A Multi-Team, Full-Cycle, Product-Oriented

Scrum Simulation with LEGO Bricks

the Small & Medium Business Edition

Can be adapted to teach other iteration-based Agile frameworks.

Original paper published in Feb, 2009
Authored by Alexey Krivitsky

Current version 2.0, Oct 2011
info@lego4scrum.com

PREFACE
- WHY A LEGO SIMULATION?
- ACKNOWLEDGEMENTS AND APPRECIATIONS
- CURRENT VARIATION
- LICENSING OF THIS WORK
- WEB COMMUNITY AND TRANSLATION PROJECT

THE GAME
- DURATION, GROUP SIZE, MATERIALS
- ROLES
  - Product Owner
  - Scrum Masters
  - Team Members
  - Testers – Optional Role
  - Allow No Observers
- WHAT TO OBSERVE
  - Behaviors
  - Communication Styles
  - Broken Process
- STAGES OF THE GAME
  - PRE-GAME: Organizing Teams
  - PRE-GAME: Project Chartering
  - PRE-GAME: Building the Backlog
  - PRE-GAME: Estimating
    - The fastest technique – swimlane sizing
    - Planning poker with multiple teams
  - GAME: Sprint Planning
  - GAME: Sprinting
  - GAME: Reviewing
  - GAME: Release Cycle
  - POST-GAME: Debriefing
- VARIATIONS
  - Adding Fluctuations
- Enterprise Scrum
- Have your variation? Let us know
- THANK YOU!

PREFACE

WHY A LEGO SIMULATION?

Over the last couple of years I have co-trained a dozen Scrum classes both certified and not. All of those classes had different simulation sessions, but I've always felt like there should be better ones.

Below I am listing the features that a minimal game on Scrum should have, in my opinion.

1. OPEN BACKLOGS THAT TRIGGER IDEATING over DETAILED INSTRUCTIONS TO FOLLOW

We want to start a game with an open backlog - an invitation for collaboration between a customer and teams.

Backlogs can be prepared by trainers, but they should not be closed and precise like "do this, then do that." This will sound like "good old" command and control.

We want to teach and demonstrate a totally different kind of relationship between customers and teams.

2. MINDFUL PRODUCT DEVELOPMENT over A SERIES OF TASKS TO ACCOMPLISH

We need to teach product development, not micromanagement on a task level.

Hence, backlogs or instructions should not be composed of a series of tasks, rather they should be a vision of a product – a big thing teams are to build.

3. TEAMS COLLABORATING TOWARD COMMON SUCCESS over COMPETITION FOR SCORE

The game should be scalable to suit classes of 20 people or more. Which, of course, will result in splitting the group into teams. This should be used as a chance to practice skills of inter-team collaboration.

This needs to be done deliberately, as without specific instructions, it is natural for the teams to start competing.

4. USEFUL METRICS TO ASSESS BENEFITS OF AGILE over FIGURES THE TRAINERS ASKED TO COLLECT

All metrics trainers ask participants to collect need to have an obvious benefit for the teams, as the games need to teach them to own their process.

5. CONTINUOUS IMPROVEMENTS over WINNING OR LOSING THE GAME WITH ONE TRY

The game should be designed so that the teams get multiple tries at it. Each session generates lessons learned and helps them to figure out better processes.

ACKNOWLEDGEMENTS AND APPRECIATIONS

In early 2009 Mykola Gurov who helped me realize the potential of LEGOs as an "API"[1] for product development simulations.

Later that year, I created an early version of the game called "LEGO for extended SCRUM simulation" after discussing ideas with William Wake, Jurgen De Smet, Yves Hanoulle, and Xavier Quesada Allue.

