger

unum

CapitalOne

EBSCO

SimpliSafe Security Systems

clearfly

Walmart

Merchant Warehouse

IKARIA ADVANCING CRITICAL CARE

Jeff Sutherland @jeffsutherland · Apr 30
#Scrum is the minimal process necessary to manage complexity. Don't omit anything. Add patterns as needed. #Kaizen #Agile
60 41

David J Anderson @djaa_dja · May 1
@jeffsutherland is "a" not "the". It seems your book is full of absolutes that are easily proven false. Some humility is in order
1 4

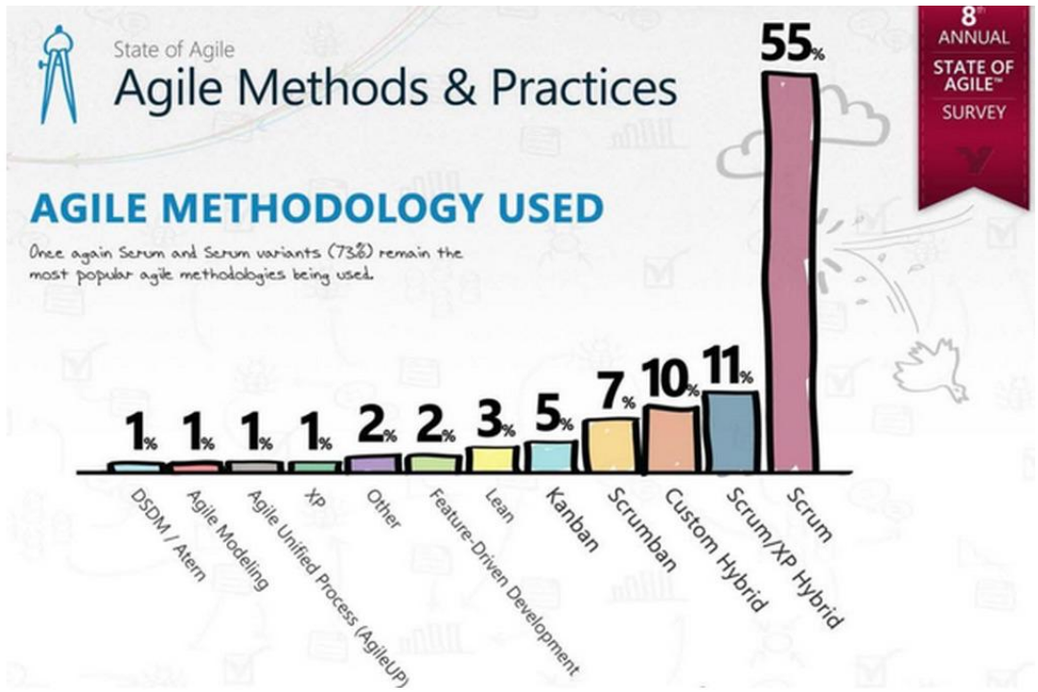
Jeff Sutherland @jeffsutherland · May 1
@djaa_dja Evidence?

David J Anderson @djaa_dja · May 1
@jeffsutherland I have a manuscript full of failed #scrum stories we've gathered from real firms some who had training from you!

Jon Fulton @JonFulton1982 · May 1
@djaa_dja @jeffsutherland Scrum Vs Kanban, which is better? Only one way to find out.... Fiiiiight! 😉
1 3

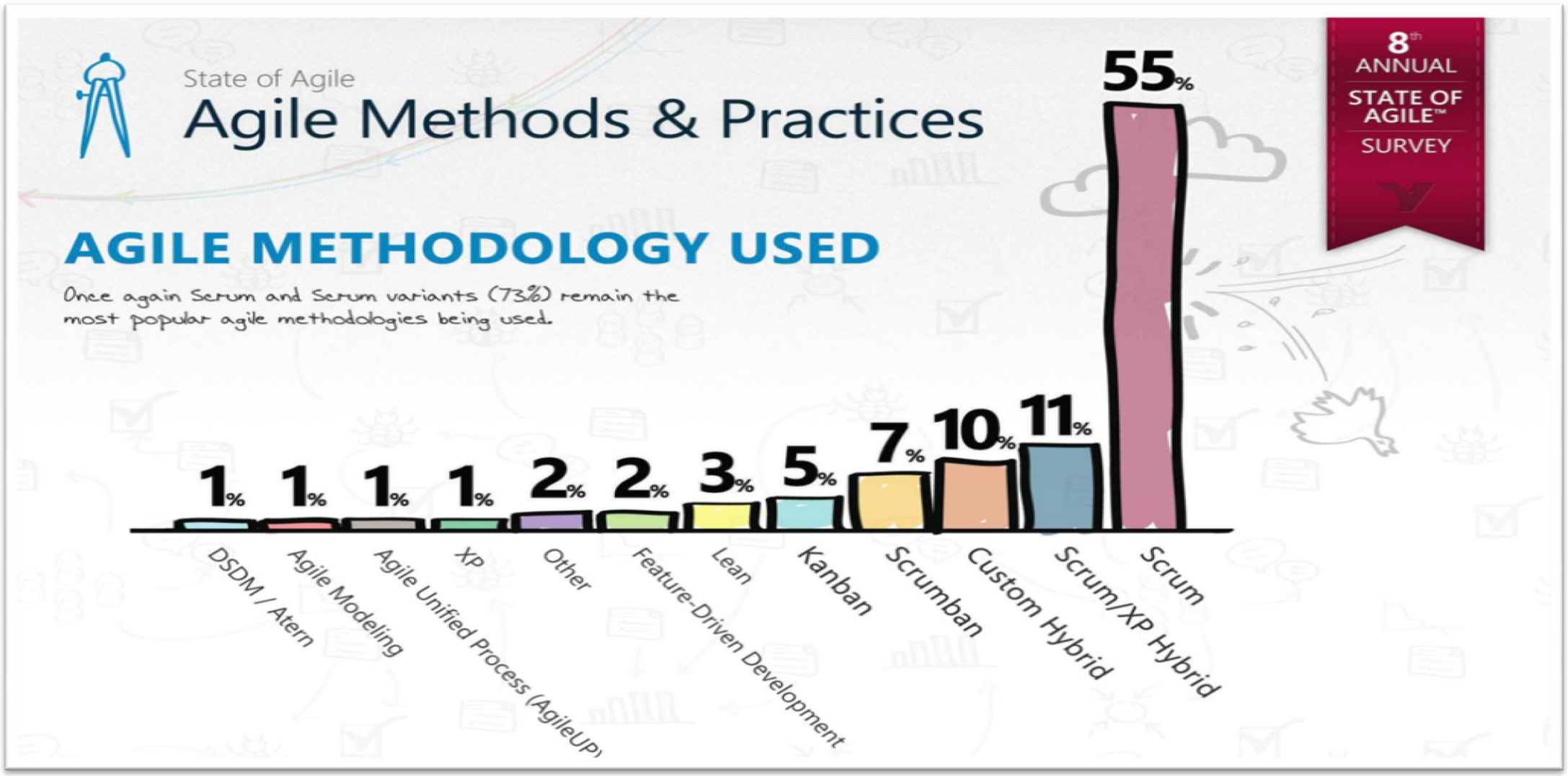
David J Anderson @djaa_dja · May 1
@JonFulton1982 @jeffsutherland which is better is the wrong question. Appropriateness and context is the right question.

“Broad is the way that leads to” ineffectiveness.



Jeff Sutherland @jeffsutherland · May 1
@djaa_dja #Scrum is one of many #Agile methods used. It helps manage complexity. I agree Its not the only way.

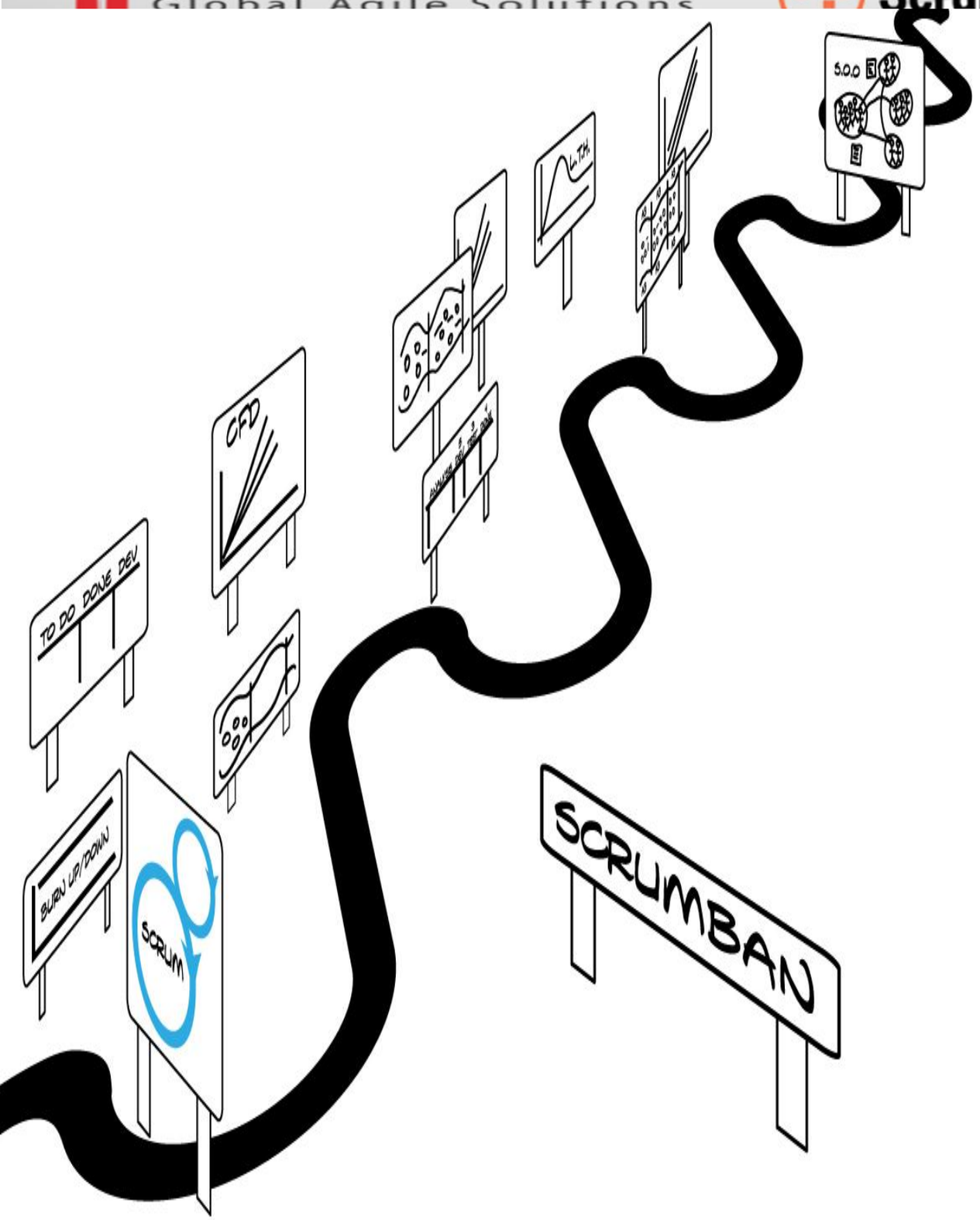
VersionOne's State of Agile across the industry



