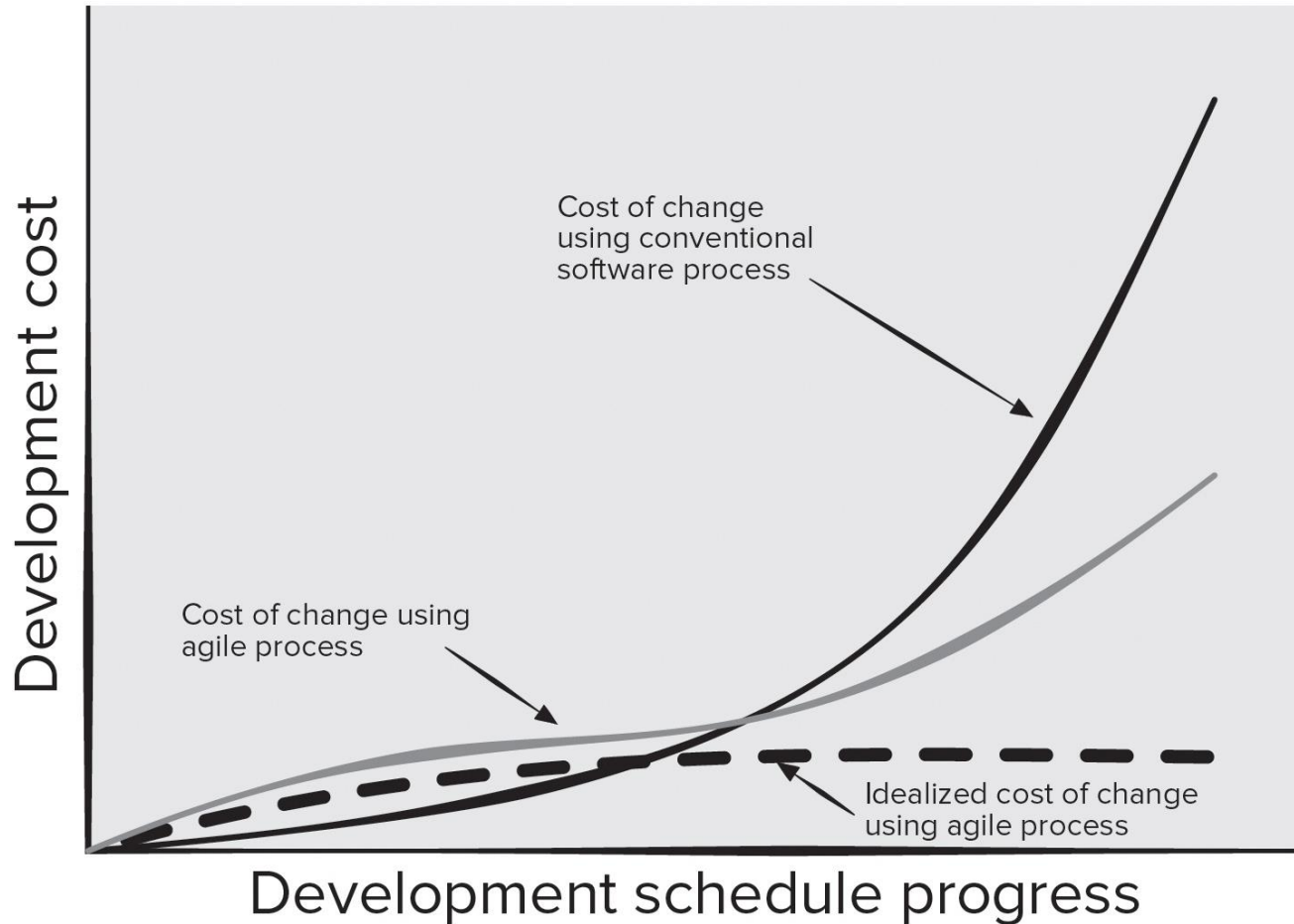


What is Agility?

- Effective (rapid and adaptive) response to change.
- Effective communication among all stakeholders.
- Drawing the customer onto the team.
- Organizing a team so that it is in control of the work performed.
- Rapid, incremental delivery of software.

Agility and Cost of Change

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

What is an Agile Process?