Since the first publication on Scrum Alliance website, I have received dozens of emails with appreciation for this work. Now I'd like to use this chance, in turn, to thank everyone who has contacted me to share their ideas and experiences with running the simulations:

Gerry Kirk, Tim Yevgrashyn, Steve Rogalsky, Andriy Yevtushenko, Geoff Watts, Laurent Godé, Radu Davidescu, Martine Devos, Jo Newcombe Cook, Jakob Frandsen Martin Muntzing, Ola Ellnestam, Dusan Kocurek, Danny (Danko) Kovatch, Gustavo Quiroz, Jukka Lindström, Eduardo Bregaida, and Nathaniel Cadwell.

Special thanks to Robin Dymond and Sergey Dmitriev for letting me run this game in their Certified Scrum Master classes.

CURRENT VARIATION

Since the first paper was published in 2009, dozens of trainers have tried this game. The current, improved, version of the simulation described in this paper reflects feedback and observations made.

LICENSING OF THIS WORK

[1] Not sure what "API" means? Check out http://en.wikipedia.org/wiki/Application_programming_interface

WEB COMMUNITY AND TRANSLATION PROJECT

We decided to create a place where people interested in teaching Scrum with LEGOs could come and collaborate.

www.lego4scrum.com - Join the community, and help us spread the word.

One of the ongoing projects of the community is to translate this paper into world languages. Check out the current status, and consider helping us out. We really appreciate your effort.

THE GAME

DURATION, GROUP SIZE, MATERIALS

It is proven that the game can be adapted to fit the particular needs of trainers and serve different sizes of audience.

A "standard" game is described below, but you are encouraged to adopt it to fit your particular needs.

Timing: 100-120 minutes

- 100 minutes - when using fast team estimation techniques
- 120 minutes - when using planning poker or other estimation tools

Group Size: 4-25 people

- Ideal is 2-3 teams of 4-6 people (gives 8-18 people)
- Can be extended with Scrum Masters

LEGO Boxes: a LEGO box for a team of 4-6 people

- I use "Basic Brick Set #6177"[2]
- It takes four boxes for 20 people

Stationary: standard training package

- Stickers, flip chart paper sheets, markers
- Planning poker cards (or hand-made ones)

[2] Visit the online LEGO store: http://shop.lego.com/en-US/LEGO-Basic-Bricks-Deluxe-6177

Room Setup: a table for each team of 4-6 people

- Extra space (a table or a floor spot) for the integrated product is handy

ROLES

Product Owner

As the trainer, I play the role of the Product Owner."

The goal is to illustrate how Product Owners behave, what they typically expect and require, which team behaviors they appreciate and which they don't.

Scrum Masters

This game can be played without dedicated Scrum Masters.

Sometimes I do have Scrum Masters in the game by inviting co-instructors. Another option would be to ask the team to select Scrum Masters.

By having skillful facilitators playing Scrum Masters - who are constantly focused on the process, and a dedicated trainer - playing the business role, makes the game much more natural and an easy-going activity.

Team Members

Other students are team members.

Testers – Optional Role

You can have testers on the teams. Their main responsibility would be to help their teams document agreements on requirements and designs in order to perform acceptance testing.

The downside, which I've experienced, is that rather than building LEGOs, the testers were observing quality. As the goal of such games is to learn by doing, I think it makes sense to encourage everyone to get engaged in the building process.

Allow No Observers

This game is so much fun playing that the observers will be losing more than they are gaining – my opinion. Otherwise, I'd love to hear good stories on this from *you*.

WHAT TO OBSERVE

Behaviors

From my observations, certain behaviors people demonstrate in games are projections of working habits. And under stress people will tend to fall back to their natural behaviors.

This game is deliberately designed to be *stressful,* so it might expose bad habits that could harm real Agile adoptions. My goal as a trainer is to point them out to the group and turn them into learning points and warnings to keep an eye on.

Communication Styles

Watch out for: “managers”, “dictators”, “loud voices,” and similar projections. This is a fruitful area for debriefing and topics for personal coaching.

Broken Process

Keep an eye on parts of the process the teams don’t do well.

For example, during requirements discussion the teams might not ask as many clarifying questions as they need.