“75% of organizations using Scrum will not succeed in getting the benefits that they hope for from it.”

-Ken Schwaber

> “Scrum is a good example of an organizational framework. It has well-defined components, namely roles, meetings, artifacts, and values. These are fixed, and failure to embrace the whole usually results in a collapse of the framework.”
>
> *-Tobias Mayer*



What is Scrumban?

Three Essential Flavors

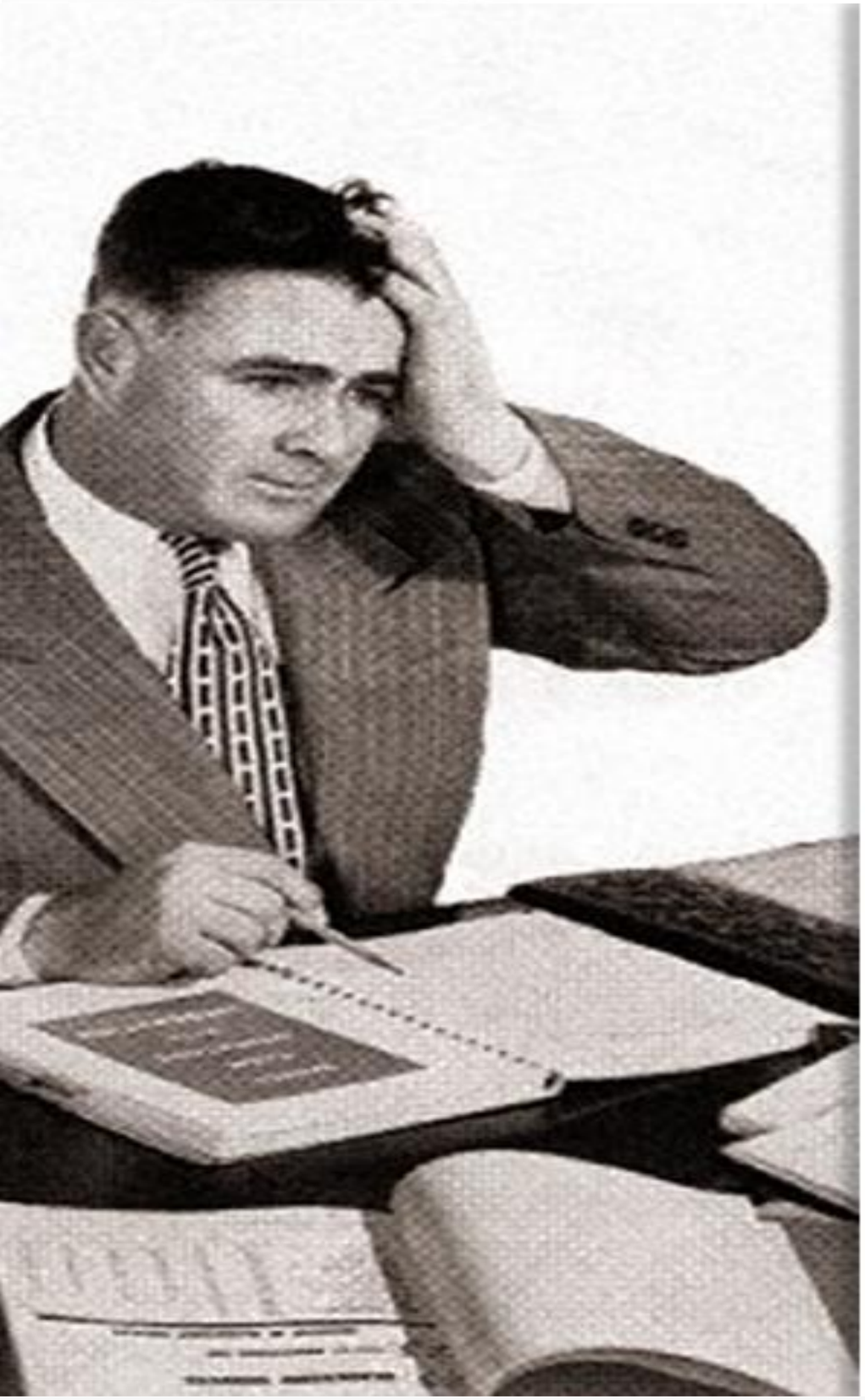
Using Kanban as a lens through which we can view and manage a Scrum work process.

Recognized Manifestations

1. A framework for introducing and adopting Scrum as a software development methodology.
2. A framework for overcoming common challenges with scaling Scrum across an Enterprise.
3. A framework for evolving from Scrum to a unique set of processes and practices.

- ✓ Cargo-cult implementation (practice focused implementation)
- ✓ Consistently broken commitments
- ✓ Disruptive implementation causing psychological barriers
- ✓ Product owner role poorly reflecting business , environmental, technical risks
- ✓ Forcing artificial team sizes
- ✓ Meetings taking too long
- ✓ Forcing co-location affecting retention rates of trusted employees

@ajrdy
ajay.reddy@codegenesys.com



Consistent Problem areas

Perceived productivity

Estimation

Politics, Agenda misalignment.

Profitability / “Cost overruns”

Process coaches & CC management

Plasticized, Increasing organizational resistance

How big is our project?

In order to forecast the time and the budget needed to deliver a new software product we need to be able to quantify "**what**" we are building since the resources required are related to "**how much**" software is built. That quantification is referred to as "**sizing**".

Software sizing is different from delivery time estimation. Sizing estimates the probable **size** of a piece of software while delivery time estimation forecasts the **time** needed to build it. The relationship between the size of a piece of software and the time needed to deliver it is referred to as **productivity**.

Agile sizing techniques

measure User Stories

T-Shirt sizes (Small, Medium, Large and so on)

http://www.mountaingoatsoftware.com/blog/estimating-with-tee-shirt-sizes

Story points (Fibonacci numbers or Exponential scale)

http://www.mountaingoatsoftware.com/blog/dont-equate-story-points-to-hours

Story points are about effort?

But…software sizing is different from software effort estimation!

http://www.mountaingoatsoftware.com/blog/story-points-are-still-about-effort

Sizing estimates the **probable size** of a piece of software while effort estimation estimates the **effort** needed to build it.

That makes it difficult to use **story points** for sizing a project unless we change the definition of a story point and equate it ONLY with **complexity**.

One dimension of complexity is the number of **tasks** per story.

Present days project sizing?

No matter which of function points, t-shirt sizes, story points and tasks we decide to use project sizing requires that all user stories in the product backlog are analyzed and estimated. Then we sum up all story sizes and arrive at the total project size.

This practice is time consuming and probably great part of this effort will be pure waste!

How can we estimate the total story points for a project without prior identification, analysis and sizing of every single user story?