- Driven by customer descriptions of what is required (scenarios).
- Customer feedback is frequent and acted on.
- Recognizes that plans are short-lived.
- Develops software iteratively with a heavy emphasis on construction activities.
- Delivers multiple ‘software increments’ as executable prototypes.
- Adapts as project or technical changes occur.

Agility Principles ₁

- Customer satisfaction is achieved by providing value through software that is delivered to the customer as rapidly as possible.
- Develop recognize that requirements will change and welcome changes.
- Deliver software increments frequently (weeks not months) to stakeholders to ensure feedback on their deliveries is meaningful.
- Agile team populated by motivated individuals using face-to-face communication to convey information.
- Team process encourages technical excellence, good design, simplicity, and avoids unnecessary work.

Agility Principles ²

- Working software that meets customer needs is the primary goal.
- Pace and direction of the team's work must be “sustainable,” enabling them to work effectively for long periods of time.
- An agile team is a “self-organizing team”—one that can be trusted develop well-structured architectures that lead to solid designs and customer satisfaction.
- Part of the team culture is to consider its work introspectively with the intent of improving how to become more effective its primary goal (customer satisfaction).

Scrum Details

- **Backlog Refinement Meeting** Developers work with stakeholders to create product backlog.
- **Sprint Planning Meeting** Backlog partitioned into “sprints” derived from backlog and next sprint defined.
- **Daily Scrum Meeting** Team members synchronize their activities and plan work day (15 minutes max).
- **Sprint Review** Prototype “demos” are delivered to the stakeholders for approval or rejection.
- **Sprint Retrospective** After sprint is complete, team considers what went well and what needs improvement.

Pros

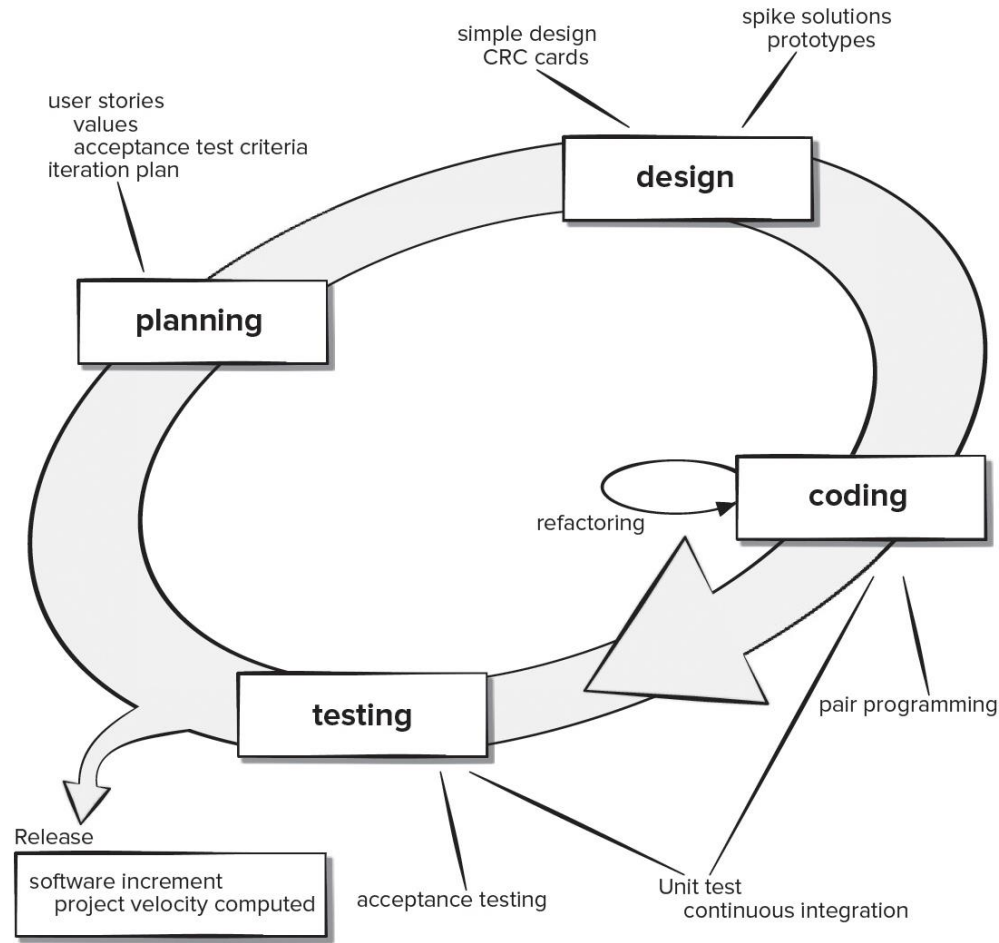
- Product owner sets priorities.
- Team owns decision making.
- Documentation is lightweight.
- Supports frequent updating.