Most likely they also have this issue, or will have this issue, in the particular area of a real project. Making this visible on debrief is one way of handling this.

STAGES OF THE GAME

The simulation has natural three parts: **pre-game**, **game**, and **post-game** or debrief.

Pre-game

- Organizing teams
- Defining the Process
- Project chartering
- Building the Backlog
- Estimating

Game

- Sprint Planning
- Sprinting
- Sprint Reviewing

Post-game

- Debriefing

PRE-GAME: Organizing Teams

Will take 5 minutes.

There is no reason why this activity cannot be a part of the game – a learning process.

Striving to demonstrate self-organization in action I usually ask the team to self-organize in groups of 4-6 people and allocate working space

This is a good warm-up activity since it might require moving tables around and cleaning them up.

PRE-GAME: Project Chartering

Will take 10 minutes. You are 5 minutes into the game now.

As a trainer who is playing the Product Owner I need to communicate the following messages:

1. All teams will be building a single product – they are not competing, rather they work for the same vendor.
2. The product is a CITY with certain features.
3. The main building elements are LEGOs, though any other material can be used in addition.
4. I am the main decision maker of the product – it is *my* city.
5. I will be involved in the development process by being available to answer questions and provide feedback.

Running this activity as *collaborative chartering* might be a good way.

My goal is to make sure the teams practice Scrum by building "products" with LEGO bricks. Now the tricky question is how to combine the two roles: a Product Owner (who is not owning the process) and a class trainer (who has an interest in running it with Scrum).

There are several ways I've tried doing this:

1. **Changing hats – explain Scrum rules to the team**
   I explicitly state whether I am currently a Product Owner or a trainer so that people are not confused.
2. **Playing a newbie Product Owner – let the team sell you Scrum**
   Most often I play a Product Owner who doesn't know much about Agile or Scrum and after presenting my vision of a city I ask the team to help us design out the process that suits.

Personally, I like this second approach more since it helps amplify learning from the class and let students practice articulating Agile values.

PRE-GAME: Building the Backlog

Will take 15 minutes. You are 15 minutes in the game now.

Once you have sketched a charter and agreed on the process, it is the time to share the features of the city.

I usually do this by showing the showing a team pre-prepared set of sticky notes hanging on a sheet of flip-chart paper.

Usually it includes the following items:

- One storey building (several of these, one per sticky note)
- Two storey building (several)
- Shop
- School
- Church
- Hospital
- Kindergarten
- Bus stop
- Intersection (can be drawn)
- Park (can be drawn)
- River (can be drawn)
- Bridge

Some of the items can be drawn on flip chart paper, and then the LEGOs are on top.

> Here you can get creative and build something more entertaining rather than a simple city.
>
> Once we've played this game with a start-up team, so we were building a "silicon village".
>
> It obviously had some other items to be built, like a presentation hall with an iPad (presenting a screen), a bunch of open co-working areas in the city, a secure building for web servers, and a monument to a hero entrepreneur (a fancy monument on rails). That was fun!

While presenting the backlog I briefly explain what I think each of the items may look like. And I try to postpone discussions until later.

PRE-GAME: Estimating

Will take up to 20 minutes. You are 30 minutes into the game now.

Estimations. The hardest part, somehow.

I may want to:

1. Drop estimates (as agile gurus would recommend)
2. Do it faster and simpler

3. Spend a bit of time on practicing planning poker

> RT @RonJeffries: "Every year there is new estimation technique. The real Agile approach would be to throw estimations out." - @agilemanager [YES!]

Based on how much time we have I can decide to do the simplest technique or poker.

The fastest technique – swimlane sizing

I've learned this term from www.theagilepirate.net. Apparently, I do it in a less sophisticated manner.

Check out a sketch below.

Based on the triangulation concept[3] and swimlane sizing[4] we simply arrange columns to mark different story sizes (1 2 3 5 8 13 if you prefer Fibonacci - a little taste of science is always good), and ask students to drag stories to columns that represent story size. We do it in groups or altogether.