Randomized Branch Sampling (RBS)

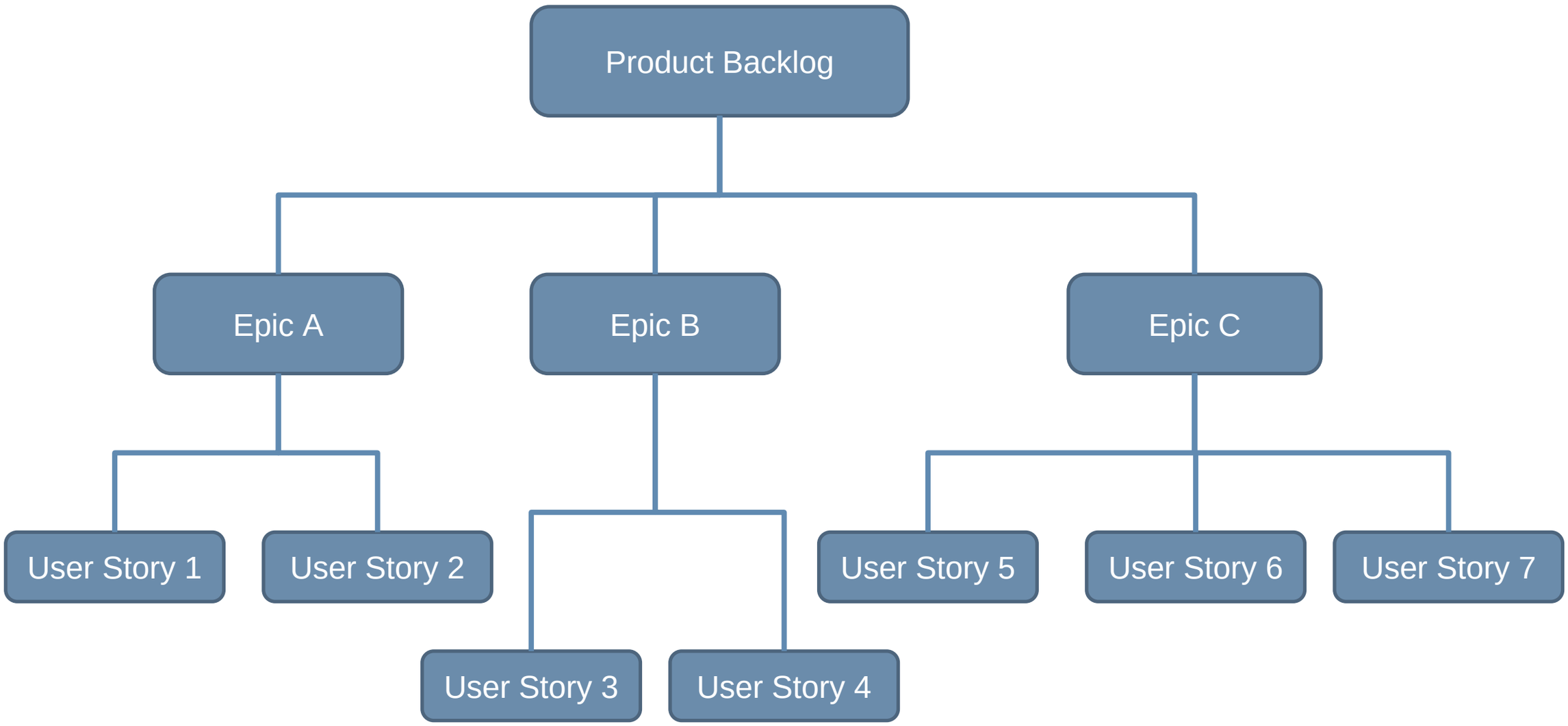
Raymond J. Jessen
1910–2003

The technique was designed to efficiently estimate the total number of fruit found in the canopy of a tree while only having to count the fruit on select branches. RBS is a method for sampling tree branches which does not require prior identification of all branches, and provides the sampler with unbiased tree level estimates.

Randomized branch sampling (RBS)

- A multi-stage unequal probability sampling method which doesn't require prior identification of all branches in the crown, and provides the sampler with unbiased tree level estimates.
- Designed to efficiently estimate the total number of fruit found in the canopy of a tree while only having to count the fruit on select branches.
- A tree level estimate is derived by combining the number of fruit from the terminal branch and the associated probability with which that particular branch was selected.

Product backlog as a branching system



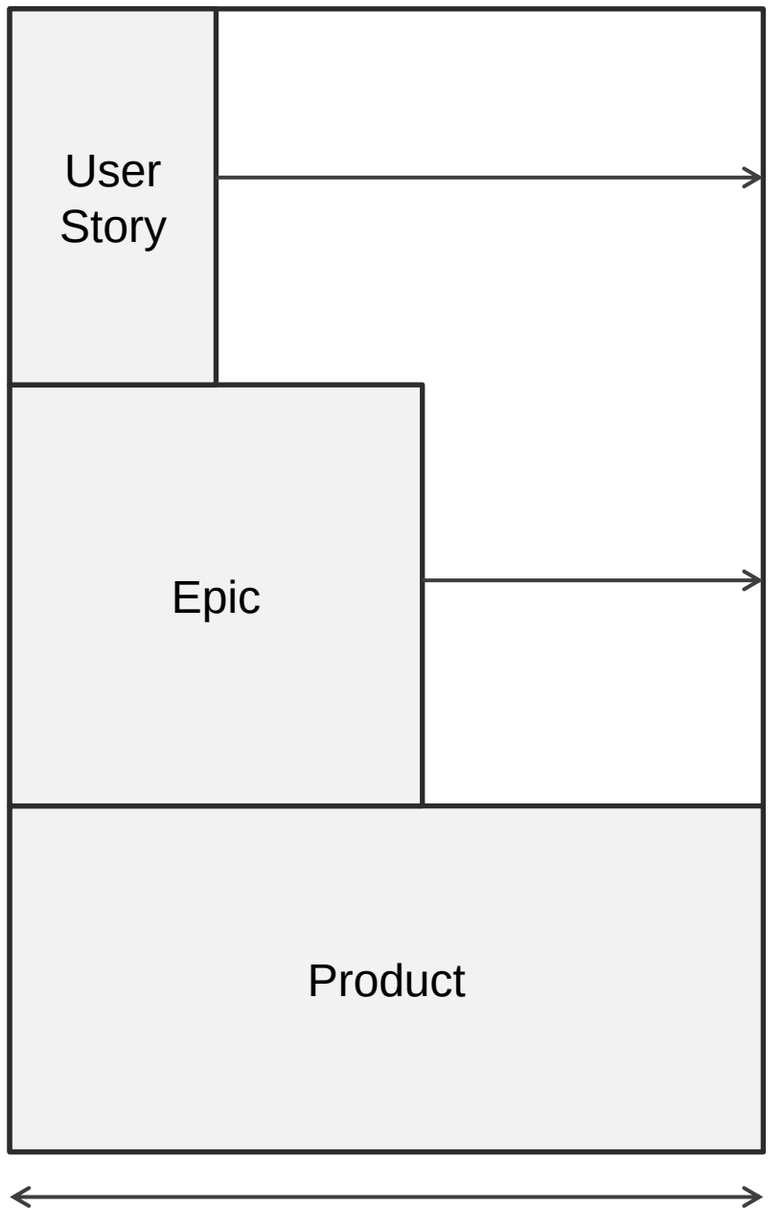
Horvitz-Thompson estimator

$$\hat{X} = \frac{x_i}{Q_i}$$

Unconditional selection probability (Q_i)

$$Q_i = \prod_{k=1}^{i} q_k = q_{product\ backlog} q_{epic} q_{story} = q_{epic} q_{story}$$

Applications of RBS to project sizing



The user story rectangle represents the estimated size of a randomly sampled user story. The size of that user story is expanded to an estimated total project size by dividing that size by its selection probabilities which is indicated here by the arrows. The selection probabilities assigned to epics and user stories are arbitrary. Unless the probabilities do not sum to one they will not affect the unbiasedness of the resultant estimate, but they will affect its precision.

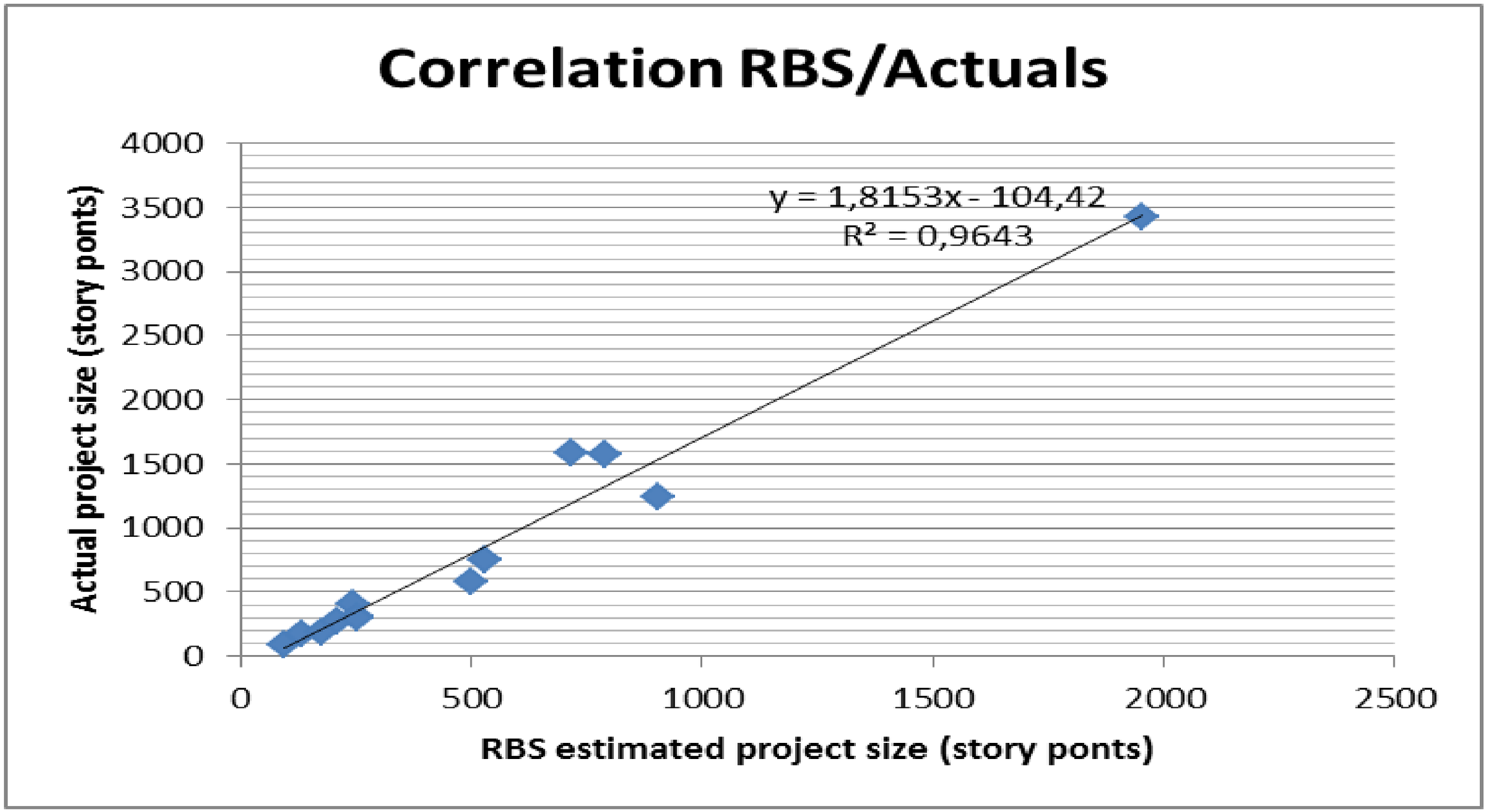
Is RBS applicable to software development?