Cons

- Difficult to control the cost of changes.
- May not be suitable for large teams.
- Requires expert team members.

Extreme Programming (XP) Framework

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

XP Details

- **XP Planning** – Begins with user stories, team estimates cost, stories grouped into increments, commitment made on delivery date, computer project velocity.
- **XP Design** – Follows KIS principle, encourages use of CRC cards, design prototypes, and refactoring.
- **XP Coding** – construct unit tests before coding, uses pair.
- **XP Testing** – unit tests executed daily, acceptance tests define by customer.

Pros

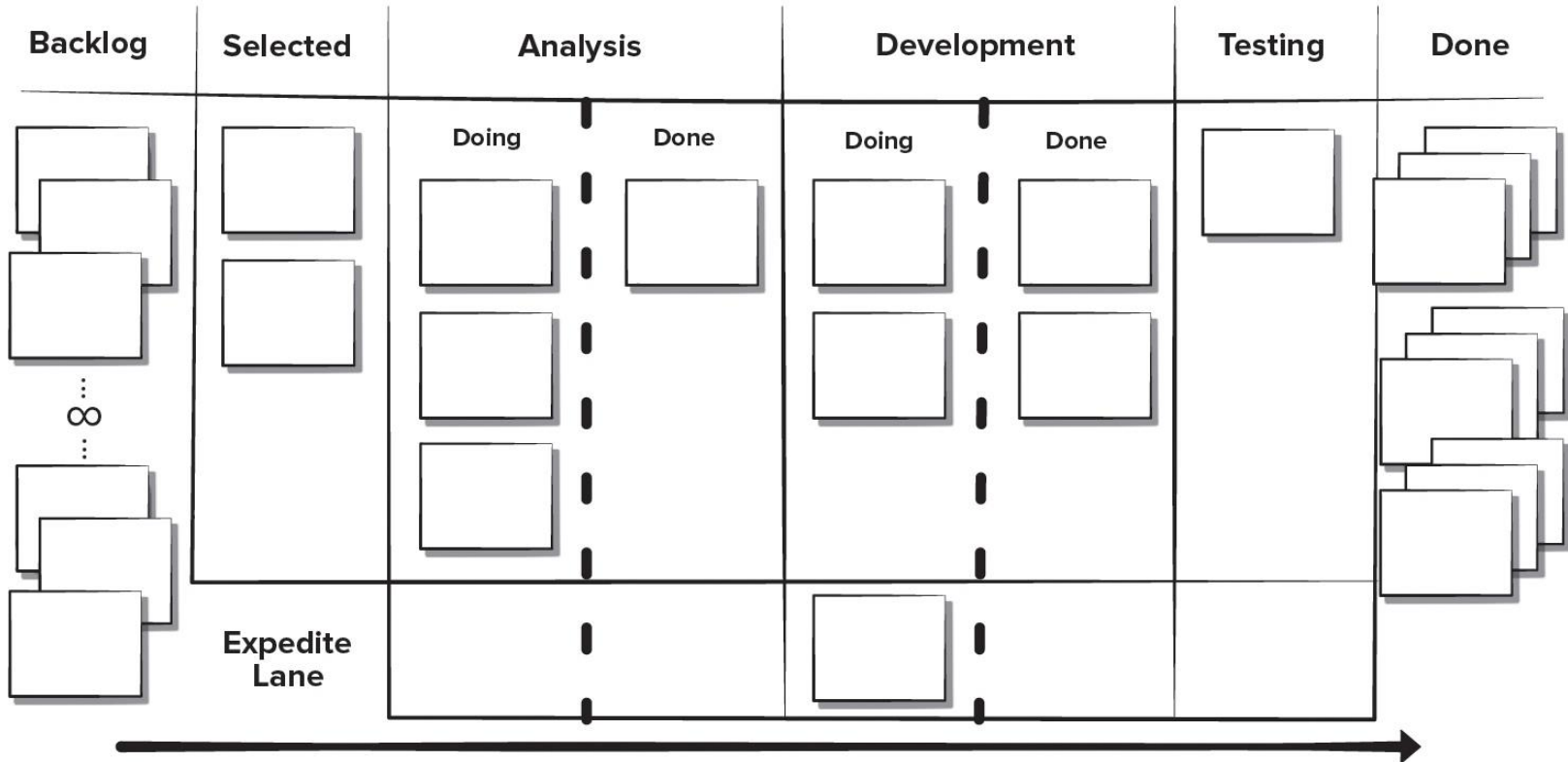
- Emphasizes customer involvement.
- Establishes rational plans and schedules.
- High developer commitment to the project.
- Reduced likelihood of product rejection.