This activity can also be run silently.

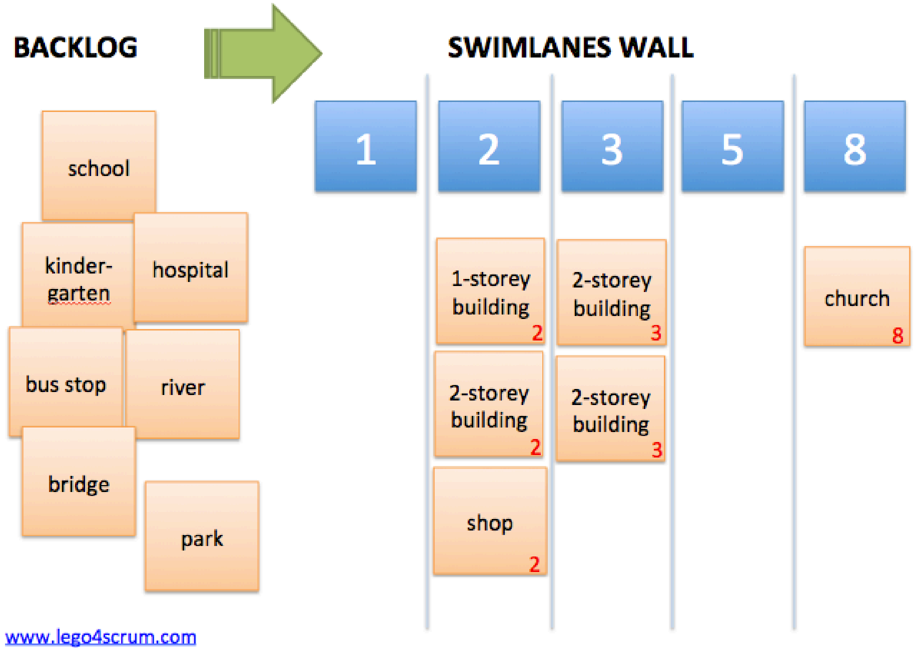


Figure 1: Swimlanes for group estimations

[3] Triangulation and other concepts of Agile estimating and planning, by Mike Cohn http://www.mountaingoatsoftware.com/presentations/85-an-introduction-to-agile-estimating-and-planning

[4] Swimlane Sizing – Complete & Fast Backlog Estimation http://theagilepirate.net/archives/109

If a group is big enough that it cannot fit in front of the board, I ask each team to send a pair. When these pairs are done, the next come, until everyone got a chance to touch the board.

Once done, I ask the group whether this is "good enough" to start with and whether they would like to do *some real* work now.

Planning poker with multiple teams

Running planning poker[5] with multiple teams requires agreeing on sample size as a whole group, firstly.

Agreeing on size is simple: pick up an item which is small and simple enough, but not trivial and assign it "2". Usually everyone agrees to have all one-stored buildings as twos.

Another approach to selecting sample stories to do t-shirt size estimates[6] (XS, S, M, L, XL) and then mark all S-sized stories as "2" and continue with the poker.

I'd like to share hints that help me make multi-team planning poker sessions work:

- Organize swimlanes wall (see sketches above)
- Ask teams to pull stories one by one for estimation.
- Ask teams to attach details to each story when they get clarifications from Product Owner (as it may be another team building the item).
- Actively encourage and appreciate team members asking clarifying questions which help define size.
- Once estimated, a story needs to be put on the wall, so that other teams can benefit from the new information.
- When done ask people to come up to the wall and make a sanity check with necessary changes (in my experience changes are rarely needed).

> If the teams do not know much about planning poker, it is worth doing a test run so that you can observe them using the technique correctly. I usually ask teams to guesstimate:
>
> "How much does a pint of Guinness cost in the U.K?"