The assumption behind using RBS for software development is that project size depends on the context – the customer, the people developing the product and the methodology they use for managing the requirements, breaking down the product into stories and sizing a story. It doesn’t matter what the methodology is. What is important is the methodology to be cohesive, explicit and to be consistently applied during project execution when we slice the requirements into user stories.

RBS compared to the actual results of 13 real ScrumDo.com projects

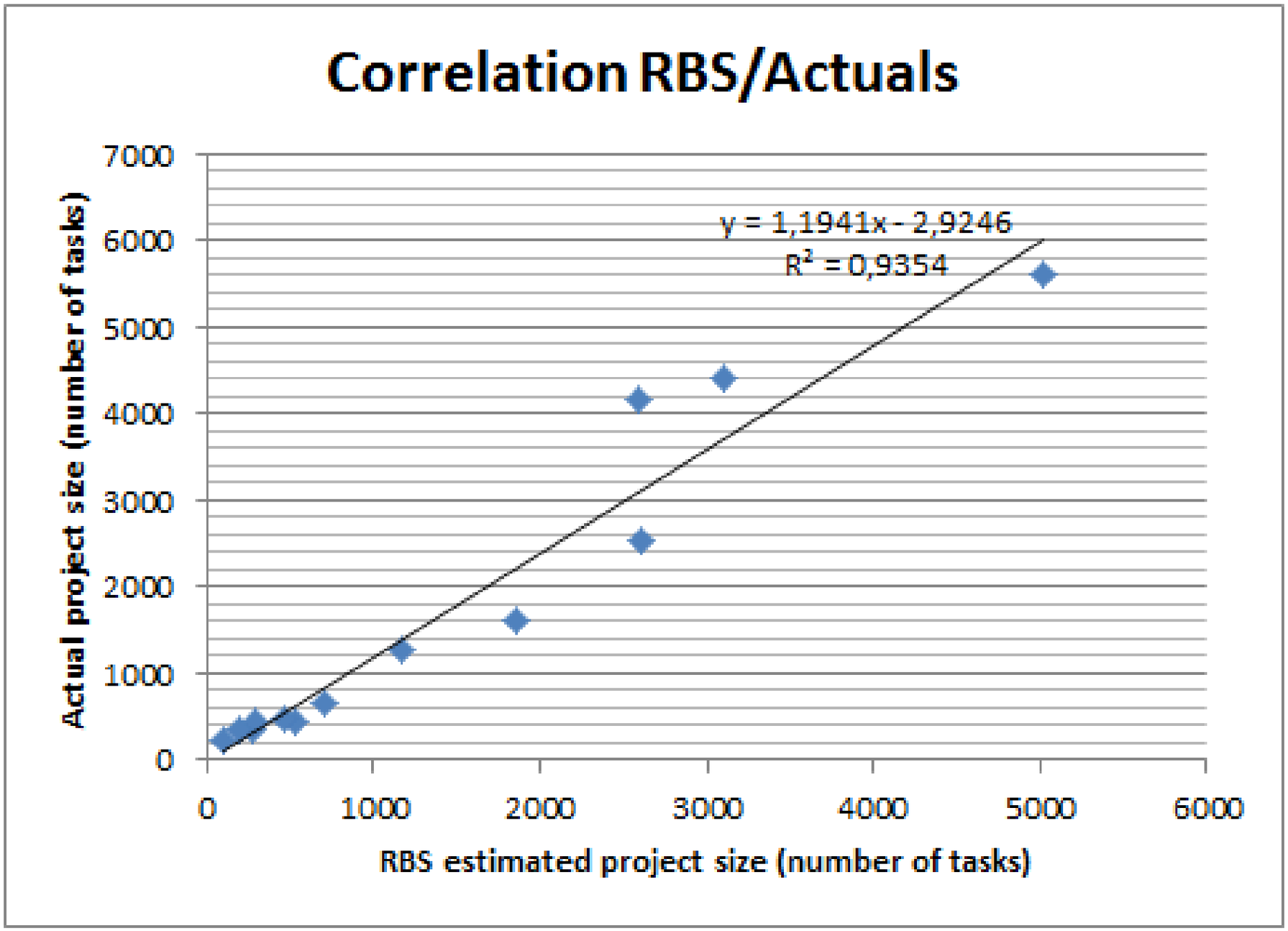
- Epic-Story-Task breakdowns
- Successful release history
- Stable teams (systems)
- Have an active ScrumDo coach or scrum master
- Commercial projects
- Have a minimum size of 12 epics/features.

RBS estimated total story points



ScrumDo data and results here.

RBS estimated total number of tasks



Conclusions from Scrumdo.com data

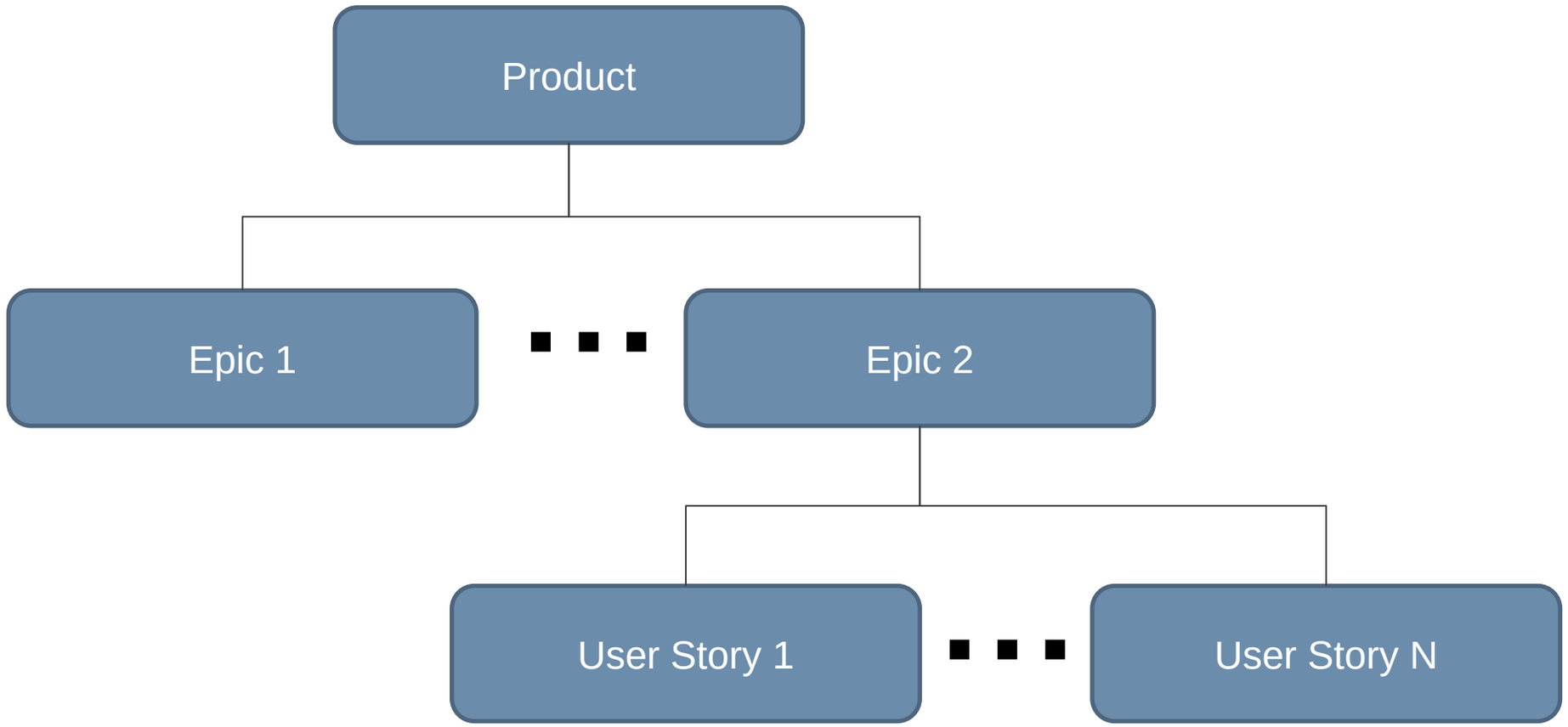
- During project execution all project teams consistently applied a methodology for slicing the requirements into user stories and sizing them using story points
- During project execution all project teams maturely managed the emergent and high-change-risk requirements
- Execution is more important than planning!!!