Cons

- Temptation to “ship” a prototype.
- Requires frequent meetings about increasing costs.
- Allows for excessive changes.
- Depends on highly skilled team members.

Kanban Framework

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

Kanban Details

- Visualizing workflow using a Kanban board.
- Limiting the amount of *work in progress* at any given time.
- Managing workflow to reduce waste by understanding the current value flow.
- Making process policies explicit and the criteria used to define “done”.
- Focusing on continuous improvement by creating feedback loops where changes are introduced.
- Make process changes collaboratively and involve all stakeholders as needed.

Pros

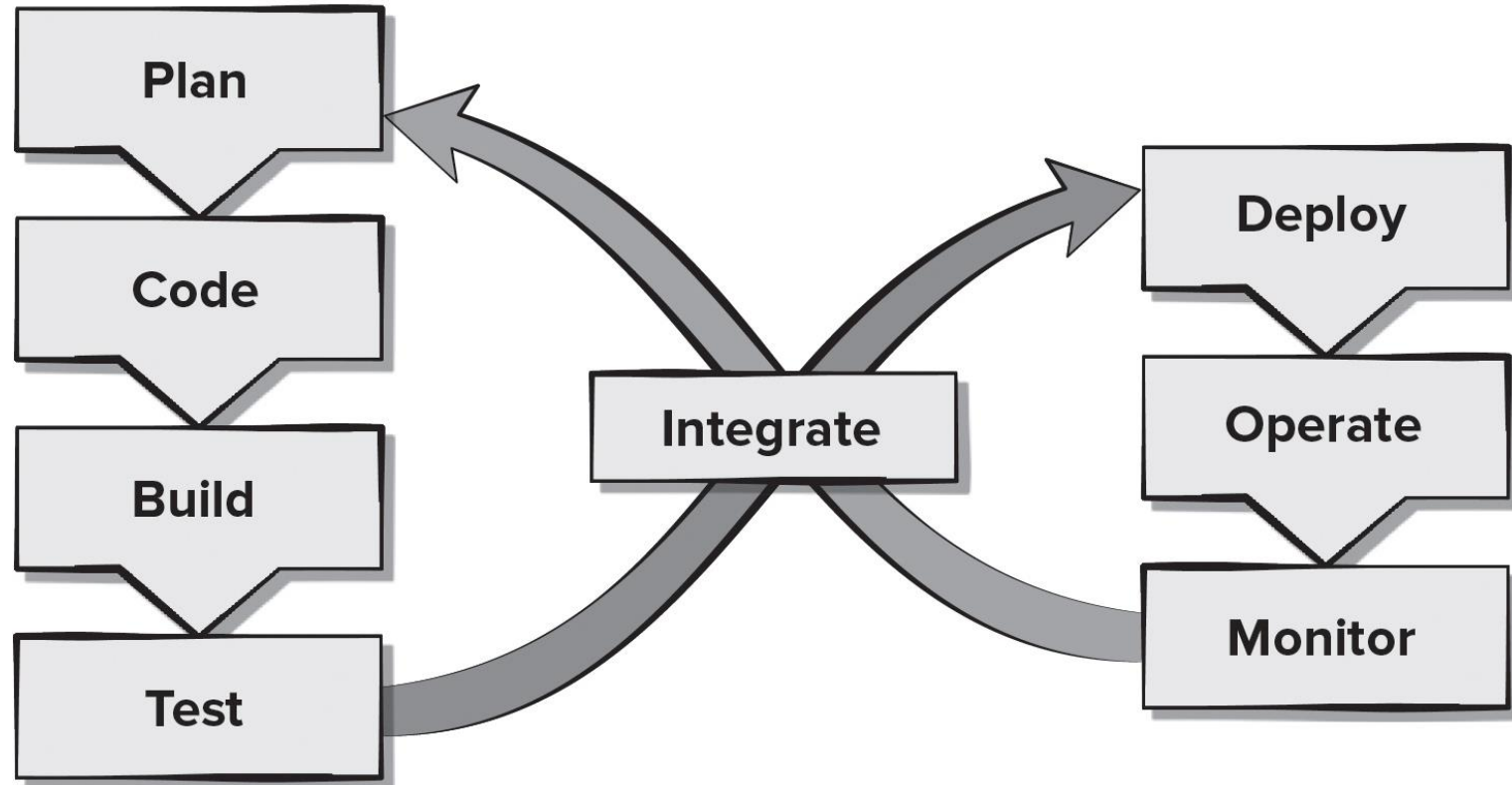
- Lower budget and time requirements.
- Allows early product delivery.
- Process policies written down.
- Continuous process improvement.