[5] Planning Poker is found by James Grening in 2002 and popularized by Mike Cohn: http://en.wikipedia.org/wiki/Planning_poker

[6] T-shirt sizing http://blogs.msdn.com/b/oldnewthing/archive/2009/05/12/9605143.aspx

This implies asking questions on points to be used, place and date of buying it – acts as a good warm-up for estimating stories.

Interestingly, both techniques: swimlanes and planning poker provide the necessary precision for release planning as it has proven by hundreds of release burn-down charts.

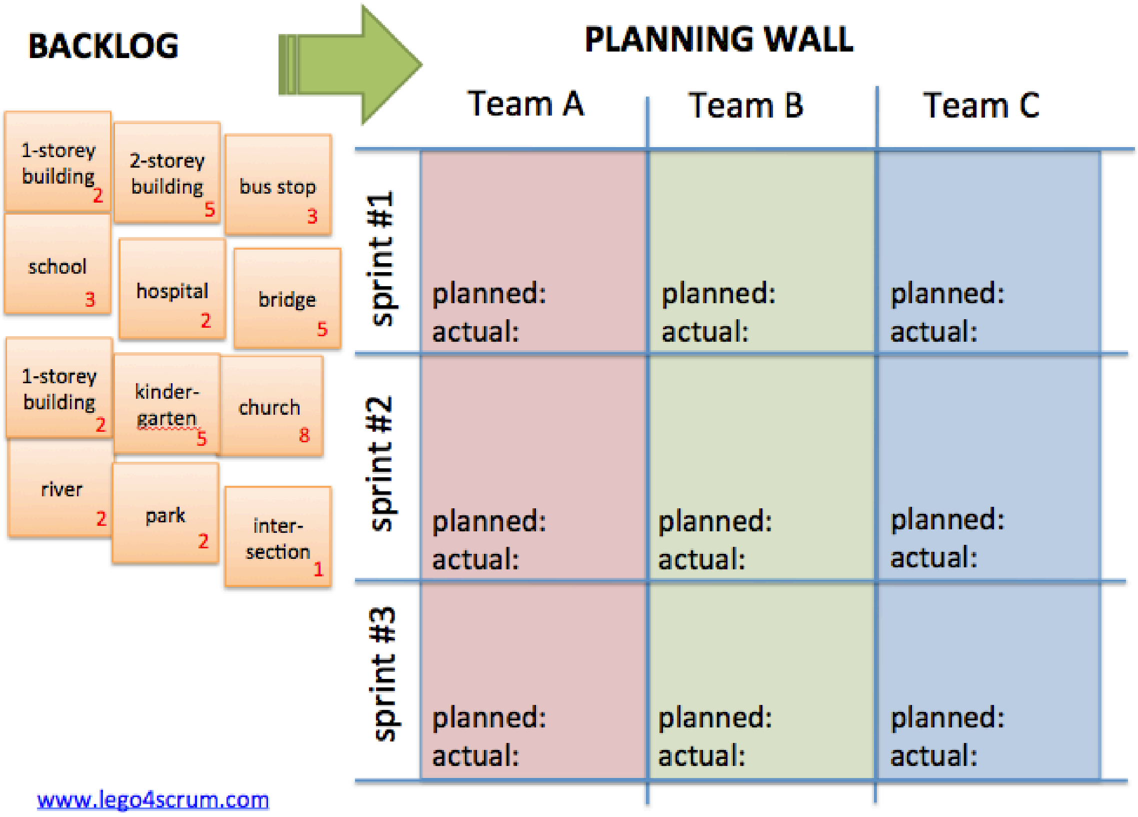
GAME: Sprint Planning

You are 50 minutes into the game now.

(And zero things are built! Does it prove enough that estimations are wasteful?)

Now when the stories are estimated, you have to move them from the Swimlanes Wall back to the Backlog.

Since we would like to make sprint planning highly visible, we are building a special Planning Wall that accumulates all teams' plans for all sprints in the game.