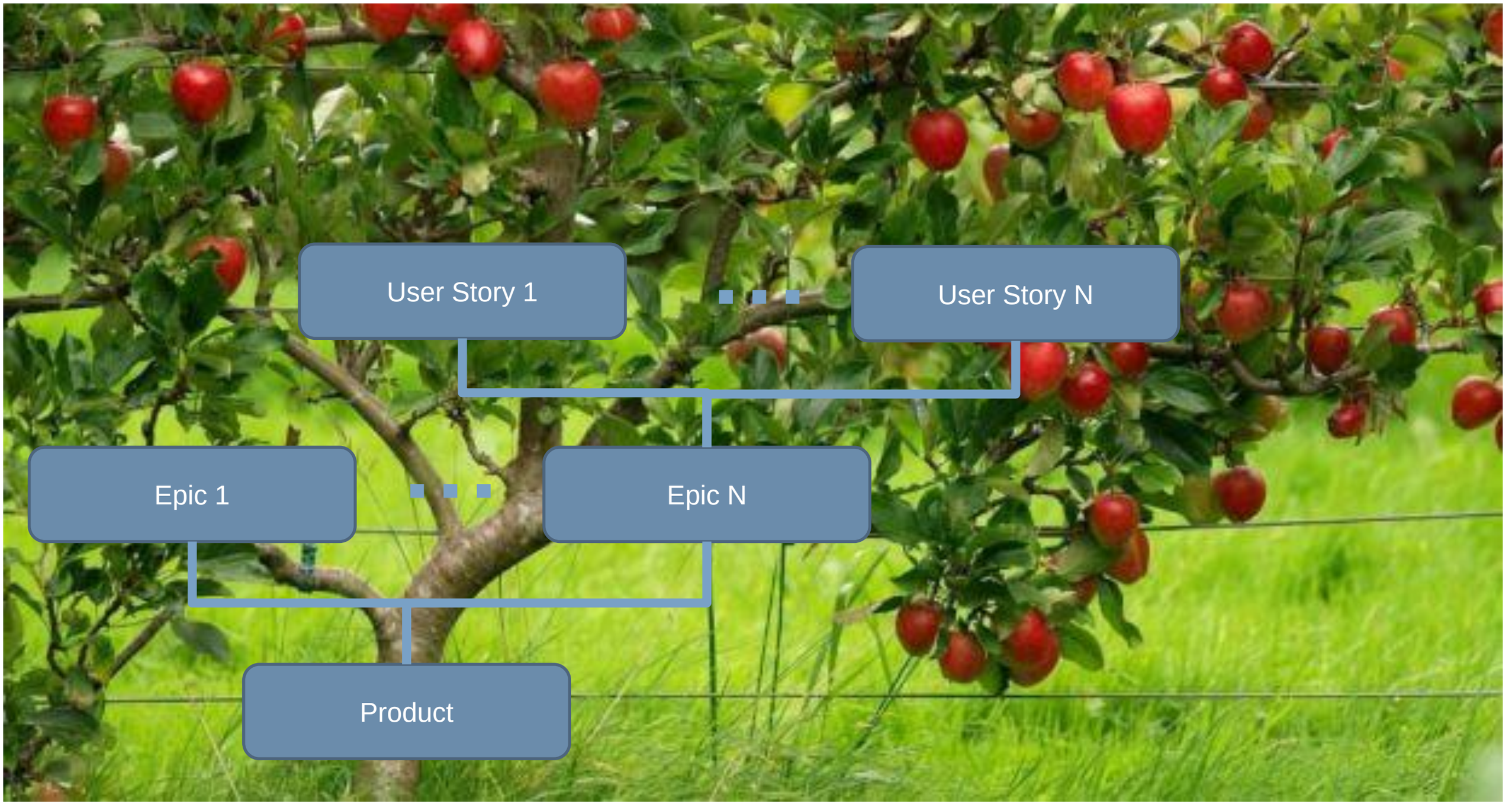
Applications of RBS

1. Applying RBS for estimating total number of user stories in a project
2. Applying RBS for estimating total Story points in a project
3. Applying RBS for estimating total number of tasks in a project
4. Assessing the **maturity** of Agile teams in their usage of a sizing methodology

Applying RBS for estimating total number of user stories in a project

Stories based sizing model




User Story 1
User Story N
Epic 1
Epic N
Product

Mapping

Product	Trunk
Epic	Branch
User Story	Terminal Shoot

RBS estimate of the of total number of user stories for a project

$$\hat{X}_i = \frac{\left(\begin{array}{c}Number\ of\ user\ stories\\ in\ the\ sampled\\ Epic\end{array}\right)}{\left(\begin{array}{c}\frac{1}{Number\ of}\\ Epics\\ in\ the\ project\end{array}\right)} \quad (1)$$

Where:

$\hat{X}_i$ is an estimate of the total number of user stories for the project.

Total number of user stories for the project

$$\hat{X} = \frac{1}{m}\sum_{i=1}^{m}\hat{X}_i = \frac{1}{m}\sum_{i=1}^{m}\frac{\hat{S}_i}{\frac{1}{n}} \quad (2)$$

$\hat{X}$ is an unbiased estimator of the total number of user stories for the project

$\hat{S}_i$ is the number of user stories in the m-th epic

m is the number of estimates done

n is the number of epics in the project

Algorithm

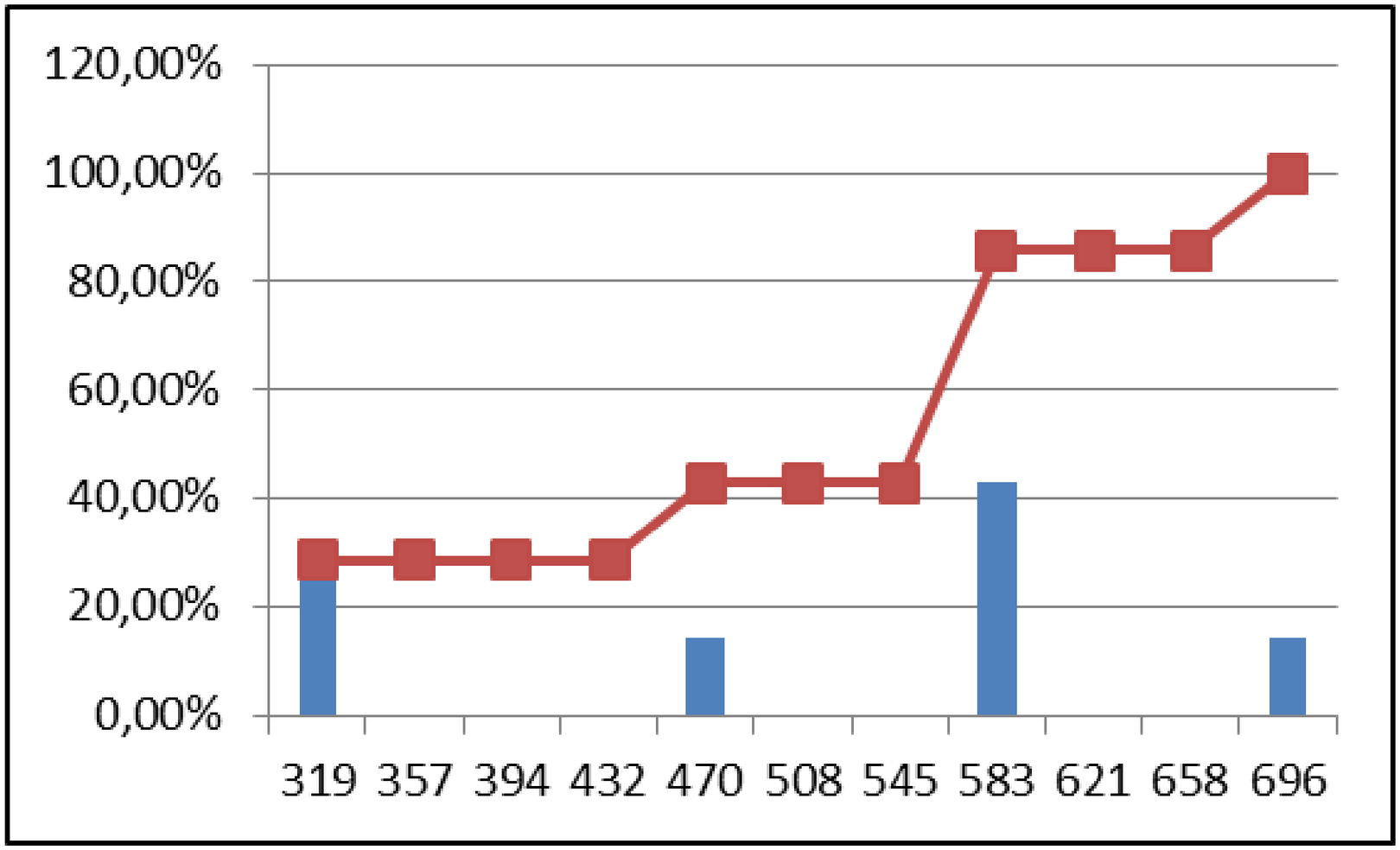
1. Divide the project scope into epics.
2. Randomly sample one of the epics
3. Analyze how many stories are in the sampled epic. Write down the number of stories.
4. Using formula (1) calculate one estimate of the total number of stories for the project
5. Repeat points 2-4 between 7 and 11 times
6. Using formula (2) calculate the total number of stories for the project

Following is a calculation with data from a real ScrumDo.com project.

When the project finished in the backlog there were 29 epics and a total of 529 user stories.