Cons

- Team collaboration skills determine success.
- Poor business analysis can doom the project.
- Flexibility can cause developers to lose focus.
- Developer reluctance to use measurement.

DevOps

Copyright © McGraw-Hill Education. All rights reserved. No reproduction or distribution without the prior written consent of McGraw-Hill Education.



[Access the text alternative for slide images.](#)

DevOps Details

- **Continuous development.** Software delivered in multiple sprints.
- **Continuous testing.** Automated testing tools used prior to integration.
- **Continuous integration.** Code pieces with new functionality added to existing code running code.
- **Continuous deployment.** Integrated code is deployed to the production environment.
- **Continuous monitoring.** Team operations staff members proactively monitor software performance in the production environment.

Pros

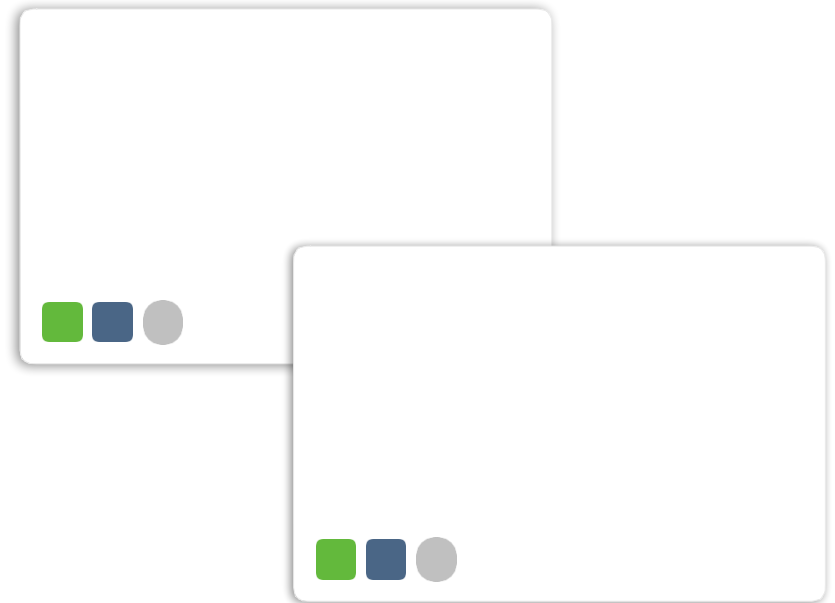
- Reduced time to code deployment.
- Team has developers and operations staff.
- Team has end-to-end project ownership.
- Proactive monitoring of deployed product.

Cons

- Pressure to work on both old and new code.
- Heavy reliance on automated tools to be effective.
- Deployment may affect the production environment.
- Requires an expert development team.

WORKSHOP

Writing User Stories



contents

SECTION

1

user stories

what is a user story?

4

user story template

5

examples: user stories

6

user story checklist

7

why not tasks?

8

SECTION

2

writing acceptance criteria

10

what is acceptance

11

criteria? example:

13

acceptance criteria

acceptance criteria

checklist

What Is A User Story?

definition: user story

A user story is a tool used in agile software development to capture the description of a software feature from an end-user perspective. The user story describes the type of user, what they want and why, A user story helps to create a simplified description of a requirement.