BACKLOG

- 1-storey building 2
- 2-storey building 5
- bus stop 3
- school 3
- hospital 2
- bridge 5
- 1-storey building 2
- kinder-garten 5
- church 8
- river 2
- park 2
- inter-section 1

PLANNING WALL

	Team A	Team B	Team C
sprint #1	planned: actual:	planned: actual:	planned: actual:
sprint #2	planned: actual:	planned: actual:	planned: actual:
sprint #3	planned: actual:	planned: actual:	planned: actual:

www.lego4scrum.com

Figure 2: Multi-team Planning Wall, before planning sprint #1

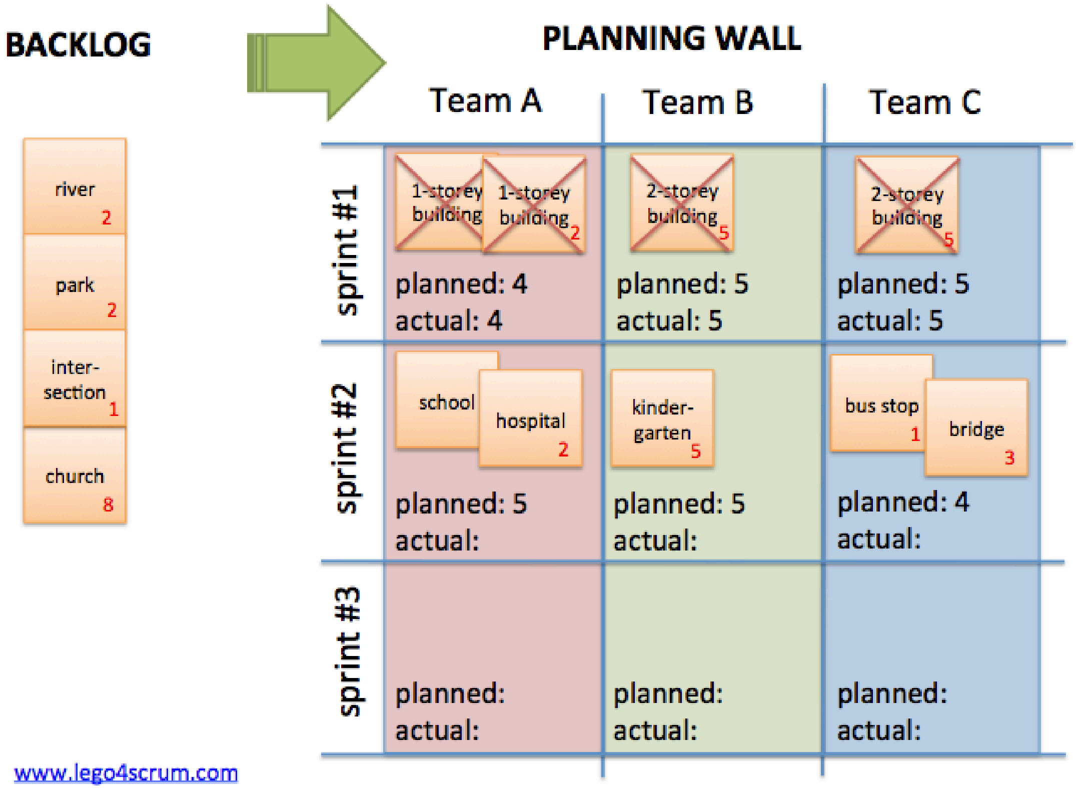


Figure 3: Multi-team Planning Wall, inside sprint #2

We time-box Sprint planning activity to three minutes asking teams to pull stories into their team’s sprint boxes.

When all is done, we ask whether the teams are *un*comfortable enough with their plans to give it a try!

GAME: Sprinting

Will take 7 minutes.

We prefer 7-minutes sprints, as this time is quite enough to build several items with several people collaborating and without too much polishing it.

To make sure students are stressed enough we show a big visible stopwatch from a notebook or an overhead projector:

Figure 4: A stopwatch from www.online-stopwatch.com - timers of different forms, can be used offline.