Random epic selector	Epic #	Number of User Stories inside the epic	Epic's selection probability	Estimated total stories
0,733796	22	19	0,034483	551,00
0,596877	18	16	0,034483	464,00
0,30461	9	24	0,034483	696,00
0,988762	29	19	0,034483	551,00
0,191704	6	11	0,034483	319,00
0,184528	6	11	0,034483	319,00
0,091998	3	20	0,034483	580,00

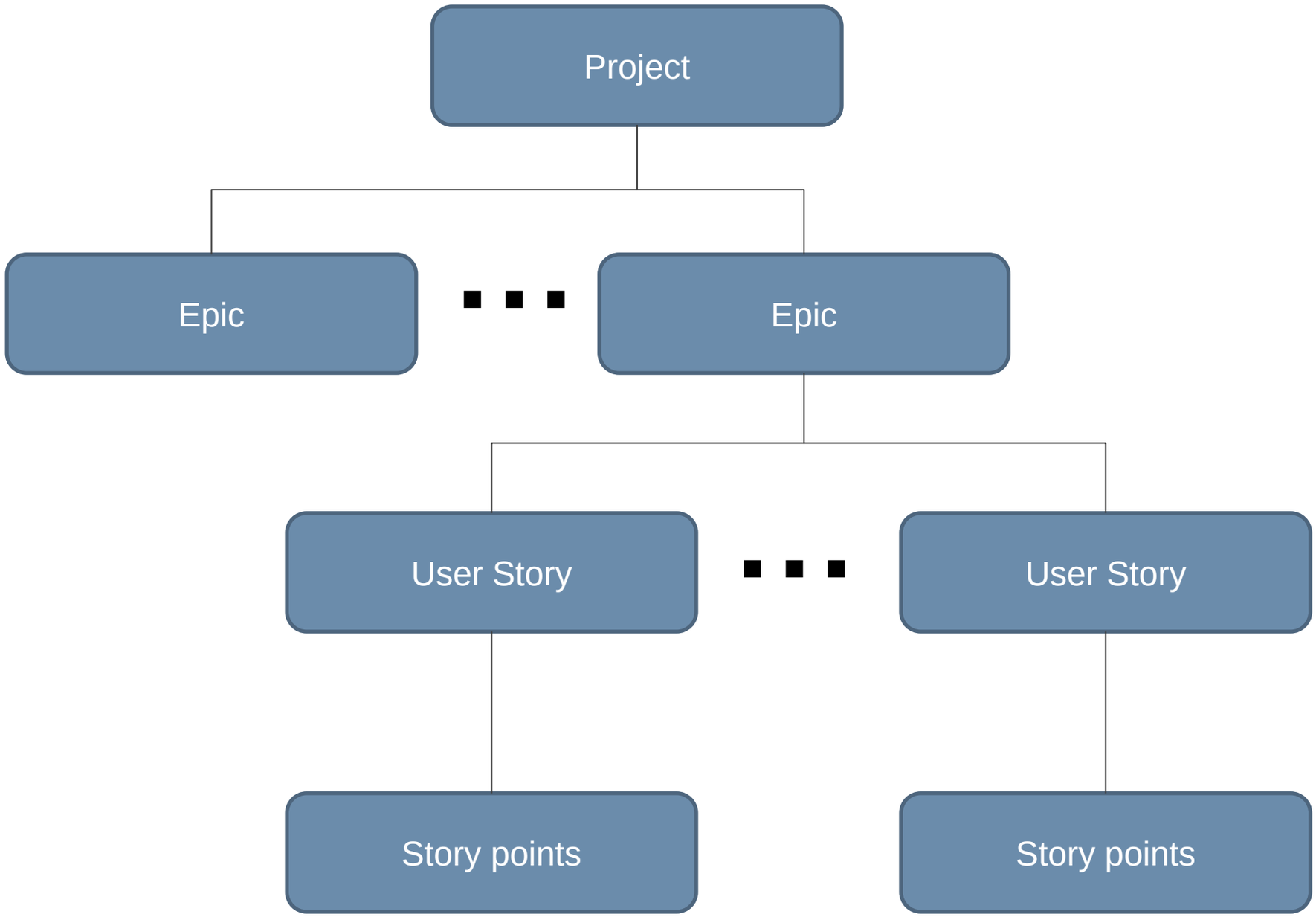
Total Number of tasks for the project

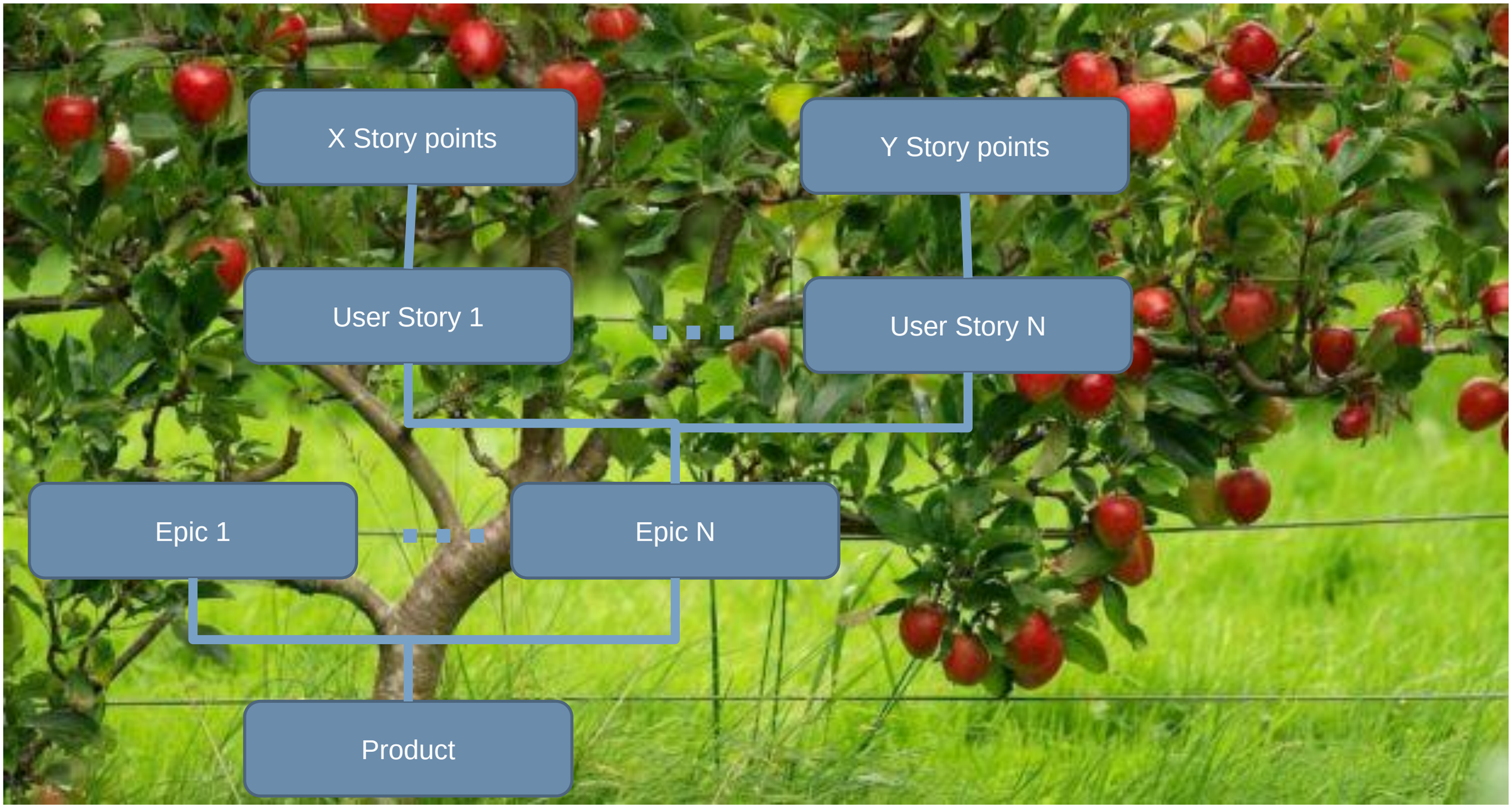


Estimated project size	497
Number of RBS paths	7
SD	53
Median	551
Mode	551

Applying RBS for estimating total Story points in a project

Story points based sizing model




X Story points
Y Story points
User Story 1
User Story N
Epic 1
Epic N
Product

Mapping

Product	Trunk
Epic	Branch
User Story	Terminal Shoot
Story points per story	Number of Fruit on the Shoot

Estimate of the of total story points for a project

$$\hat{X}_i = \frac{\left(\begin{array}{c} Story\ points \\ of\ the\ sampled \\ User\ story \end{array}\right)}{\left(\frac{1}{\begin{array}{c} Number\ of \\ Epics \\ in\ the\ project \end{array}}\right)\left(\frac{1}{\begin{array}{c} Number\ of \\ User\ stories \\ in\ the\ sampled\ Epic \end{array}}\right)} \quad (1)$$

Where:

$\hat{X}_i$ is an unbiased estimator of the population total of the of story points for the project.