A user story often follows the following 'equation':

As a <type of user>, I want <some feature> so that <reason>

A simple example of this could be:

As an online shopper, I want to add an item to my cart, so that I can purchase it

user story template

WHO are we building it for? Who is the user?	As a <type of user>
WHAT are we building? What is the intention?	I want <some goal or objective>
WHY are we building it? What is the value for the customer?	So that <benefit/value>

examples: user stories

As an internet banking customer
I want to see a rolling balance for my everyday accounts
So that I know the balance of my account after each transaction is applied



As an administrator
I want create other administrators
So that I can delegate tasks



As a marketer
I want create automated email campaigns
So that I can keep evaluators engaged



user story checklist

- ☒ Keep them short
- ☒ Keep them simple
- ☒ Write from the perspective of the user
- ☒ Make the value/benefit of the story clear - what is the reason for the story?
- ☒ Describe **one** piece of functionality. If you have to write **and** break it into 2 stories
- ☒ Write stories as a team
- ☒ Use acceptance criteria to show a MVP



☒ _____

☒ _____

☒ _____

☒ _____

why not just use 'tasks'?

 user stories	 tasks
a user story = the WHAT	the task = the HOW
user stories describe a piece of functionality from the point of view of the user	“what are the activities we need to perform in order to deliver outcomes (user stories)”
divided features into business processes	tasks are individual pieces of work

Requirement #1: Move around things

There is the need to move blocks and material around the city

copyright agile42, Berlin 2009

User Story #1.1: Tractor

As a house builder I would like to have a Tractor so that I can move around things easily

copyright agile42, Berlin 2009

User Story #1.2: Garage for the Tractor

copyright agile42, Berlin 2009

Requirement #2: Attractive houses

There is the need to have attractive houses, they should represent the ideal house with a garden and a carport

copyright agile42, Berlin 2009

User Story #2.1: House with a front garden

As a citizen I would like to have a house with front garden so that I can enjoy the sun in summer

Acceptance Criteria:

- the garden should be surrounded with a white fence

copyright agile42, Berlin 2009

User Story #2.2: Car port for SUV

As an owner of an SUV I would like to park it beside the house, protected against the bad weather so that it will not be ruined

copyright agile42, Berlin 2009

User Story #2.3: SUV

As a citizen I would like to drive a SUV so that I can load a lot of stuff and drive a bit off-road

copyright agile42, Berlin 2009

Requirement #3: Pedestrian crossing the river

There is the need to have pedestrian crossing the river without getting any risk

copyright agile42, Berlin 2009

User Story #3.1: Pedestrian Bridge

As a pedestrian I want to have a bridge so that I can easily cross the river without getting wet

Acceptance Criteria:

- The bridge has to have arcs
- The bridge has to have a maximum of 3 colors
- The bridge needs to be exactly 36 units long

copyright agile42, Berlin 2009

Requirement #4: Coffee and Ice-cream

There is the need to provide citizen with coffee and ice-cream on the way

copyright agile42, Berlin 2009

User Story #4.1: Kiosk

As a citizen I want to have a Kiosk where I can stop by and have a coffee or and ice-cream so that I can make my children happy

Acceptance Criteria:

- Coffee machine and ice-cream should be visible
- A child with his mother should stay in front

copyright agile42, Berlin 2009

Requirement #5: Build high buildings

There is the need to be able to build high buildings, at least of 2 story

copyright agile42, Berlin 2009

User Story #5.1: Tower crane

As a house builder I want to have a tower crane so that I can easily transport material to the high story of a house

Acceptance Criteria:

- The crane should be stable
- The crane should reach the roof of a 2 story building

copyright agile42, Berlin 2009

Requirement #6: high buildings

There is the need to have at least one building of two story, the city must offer different living possibilities to the citizens

copyright agile42, Berlin 2009

User Story #6.1: 2 story house

As a citizen I would like to have a 2 story house, so that I can have more rooms without having a big piece of land

Acceptance Criteria:

- The house should look like the one in the LEGO catalog

Requirement #7: public transport

The city needs to offer public transportation services, not all citizens are supposed to have their own

copyright agile42, Berlin 2009

User Story #7.1: Bus

As a citizen I would like to take the bus to move around the city so that I can travel without having to

copyright agile42, Berlin 2009

User Story #7.2: Bus stop

As a citizen I would like to have a covered bus stop with seats so that also with bad weather would be comfortable to wait for the bus

copyright agile42, Berlin 2009