GAME: Reviewing

Will take 5 minutes.

When the time is up, I make sure students really stop building things and start demanding: "*where is my city?*"

As it has been observed, it is only after the second sprint teams start doing continuous deployments of stories to the demonstration environment (a flip chart paper). So in most cases after the first sprint, no one has ever thought on how to organize the demonstration. Sounds similar to the real life?

I always got comments on debrief that I am playing the kindest Product Owner ever seen. Still, in most cases nothing is accepted after the first sprint, because after I am shown the buildings, I "come to realize":

- I like symmetry.
- "All of the same color" actually meant "solid colors of buildings, but different for each."
- Buildings are either too small, too big, or too diverse.
- Windows of different floors are not lined up.
- <make up your own reasoning>

Unfinished items are brought back to the Backlog from the Planning Wall. Remaining work can be re-estimated, although we rarely update estimates.

Once stories are accepted, the Release Burndown chart is updated by a PO, who makes a clear and loud announcement that the release has to be done in three sprints and now it looks like we will not be able to accomplish all stories.

A few minutes can be spent on retrospecting on the topic of "how can we make it better the next sprint?".

GAME: Release Cycle

Without too much time wasted on discussing failures of the first sprint, which is almost always a disaster, we jump back to Sprint Planning.

I've learned it takes an average three sprints to build 80% of the backlog with anticipated quality, so the full cycle usually looks like this:

1. Sprint #1
    a. Planning – 3 minutes
    b. Sprinting – 7 minutes
    c. Review – 5 minutes

2. Sprint #2
    a. Planning – 3 minutes
    b. Sprinting – 7 minutes
    c. Review – 5 minutes

3. Sprint #3
    a. Planning – 3 minutes
    b. Sprinting – 7 minutes
    c. Review – 5 minutes

Subtotal: 45 minutes

Since the preparations took us about 1 hour (from chartering to estimated backlog), sprints took 45 minutes, it will take about 15 minutes for debrief, **the full game takes 120 minutes.**

Once practiced and with help from co-trainers, who play Scrum Master, it can be done a bit faster.

POST-GAME: Debriefing

It is probably wise to take a short break once the last sprint is reviewed and before jumping into debrief to calm down emotions and have a short rest (Did I tell you the game is designed to be exhaustive? Not only for the teams...)

When gathered back in a circle, we run a facilitated discussion around the following open-ended questions:

- What did students observe?
- How did it feel being on a Scrum team?
- How did the short iterations go?

- How accurate were the estimations (provided the Release Burndown is there)
- What would we have done differently from the beginning, if we had another chance to play the game?
- What was the job of the Product Owner?
- How did it feel after the first sprint when almost all items required re-work?
- What did the Scrum Masters do?
- How will your strategy change, if you know the Product Owner is unavailable during sprints?
- How did inter-team communication go? Were there any dependencies? How were they resolved?
- What did students learn?

VARIATIONS

Adding Fluctuations

Good friends of mine (Askhat Urazbaev and Nikita Filippov) have designed a similar game, which includes randomized fluctuations to the team size and complexity.

Simply, after sprint planning is done and just before sprint start, the teams throw dice that either multiply story point estimates or make some team members gone "sick" for the sprint while the team will have to keep the sprint plan.

The point this game makes is that team collaboration is essential for task balancing during sprints, as things might go differently than was planned.

Enterprise Scrum

I was able to scale the simulation with LEGOs to engage 100 students playing in 12 teams that were building 4 cities simultaneously. It requires quite a few tons of LEGOs but it seems to be a good way to demonstrate enterprise-level Scrum. This deserves another article to cover all rules and settings.

Have your variation? Let us know

We would like to hear your stories, your variation to the simulation – please join us at www.lego4scrum.com and email us your ideas at info@lego4scrum.com

THANK YOU!

Have playful projects!

Alexey Krivitsky
www.lego4scrum.com