Total story points for the project

$$\hat{X} = \frac{1}{m}\sum_{i=1}^{m} \hat{X}_i \quad (2)$$

Where:

$\hat{X}$ is an unbiased estimator of the total story points for the project.

m is the number of estimates done

Algorithm

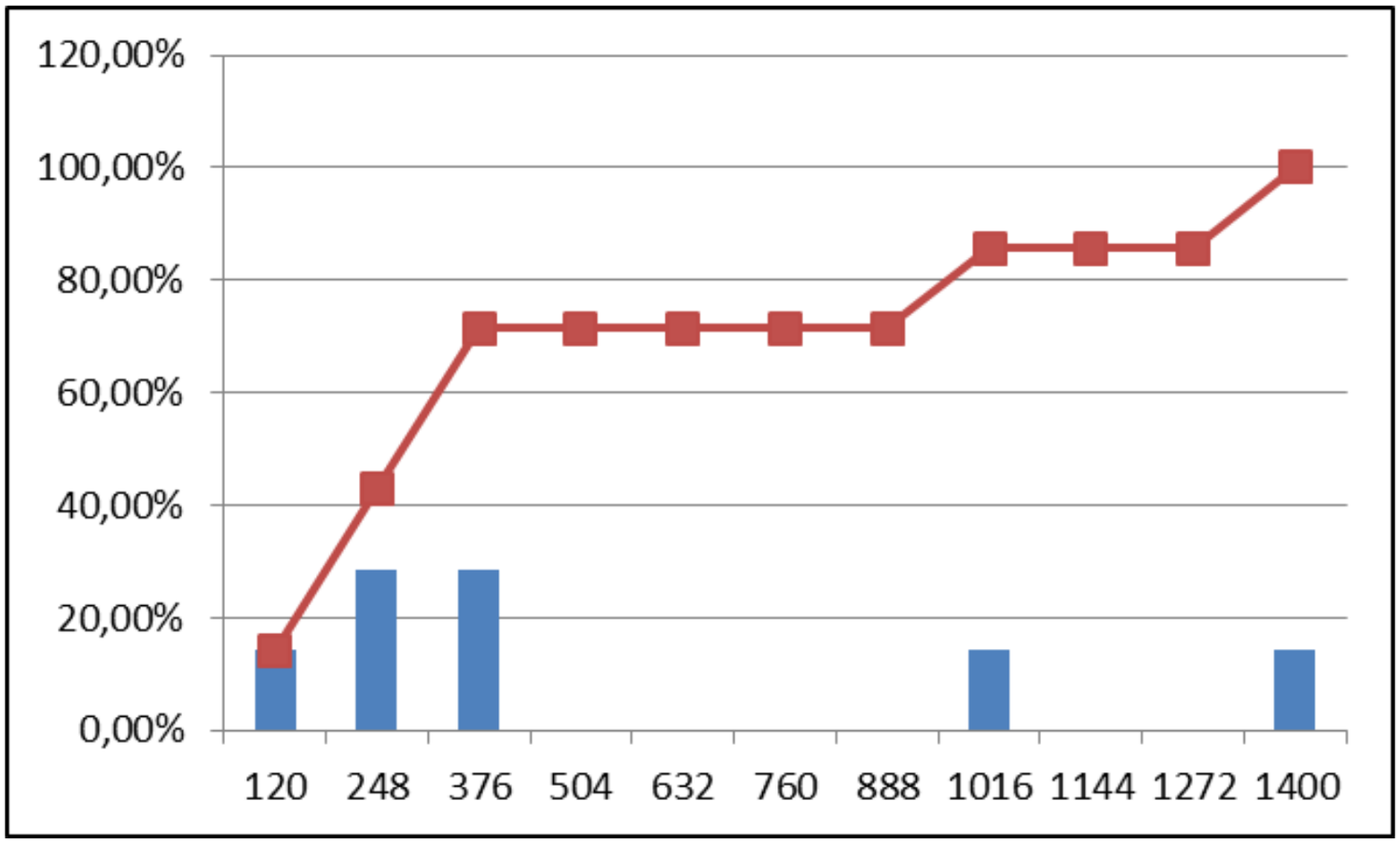
1. Divide the project scope into epics.
2. Randomly sample one of the epics
3. Analyze how many stories are in the epic. Write down the number of stories.
4. Randomly sample one of the stories of the epic from p.2
5. Estimate the story points for the story from p.4
6. Using formula (1) calculate one estimate of the total story points for the project
7. Repeat points 2-6 between 7 and 11 times
8. Using formula (2) calculate the total story points for the project

Following is a calculation with data from a real ScrumDo.com project.

When the project finished there were delivered 20 epics, 176 user stories and a total of 573,5 story points.

Random epic selector	Epic #	Number of User Stories inside the epic	Random story selector	Selected user story	Story points for the selected story	Epic's selection probability	Story's selection probability	Conditional selection probability	Estimated total story points
0,091236	2	14	0,57786827	313067	5	0,05	0,071429	0,0035714	1400,00
0,694128	14	10	0,29687134 2	307842	1	0,05	0,1	0,005	200,00
0,711787	15	13	0,21917813 5	302447	1	0,05	0,076923	0,0038462	260,00
0,623319	13	6	0,34026452 4	308115	1	0,05	0,166667	0,0083333	120,00
0,893093	18	12	0,21872392 6	308016	1	0,05	0,083333	0,0041667	240,00
0,34069	7	8	0,00535048 1	305382	2	0,05	0,125	0,00625	320,00
0,622925	13	6	0,89401048 9	325545	8	0,05	0,166667	0,0083333	960,00

Total Story points for the project

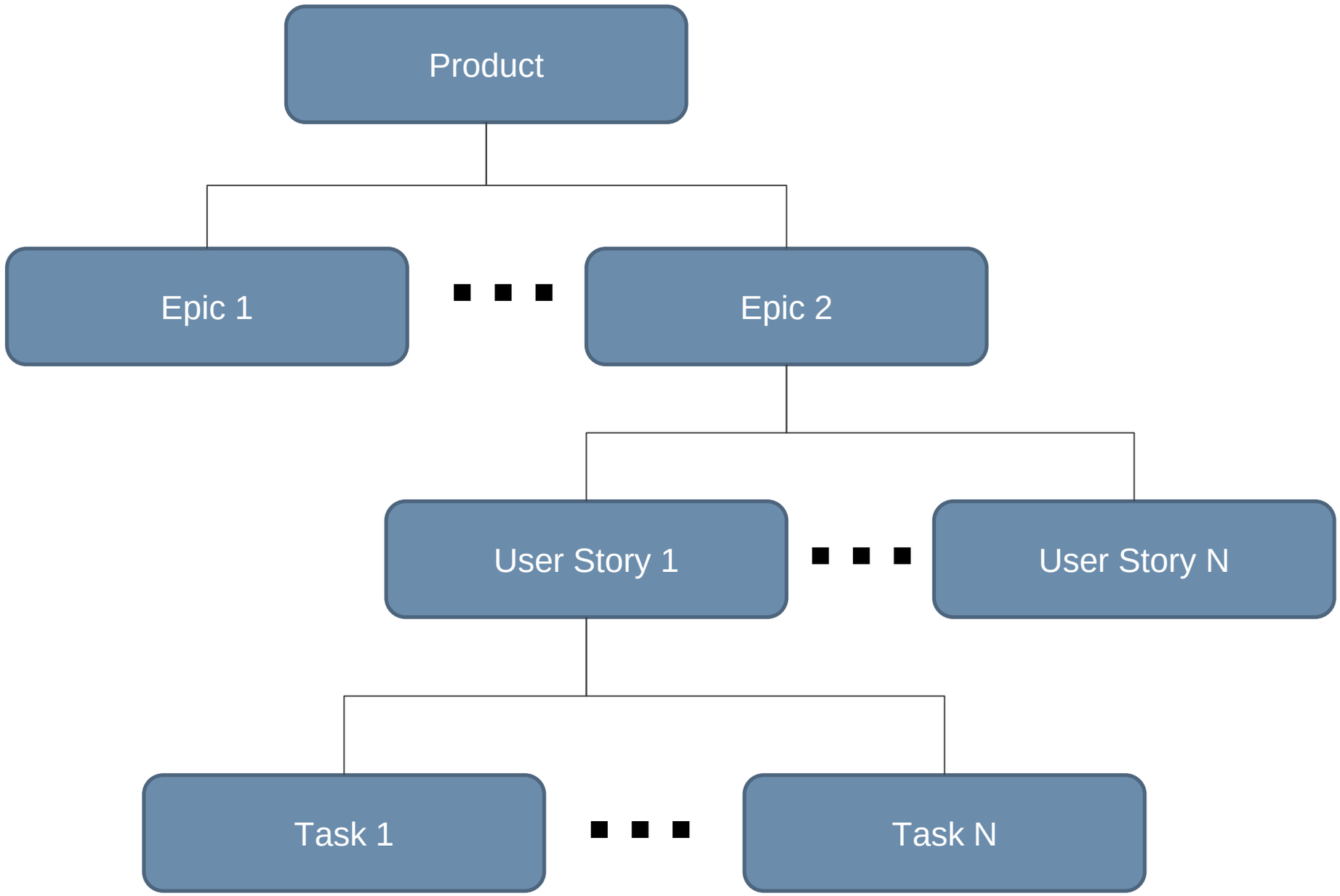


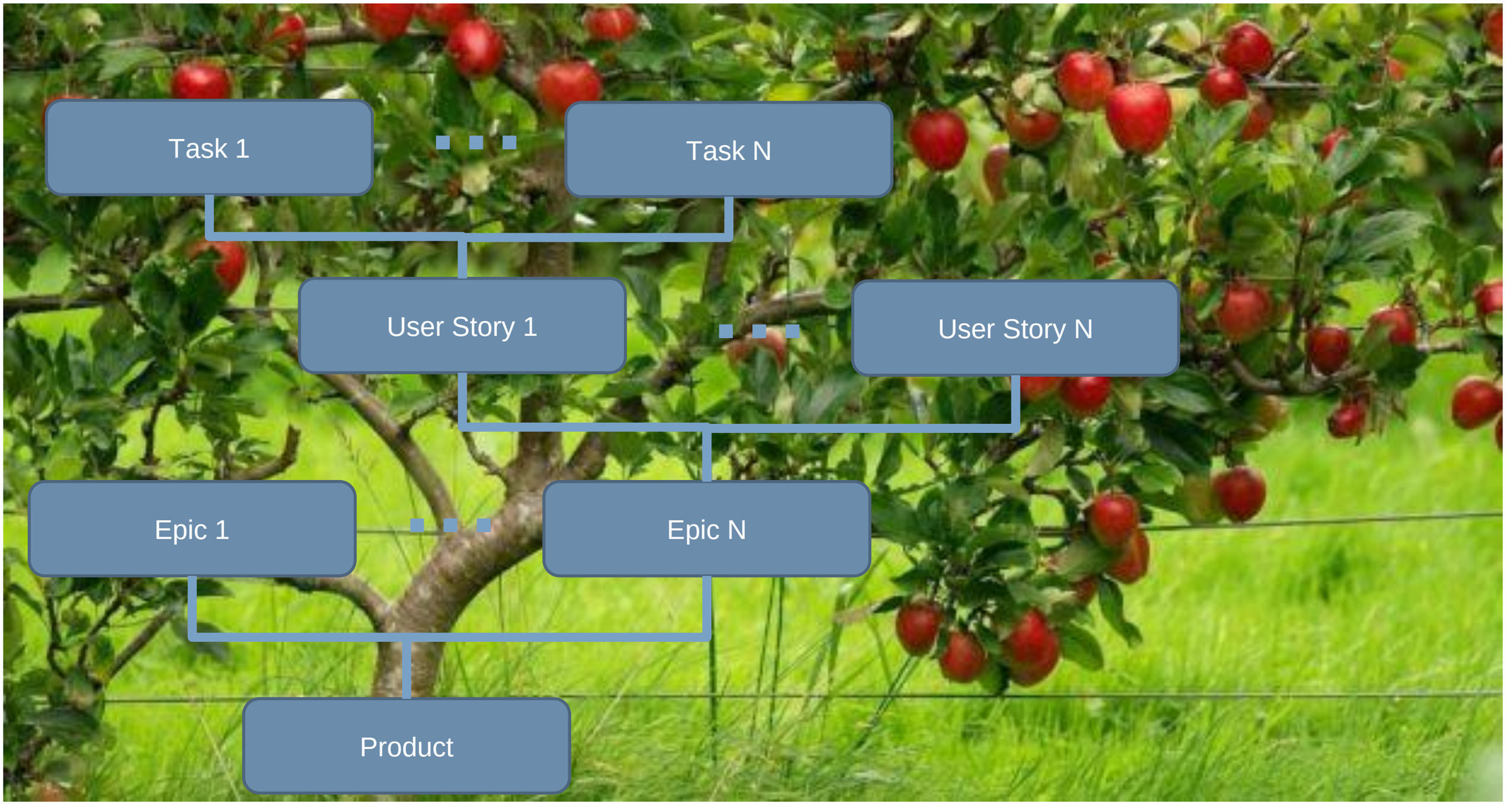
Estimated project size	500
Number of RBS paths	7
SD	183
Median	260

Applying RBS for estimating total number of tasks in a project

When we size each user story in the number of tasks then the project size is the total of all tasks.

Tasks based sizing model




Task 1
Task N
User Story 1
User Story N
Epic 1
Epic N
Product

Mapping

Product	Trunk
Epic	Branch
User Story	Terminal Shoot
Number of tasks per User story	Number of Fruit on the Shoot

Estimate of the of total number of tasks for a project

$$\hat{X}_i = \frac{\left(\begin{array}{c}Number\ of\ tasks\\ in\ the\ sampled\\ User\ story\end{array}\right)}{\left(\frac{1}{\begin{array}{c}Number\ of\\ Epics\\ in\ the\ project\end{array}}\right)\left(\frac{1}{\begin{array}{c}Number\ of\\ User\ stories\\ in\ the\ sampled\ Epic\end{array}}\right)} \quad (3)$$

Where:

$\hat{X}_i$ is an unbiased estimator of the population total of the of story points for the project.

Total number of tasks for the project

$$\hat{X} = \frac{1}{m}\sum_{i=1}^{m} \hat{X}_i \quad (4)$$

Where:

$\hat{X}$ is an unbiased estimator of the total number of scenarios for the project.

m is the number of estimates done

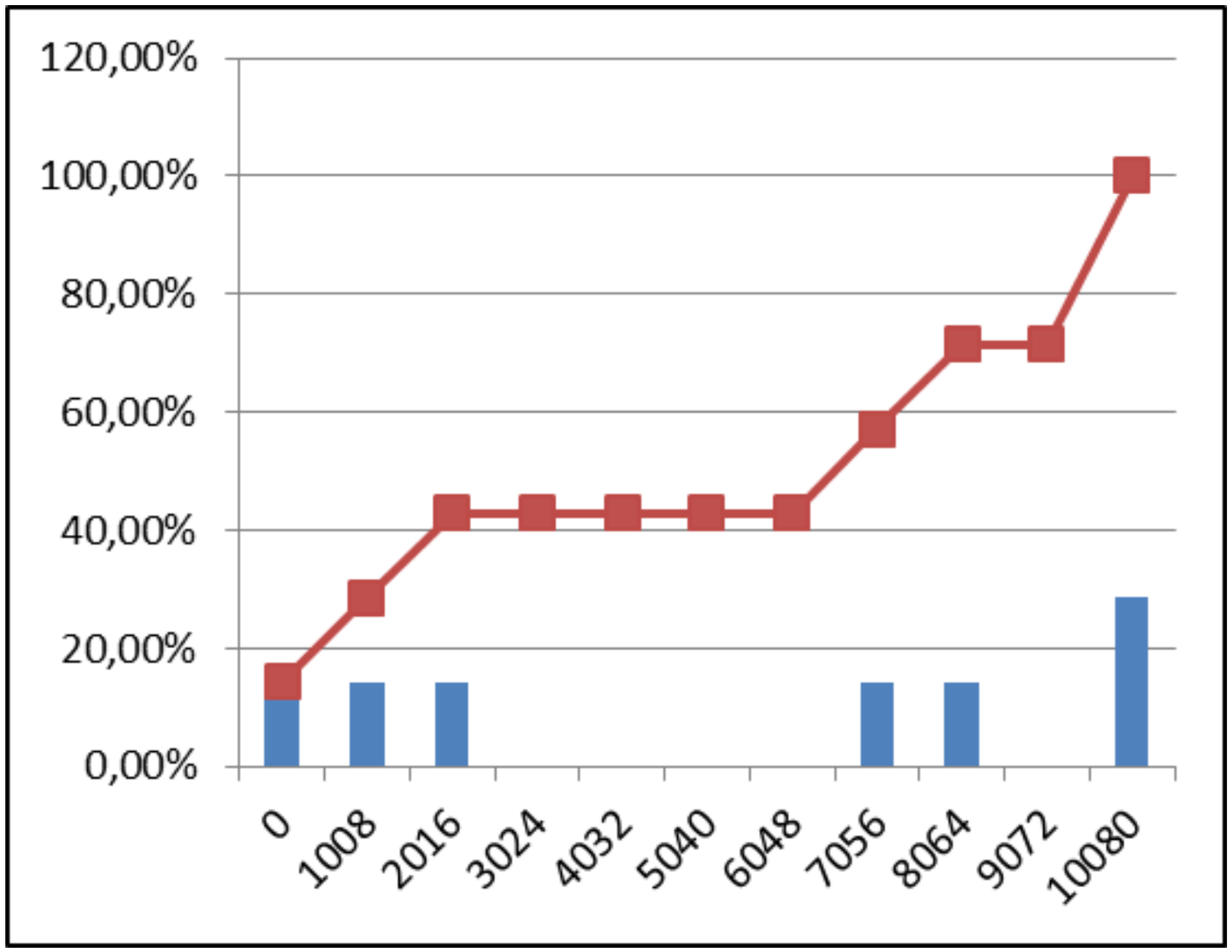
Algorithm

1. Divide the project scope into epics.
2. Randomly sample one of the epics
3. Analyze how many stories are in the sampled epic. Write down the number of stories.
4. Randomly sample one of the stories of the epic from p.2
5. Establish the tasks for the story from p.4
6. Using formula (3) calculate one estimate of the total number of tasks for the project
7. Repeat points 2-6 between 7 and 11 times
8. Using formula (4) calculate the total number of tasks for the project

Following is a calculation with data from a real project. When the project finished in the backlog there were 15 epics, 720 user stories and a total of 5591 tasks.

Random epic selector	Epic #	Number of User Stories inside the epic	Random story selector	Selected user story	Tasks for the selected story	Epic's selection probability	Story's selection probability	Conditional selection probability	Estimated total tasks
0,887642	14	42	0,649722871	545769	12	0,066667	0,02381	0,0015873	7560,00
0,763994	12	51	0,017087888	506420	8	0,066667	0,019608	0,0013072	6120,00
0,897303	14	42	0,57181417 8	541008	1	0,066667	0,02381	0,0015873	630,00
0,542088	9	37	0,559320969	544703	2	0,066667	0,027027	0,0018018	1110,00
0,510646	8	48	0,360797137	527216	14	0,066667	0,020833	0,0013889	10080,00
0,457058	7	46	0,892151817	564853	14	0,066667	0,021739	0,0014493	9660,00
0,736139	12	51	0,972991924	567925	0	0,066667	0,019608	0,0013072	0,00

Total Number of tasks for the project



Estimated project size	5023
Number of RBS paths	7
SD	1652
Median	6120

Conclusion

- RBS is a forecasting technique for sizing software projects without prior identification, analysis and sizing of every single user story. Project size may be measured in story points, number of tasks.
- By running RBS on past data from actual projects, we found that the RBS would have estimated the same size without all the usual effort.
- RBS helps us reduce uncertainty regarding “how much” software needs to be developed when we have to make portfolio related decisions, provide quotations on prospect projects etc.

@dimiterbak

Dimitar Bakardzhiev is the Managing Director of Taller Technologies Bulgaria and an expert in driving successful and cost-effective technology development. As a Lean-Kanban University (LKU)-Accredited Kanban Trainer (AKT) and avid, expert Kanban practitioner, Dimitar puts lean principles to work every day when managing complex software projects with a special focus on building innovative, powerful mobile CRM solutions. Dimitar has been one of the leading proponents and evangelists of Kanban in his native Bulgaria and has published David Anderson's Kanban book as well as books by Eli Goldratt and W. Edwards Deming in the local language.